Hewlett Packard Enterprise

Measure リファレンスマニュアル

摘要

本書では、Measure パフォーマンスモニターのコマンドおよびプログラミングインターフェイスについて説明します。本書は、HPE NonStop システムのパフォーマンスを監視するシステムアナリストおよびパフォーマンス管理スペシャリストを対象としています。

部品番号: 867649-193
発行: 2017 年 3 月
版数: L15.02、J06.03、H06.03、G06.03、D40.00、およびそれ以降のすべての L シリーズ、J シリーズ、H シリーズ、G シリーズ、および D シリーズの RVU。

© Copyright 2015, 2017 Hewlett Packard Enterprise Development LP

ご注意

本書の内容は、将来予告なしに変更されることがあります。Hewlett Packard Enterprise 製品およびサービスに対する保証については、当該製品およびサービスの保証規定書に記載されています。本書のいかなる内容も、新たな保証を追加するものではありません。本書の内容につきましては万全を期しておりますが、本書中の技術的あるいは校正上の誤り、脱落に対して、責任を負いかねますのでご了承ください。

本書で取り扱っているコンピューターソフトウェアは秘密情報であり、その保有、使用、または複製には、Hewlett Packard Enterprise から使用許諾を得る必要があります。FAR 12.211 および 12.212 に従って、商業用コンピューター・ソフトウェア、コンピューター・ソフトウェア資料、および商業用製品の技術情報は、ベンダー標準の商業用ライセンスのもとで米国政府に使用許諾が付与されます。

他社の Web サイトへのリンクは、Hewlett Packard Enterprise の Web サイトの外に移動します。Hewlett Packard Enterprise は、Hewlett Packard Enterprise の Web サイト以外にある情報を管理する権限を持たず、また責任を負いません。

商標

Intel®、インテル、Itanium®、Pentium®、Intel Inside®、および Intel Inside ロゴは、インテルコーポレーションまたはその子会社のアメリカ合衆国およびその他の国における商標または登録商標です。

Microsoft®および Windows®は、米国および/またはその他の国における Microsoft Corporation の登録商標または商標です。

Adobe®および Acrobat®は、米国 Adobe Systems Incorporated の登録商標です。

Java®および Oracle®は、Oracle および/またはその関連会社の登録商標です。

UNIX®は、The Open Group の登録商標です。

目次

本書について.....17
対象となるリリースバージョンアップデート（RVU）..... 17
対象読者.....17
新規情報と更新内容.....17
867649-193 の新規情報と更新内容..... 17
867649-002 の新規情報と更新内容..... 17
867649-001 の新規情報と更新内容..... 17
523324-023 の新規情報と更新内容..... 17
523324-022R の新規情報と更新内容.....18
523324-022 の新規情報と更新内容..... 18
523324-021 の新規情報と更新内容..... 19
523324-020 の新規情報と更新内容..... 21
523324-019 の新規情報と更新内容..... 21
523324-018 の新規情報と更新内容..... 22
表記規則.....22
一般的な構文の表記.....22
メッセージの表記.....25
管理プログラミングインターフェイスの表記..... 26
関連情報.....27
発行履歴.....27

Measure の概要.....29
動作の概要..... 29
Measure サブシステム.....30
ユーザーインターフェイス.....30
エンティティおよびカウンター.....30
測定の 3 つのステップ.....30
手順 1. 測定を設定する.....31
手順 2. 測定を実施する.....31
手順 3. 測定を検査する.....31
測定の開始と停止の際の考慮事項..... 33

MEASCOM コマンド..... 36
MEASCOM コマンドの概要..... 36
MEASCOM の使用に関する一般的な情報.....39
MEASCOM コマンドの構文規則..... 39
MEASCOM セッションの実行.....41
カスタムの起動ファイルの作成.....42
MEASCOM.....42
MEASCOM の構文.....42
関連コマンド.....43
MEASCOM の使用上の注意.....43
MEASCOM の例.....43
ADD entity-type.....43
ADD entity-type の構文..... 43
関連コマンド.....44
すべての PVU および RVU での ADD entity-type の使用上の注意..... 44
L シリーズの PVU および RVU での ADD entity-type の使用上の注意.....44

H シリーズ以降の PVU および RVU についての ADD entity-type の使用上の注意..... 44
ADD entity-type の例..... 45
ADD COUNTER..... 46
ADD COUNTER の構文..... 46
関連コマンド..... 47
ADD COUNTER の使用上の注意..... 47
ADD COUNTER の例..... 48
ADD MEASUREMENT..... 48
ADD MEASUREMENT の構文..... 49
関連コマンド..... 49
すべての PVU および RVU での ADD MEASUREMENT の使用上の注意..... 49
L シリーズの PVU および RVU での ADD MEASUREMENT の使用上の注意..... 50
H シリーズ以降の PVU および RVU についての ADD MEASUREMENT の使用上の注意..... 50
G シリーズの PVU および RVU についての ADD MEASUREMENT の使用上の注意..... 50
ADD MEASUREMENT の例..... 51
ADD PLOT..... 51
ADD PLOT の構文..... 51
関連コマンド..... 53
ADD PLOT の使用上の注意..... 53
ADD PLOT の例..... 53
ASSUME..... 54
ASSUME の構文..... 54
ASSUME の例..... 55
COMMENTS..... 55
COMMENTS の構文..... 55
関連コマンド..... 56
COMMENTS の使用上の注意..... 56
COMMENTS の例..... 56
DELETE entity-type..... 56
DELETE entity-type の構文..... 56
関連コマンド..... 57
すべての PVU および RVU での DELETE entity-type の使用上の注意..... 57
L シリーズの PVU および RVU での DELETE entity-type の使用上の注意..... 57
H シリーズ以降の PVU および RVU についての DELETE entity-type の使用上の注意..... 57
DELETE entity-type の例..... 57
DELETE COUNTER..... 58
DELETE COUNTER の構文..... 58
関連コマンド..... 59
DELETE COUNTER の例..... 59
DELETE MEASUREMENT..... 59
DELETE MEASUREMENT の構文..... 59
関連コマンド..... 59
DELETE MEASUREMENT の使用上の注意..... 59
DELETE MEASUREMENT の例..... 60
DELETE PLOT..... 60
DELETE PLOT の構文..... 60
関連コマンド..... 60
DELETE PLOT の例..... 61
ENV..... 61
ENV の構文..... 61
ENV の使用上の注意..... 62
ENV の例..... 62
EXIT..... 63
EXIT の構文..... 63
関連コマンド..... 63
EXIT の使用上の注意..... 63

EXIT の例 64
FC 64
FC の構文 64
関連コマンド 64
FC の例 64
HELP 65
HELP の構文 65
HELP の使用上の注意 66
HELP の例 66
HISTORY 67
HISTORY の構文 67
関連コマンド 67
HISTORY の例 67
INFO entity-type 68
INFO entity-type の構文 68
関連コマンド 68
すべての PVU および RVU での INFO entity-type の使用上の注意 68
L シリーズの PVU および RVU での INFO entity-type の使用上の注意 68
H シリーズの PVU および RVU での INFO entity-type の使用上の注意 69
G シリーズの PVU での INFO entity-type の使用上の注意 69
INFO entity-type の例 69
INFO COUNTER 70
INFO COUNTER の構文 70
関連コマンド 70
すべての PVU および RVU での INFO COUNTER の使用上の注意 70
G シリーズの PVU での INFO COUNTER の使用上の注意 70
INFO COUNTER の例 70
INFO MEASUREMENT 71
INFO MEASUREMENT の構文 71
関連コマンド 71
すべての RVU での INFO MEASUREMENT の使用上の注意 72
H シリーズの PVU および RVU での INFO MEASUREMENT の使用上の注意 72
G シリーズの PVU での INFO MEASUREMENT の使用上の注意 72
L シリーズの PVU での INFO MEASUREMENT の使用上の注意 73
INFO MEASUREMENT の例 73
INFO PLOT 74
INFO PLOT の構文 74
関連コマンド 75
INFO PLOT の使用上の注意 75
INFO PLOT の例 75
LIST entity-type 75
LIST entity-type の構文 76
関連コマンド 82
すべての PVU および RVU での LIST entity-type の使用上の注意 83
L シリーズの PVU および RVU での LIST entity-type の使用上の注意 83
H シリーズ以降の PVU および RVU での LIST entity-type の使用上の注意 83
G シリーズの PVU および RVU での LIST entity-type の使用上の注意 84
LIST entity-type の例 84
LIST EXTNAMES 94
LIST EXTNAMES の構文 94
EXTNAMES ファイルの DDL レコード 94
LIST EXTNAMES の例 95
LIST OSSNAMES 95
LIST OSSNAMES の構文 95
OSSNAMES ファイルの DDL レコード 95
DDL レコード記述フィールド 96
LIST OSSNAMES の使用上の注意 96

LIST OSSNAMES の例 97
LIST PLOT 97
LIST PLOT の構文 97
関連コマンド 99
H シリーズ以降の RVU での LIST PLOT の使用上の注意 100
LIST PLOT の例 100
LISTACTIVE entity-type 104
LISTACTIVE entity-type の構文 104
LISTACTIVE エンティティ指定の特殊なケース 108
関連コマンド 113
すべての PVU および RVU での LISTACTIVE entity-type の使用上の注意 113
L シリーズの PVU および RVU での LISTACTIVE entity-type の使用上の注意 114
H シリーズ以降の PVU および RVU での LISTACTIVE entity-type の使用上の注意 114
G シリーズの PVU での LISTACTIVE entity-type の使用上の注意 114
LISTACTIVE entity-type の例 114
LISTALL entity-type 115
LISTALL entity-type の構文 115
関連コマンド 116
すべての PVU および RVU での LISTALL entity-type の使用上の注意 117
L シリーズの PVU および RVU での LISTALL entity-type の使用上の注意 117
H シリーズ以降の PVU および RVU での LISTALL entity-type の使用上の注意 117
G シリーズの PVU および RVU での LISTALL entity-type の使用上の注意 118
LISTALL entity-type の例 118
LISTENAME 123
LISTENAME の構文 123
LISTENAME の例 123
LISTGNAME 123
LISTGNAME の構文 124
LISTGNAME の使用上の注意 124
LISTGNAME の例 125
LISTPNAME 125
LISTPNAME の構文 125
LISTPNAME の使用上の注意 125
LISTPNAME の例 126
LOG 126
LOG の構文 126
LOG の使用上の注意 126
LOG の例 127
OBEY 127
OBEY の構文 127
OBEY の使用上の注意 127
OBEY の例 127
OSSPATH 127
OSSPATH の構文 128
OSSPATH の例 128
OUT 128
OUT の構文 128
OUT の使用上の注意 128
OUT の例 128
PAGESIZE 128
PAGESIZE の構文 129
PAGESIZE の使用上の注意 129
PAGESIZE の例 129
RESET PLOT 129
RESET PLOT の構文 129
関連コマンド 130
RESET REPORT 130

RESET REPORT の構文....130
関連コマンド....132
RUN....132
RUN の構文....132
RUN の使用上の注意....132
RUN の例....133
SET PLOT....133
SET PLOT の構文....133
関連コマンド....135
SET PLOT の例....135
SET REPORT....137
SET REPORT の構文....137
関連コマンド....143
SET REPORT の例....143
SETPROMPT....145
SETPROMPT の構文....145
SETPROMPT の使用上の注意....146
SETPROMPT の例....146
SHOW PLOT....147
SHOW PLOT の構文....147
関連コマンド....147
SHOW PLOT の使用上の注意....147
SHOW PLOT の例....147
SHOW REPORT....148
SHOW REPORT の構文....148
関連コマンド....148
すべての PVU および RVU での SHOW REPORT の使用上の注意....148
G シリーズの PVU での SHOW REPORT の使用上の注意....148
SHOW REPORT の例....149
SQLCATALOG....149
SQLCATALOG の構文....149
SQLCATALOG の例....149
SQLSCHEMA....149
SQLSCHEMA の構文....149
SQLSCHEMA の例....150
START MEASSUBSYS....150
START MEASSUBSYS の構文....150
関連コマンド....150
すべての PVU および RVU での START MEASSUBSYS の使用上の注意....150
H シリーズ以降の PVU および RVU についての START MEASSUBSYS の使用上の注意....151
G シリーズの PVU での START MEASSUBSYS の使用上の注意....152
START MEASUREMENT....153
START MEASUREMENT の構文....153
関連コマンド....156
すべての PVU および RVU での START MEASUREMENT の使用上の注意....156
H シリーズ以降の PVU についての START MEASUREMENT の使用上の注意....156
G シリーズの PVU での START MEASUREMENT の使用上の注意....157
START MEASUREMENT の例....157
STATUS MEASSUBSYS....157
STATUS MEASSUBSYS の構文....157
STATUS MEASSUBSYS の使用上の注意....158
STATUS MEASSUBSYS の例....158
STATUS MEASUREMENT....159
STATUS MEASUREMENT の構文....159
STATUS MEASUREMENT の使用上の注意....160
STATUS MEASUREMENT の例....160

STOP MEASSUBSYS....160
STOP MEASSUBSYS の構文....160
関連コマンド....161
STOP MEASSUBSYS の使用上の注意....161
STOP MEASSUBSYS の例....161
STOP MEASUREMENT....161
STOP MEASUREMENT の構文....161
関連コマンド....162
STOP MEASUREMENT の例....163
SWAPVOL....163
SWAPVOL の構文....163
SWAPVOL の使用上の注意....163
SYSTEM....163
SYSTEM の構文....163
SYSTEM の例....163
TIME....163
TIME の構文....163
TIME の例....164
VOLUME....164
VOLUME の構文....164
VOLUME の例....164
WARNINGS....164
WARNINGS の構文....164
関連コマンド....165
WARNINGS の例....165
!（感嘆記号）....165
!（感嘆記号）の構文....165
関連コマンド....166
!（感嘆記号）の例....166

エンティティおよびカウンター....167
カウンターの概要....168
カウンター値の解釈....168
データファイルのエラーの識別....175
一般的なエンティティヘッダーフィールド....177
DDL ヘッダー（ZMS 形式）....178
DDL ヘッダー（Legacy 形式）....178
DDL ヘッダーフィールドの定義....179
オープンシステムサービス（OSS）に対する Measure のサポート....181
OSS ファイルパス名の処理....181
Guardian ファイル名の再利用....181
作成バージョンシリアル番号（CRVSN）の処理....182
Measure ID（MID）....182
OSS ジャーナルセグメント....182
エンティティレポートの形式....183
ANSI SQL 名に対する Measure のサポート....183
ANSI SQL 名の処理....183
SQL ジャーナルセグメント....184
エンティティレポートの形式....184
DLL に対する Measure のサポート....184
Measure G12 以降の PVU は DLL の測定をサポート....184
集計データの測定....185
このマニュアルの DLL に関する追加情報....185
ZMS 形式のレコードへのアクセス（MEASDDLZ）....186
CLUSTER....186

CLUSTER エンティティのエンティティ指定構文..... 187
CLUSTER エンティティの DDL レコード（ZMS 形式）..... 188
CLUSTER エンティティの DDL レコード（Legacy 形式）..... 188
CLUSTER エンティティの共通の ID フィールドの DDL 定義..... 189
CLUSTER エンティティの共通のカウンターフィールドの DDL 定義.....189
CLUSTER エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義..... 190
すべての CLUSTER エンティティの使用上の注意.....191
H シリーズ以降の CLUSTER エンティティの使用上の注意..... 191
G シリーズの CLUSTER エンティティの使用上の注意..... 191
CONTROLLER.....191
CONTROLLER エンティティのエンティティ指定構文.....192
CONTROLLER エンティティの DDL レコード（Legacy 形式）..... 193
CONTROLLER エンティティの Legacy 形式のみの ID フィールドの DDL 定義..... 193
CONTROLLER エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義.....194
CONTROLLER エンティティの使用上の注意.....194
CPU.....194
CPU エンティティのエンティティ指定構文.....195
CPU エンティティの DDL レコード（ZMS 形式）..... 195
CPU エンティティの DDL レコード（Legacy 形式）..... 197
CPU エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義..... 199
CPU エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義.....200
CPU エンティティの Legacy 形式のみの ID フィールドの DDL 定義.....208
CPU エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義..... 209
すべての CPU エンティティの使用上の注意.....211
J シリーズの CPU エンティティの使用上の注意..... 212
H シリーズ以降の CPU エンティティの使用上の注意..... 212
G シリーズの CPU エンティティの使用上の注意..... 213
D シリーズの CPU エンティティの使用上の注意.....214
DEVICE.....214
DEVICE エンティティのエンティティ指定構文.....214
DEVICE エンティティの DDL レコード（ZMS 形式）.....216
DEVICE エンティティの DDL レコード（Legacy 形式）.....217
DEVICE エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義..... 219
DEVICE エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義.....221
DEVICE エンティティの Legacy 形式のみの ID フィールドの DDL 定義..... 223
DEVICE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義.....223
すべての DEVICE エンティティの使用上の注意.....224
H シリーズ以降の DEVICE エンティティの使用上の注意.....224
G シリーズの DEVICE エンティティの使用上の注意.....225
D シリーズの DEVICE エンティティの使用上の注意.....225
例.....225
DISC.....226
DISC エンティティのエンティティ指定構文.....226
DISC エンティティの DDL レコード（ZMS 形式）..... 228
DISC エンティティの DDL レコード（Legacy 形式）..... 230
DISC エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義..... 232
DISC エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義.....235
DISC エンティティの Legacy 形式のみの ID フィールドの DDL 定義.....242
DISC エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義..... 242
すべての DISC エンティティの使用上の注意.....246
H シリーズ以降の DISC エンティティの使用上の注意..... 247
G シリーズの DISC エンティティの使用上の注意..... 247
D シリーズの DISC エンティティの使用上の注意.....249

例....249
DISCOPEN....249
DISCOPEN エンティティのエンティティ指定構文....250
DISCOPEN エンティティの DDL レコード（ZMS 形式）....251
DISCOPEN エンティティの DDL レコード（Legacy 形式）....252
DISCOPEN エンティティの共通の ID フィールドの DDL 定義....254
DISCOPEN エンティティの共通のカウンターフィールドの DDL 定義....255
すべての DISCOPEN エンティティの使用上の注意....257
G シリーズの DISCOPEN エンティティの使用上の注意....258
例....259
DISKFILE....259
DISKFILE エンティティのエンティティ指定構文....260
DISKFILE エンティティの DDL レコード（ZMS 形式）....261
DISKFILE エンティティの DDL レコード（Legacy 形式）....262
DISKFILE エンティティの共通の ID フィールドの DDL 定義....264
DISKFILE エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義....265
DISKFILE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義....268
すべての DISKFILE エンティティの使用上の注意....268
G シリーズの DISKFILE エンティティの使用上の注意....269
例....270
ENDPOINT....270
ENDPOINT エンティティのエンティティ指定構文....271
ENDPOINT エンティティの DDL レコード（ZMS 形式）....272
ENDPOINT エンティティの DDL レコード（Legacy 形式）....273
ENDPOINT エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義....275
ENDPOINT エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義....276
ENDPOINT エンティティの Legacy 形式のみの ID フィールドの DDL 定義....277
ENDPOINT エンティティの使用上の注意....277
ENDPOINT IPC および RIPC の使用上の注意....277
CLIM の使用上の注意....278
ENDPOINT クラスターの測定の構成の例....280
CLIM の測定の構成の例....280
FABRIC....281
FABRIC エンティティのエンティティ指定構文....281
FABRIC エンティティの DDL レコード....282
FABRIC エンティティの ID フィールドの DDL 定義....283
FABRIC エンティティのカウンターフィールドの DDL 定義....283
FABRIC エンティティの使用上の注意....284
FILE....284
Measure と OSS ファイルのオープン....285
OSS の命名規則....286
ANSI SQL の命名規則....288
FILE エンティティのエンティティ指定構文....288
FILE エンティティの DDL レコード（ZMS 形式）....292
FILE エンティティの DDL レコード（Legacy 形式）....293
FILE エンティティの共通の ID フィールドの DDL 定義....295
FILE エンティティの共通のカウンターフィールドの DDL 定義....298
FILE エンティティの Legacy 形式のみの ID フィールドの DDL 定義....303
FILE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義....303
すべての FILE エンティティの使用上の注意....303
H シリーズ以降の FILE エンティティの使用上の注意....304
G シリーズの FILE エンティティの使用上の注意....304
コマンドの例：OSS ファイルのオープン....305
LINE....309
LINE エンティティのエンティティ指定構文....309

LINE エンティティの DDL レコード（ZMS 形式）.....311
LINE エンティティの DDL レコード（Legacy 形式）.....311
LINE エンティティの共通の ID フィールドの DDL 定義.....312
LINE エンティティの共通のカウンターフィールドの DDL 定義.....313
LINE エンティティの Legacy 形式のみの ID フィールドの DDL 定義.....316
LINE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義.....316
H シリーズ以降の LINE エンティティの使用上の注意.....317
G シリーズの LINE エンティティの使用上の注意.....317
NETLINE.....317
NETLINE エンティティのエンティティ指定構文.....317
NETLINE エンティティの DDL レコード（ZMS 形式）.....319
NETLINE エンティティの DDL レコード（Legacy 形式）.....320
NETLINE エンティティの共通の ID フィールドの DDL 定義.....321
NETLINE エンティティの共通のカウンターフィールドの DDL 定義.....322
NETLINE エンティティの Legacy 形式のみの ID フィールドの DDL 定義.....325
NETLINE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義.....325
すべての NETLINE エンティティの使用上の注意.....326
H シリーズ以降の NETLINE エンティティの使用上の注意.....326
G シリーズの NETLINE エンティティの使用上の注意.....326
OPDISK.....327
OPDISK エンティティのエンティティ指定構文.....327
OPDISK エンティティの DDL レコード（ZMS 形式）.....328
OPDISK エンティティの DDL レコード（Legacy 形式）.....329
OPDISK エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義.....330
OPDISK エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義.....331
OPDISK エンティティの Legacy 形式のみの ID フィールドの DDL 定義.....333
OPDISK エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義.....333
すべての OPDISK エンティティの使用上の注意.....334
OSSCPU.....334
OSSCPU エンティティのエンティティ指定構文.....335
OSSCPU エンティティの DDL レコード（ZMS 形式）.....335
OSSCPU エンティティの DDL レコード（Legacy 形式）.....337
OSSCPU エンティティの共通の ID フィールドの DDL 定義.....338
OSSCPU エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義.....339
OSSCPU エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義.....348
OSSCPU エンティティの使用上の注意.....349
OSSNS.....350
OSSNS エンティティのエンティティ指定構文.....350
OSSNS エンティティの DDL レコード（ZMS 形式）.....351
OSSNS エンティティの DDL レコード（Legacy 形式）.....352
OSSNS エンティティの共通の ID フィールドの DDL 定義.....353
OSSNS エンティティの共通のカウンターフィールドの DDL 定義.....354
OSSNS エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義.....356
OSSNS エンティティの使用上の注意.....357
PROCESS.....358
PROCESS エンティティのエンティティ指定構文.....358
PROCESS エンティティの DDL レコード（ZMS 形式）.....360
PROCESS エンティティの DDL レコード（Legacy 形式）.....362
PROCESS エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義.....365
PROCESS エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義.....370
PROCESS エンティティの Legacy 形式のみの ID フィールドの DDL 定義.....382
PROCESS エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義.....382
All PROCESS エンティティの使用上の注意.....384
J シリーズ PROCESS エンティティの使用上の注意.....384

H シリーズ以降の PROCESS エンティティの使用上の注意 384
G シリーズ PROCESS エンティティの使用上の注意 386
PROCESSH 387
PROCESSH エンティティのエンティティ指定構文 388
PROCESSH エンティティの DDL レコード（ZMS 形式） 391
PROCESSH エンティティの DDL レコード（Legacy 形式） 393
PROCESSH エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義 394
PROCESSH エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義 396
PROCESSH エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義 397
H シリーズ以降の PROCESSH エンティティの使用上の注意 397
G シリーズ PROCESSH エンティティの使用上の注意 398
PROCESSH 測定の例 398
SERVERNET 399
SERVERNET エンティティのエンティティ指定構文 400
SERVERNET エンティティの DDL レコード（ZMS 形式） 402
SERVERNET エンティティの DDL レコード（Legacy 形式） 403
SERVERNET エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義 405
SERVERNET エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義 407
SERVERNET エンティティの Legacy 形式のみの ID フィールドの DDL 定義 409
SERVERNET エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義 410
SERVERNET エンティティの使用上の注意 410
SERVERNET IPC および RIPC の使用上の注意 411
CLIM の使用上の注意 412
SERVERNET クラスターの測定の設定の例 413
CLIM の測定の設定の例 413
SQLPROC 414
SQLPROC エンティティのエンティティ指定構文 414
SQLPROC エンティティの DDL レコード（ZMS 形式） 416
SQLPROC エンティティの DDL レコード（Legacy 形式） 417
SQLPROC エンティティの共通の ID フィールドの DDL 定義 418
SQLPROC エンティティの共通のカウンターフィールドの DDL 定義 420
新しい形式の SQLPROC エンティティの使用上の注意 421
SQLSTMT 421
SQLSTMT エンティティのエンティティ指定構文 422
SQLSTMT エンティティの DDL レコード（ZMS 形式） 425
SQLSTMT エンティティの DDL レコード（Legacy 形式） 426
SQLSTMT エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義 428
SQLSTMT エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義 431
SQLSTMT エンティティの Legacy 形式のみの ID フィールドの DDL 定義 433
SQLSTMT エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義 433
すべての SQLSTMT エンティティの使用上の注意 434
H シリーズ以降の SQLSTMT エンティティの使用上の注意 434
G シリーズ SQLSTMT エンティティの使用上の注意 434
例 435
SYSTEM 435
SYSTEM エンティティのエンティティ指定構文 436
すべての SYSTEM エンティティの DDL レコード（ZMS 形式） 436
すべての SYSTEM エンティティの DDL レコード（Legacy 形式） 437
SYSTEM エンティティの共通の ID フィールドの DDL 定義 437
SYSTEM エンティティの共通のカウンターフィールドの DDL 定義 438
All SYSTEM エンティティの使用上の注意 438

TERMINAL....439
TERMINAL エンティティのエンティティ指定構文....439
TERMINAL エンティティの DDL レコード（ZMS 形式）....440
TERMINAL エンティティの DDL レコード（Legacy 形式）....441
TERMINAL エンティティの共通の ID フィールドの DDL 定義....442
TERMINAL エンティティの共通のカウンターフィールドの DDL 定義....442
TERMINAL エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義....443
すべての TERMINAL エンティティの使用上の注意....444
H シリーズ以降の TERMINAL エンティティの使用上の注意....444
G シリーズ TERMINAL エンティティの使用上の注意....444
TMF....444
TMF エンティティのエンティティ指定構文....445
TMF エンティティの DDL レコード（ZMS 形式）....445
TMF エンティティの DDL レコード（Legacy 形式）....446
TMF エンティティの共通のカウンターフィールドの DDL 定義....446
TMF エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義....448
すべての TMF エンティティの使用上の注意....448
USERDEF....449
USERDEF エンティティのエンティティ指定構文....449
USERDEF エンティティの DDL レコード（ZMS 形式）....451
USERDEF エンティティの DDL レコード（Legacy 形式）....452
USERDEF エンティティの共通の ID フィールドの DDL 定義....453
USERDEF エンティティの ZMS 形式のみのカウンターフィールドの DDL 定義....456
USERDEF エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義....456
すべての USERDEF エンティティの使用上の注意....456
L シリーズ USERDEF エンティティの使用上の注意....456
H シリーズ以降の USERDEF エンティティの使用上の注意....456
G シリーズ USERDEF エンティティの使用上の注意....457

Measure 呼び出し可能プロシージャー....458

Measure プロシージャーの一覧....458
Measure プロシージャーの概要....460
Measure レコード（DDL）での読み取り....460
Legacy 形式と ZMS 形式のレコード....460
宣言ファイルでの読み取り....461
Measure 制御ブロックへの領域の割り当て....461
エンティティ記述子の指定....461
構成テーブルの作成....463
Measure プロシージャーのエラーコードの解釈....464
MEASDDLS と MEASCHMA での新しい構造との互換性の維持....464
タイマーセルの使用....464
MEAS_ADJUSTZMSRECORD_....465
使用上の注意....466
MEAS_ALLOCATE_TIMERCELLS_....466
MEAS_BUMP_TIMERCELL_....467
MEASCLOSE....468
MEAS_CODERANGENAME_DEMANGLE_....468
MEASCONFIGURE....469
関連プロシージャー....471
MEASCONTROL....471
MEASCOUNTERBUMP....473
MEASCOUNTERBUMPINIT....474
MEAS_DEALLOCATE_TIMERCELLS_....475
MEAS_GETDESCINFO_....475
使用上の注意....477

例....478
MEASGETVERSION....478
MEASINFO....481
MEASINFO の使用上の注意....484
MEASLISTCONFIG....485
使用上の注意....487
MEASLISTENAME....487
MEASLISTENAME の使用上の注意....490
例....491
MEASLISTEXTNAMES....491
MEASLISTEXTENAMES の使用上の注意....492
MEASLISTGNAME....492
MEASLISTGNAME の使用上の注意....495
例....495
MEASLISTOSSNAMES....496
MEASLISTOSSNAMES の使用上の注意....497
MEASLISTPNAME....497
MEASLISTPNAME の使用上の注意....499
MEASMONCONTROL....499
MEASMONCONTROL の使用上の注意....501
MEASMONSTATUS....501
使用上の注意....503
MEASOPEN....503
MEASOPEN 使用上の注意....506
MEASREAD....507
使用上の注意....509
MEASREAD_DIFF_....509
MEASREAD_DIFF_の使用上の注意....515
MEASREADACTIVE....515
MEASREADACTIVE の使用上の注意....518
MEAS_READACTIVE_....518
使用上の注意....521
MEAS_READACTIVE_MANY_....521
使用上の注意....523
MEASREADCONF....524
使用上の注意....528
MEAS_RETRIEVE_TIMERCELLS_....528
MEAS_SQL_MAP_INIT_....529
使用上の注意....530
例....530
MEAS_SQL_MAP_STOP_....530
MEAS_SQLNAME_COMPARE_....531
使用上の注意....532
MEAS_SQLNAME_RESOLVE_....532
使用上の注意....534
MEAS_SQLNAME_SCAN_....534
使用上の注意....534
例....535
MEASSTATUS....535
使用上の注意....538
MEASWRITE_DIFF_....538
MEASWRITE_DIFF_の使用上の注意....544

エンティティ記述子....545
CPU^DESC....547

CTRL^DESC....548
DEVICE^DESC....549
DEVICE^CLIM^DESC....550
DEVICE^SVNET^DESC....553
DEVICE^SVNET^DESC^G05....555
DISKFILE^DESC....557
DISKFILE^OSS^DESC....558
DISKFILE^ANSI^DESC....559
ENDPOINT_DESC....560
FABRIC_DESC....562
FILE^OPEN^DESC....563
FILE^OPEN^DESC^D10....566
FILE^OPEN^OSS^DESC....567
FILE^OPEN^OSS^DESC^D10....567
FILE^OPEN^ANSI^DESC....569
OPDISK^DESC....570
PROCESS^DESC....571
PROCESS^OSS^DESC....573
PROCESSH^DESC....574
CODE^SPACE^DESC....576
PROCESSH^OSS^DESC....577
CODE^SPACE^OSS^DESC....578
SQLPROC^DESC....579
SQLPROC^OSS^DESC....580
SQLSTMT^DESC....580
SQLSTMT^OSS^DESC....582
SQLSTMT^ANSI^DESC....583
SVNET^DESC....584
SYSTEM^DESC....587
USERDEF^DESC....588
COUNTER^DESC....589
USERDEF^OSS^DESC....590
WAN^DESC....591

Web サイト....593

サポートと他のリソース....594
Hewlett Packard Enterprise サポートへのアクセス....594
アップデートへのアクセス....594
カスタマーセルフリペア（CSR）....595
リモートサポート（HPE 通報サービス）....595
保証情報....595
規定に関する情報....595
ドキュメントに関するご意見、ご指摘....596

エラーメッセージ....597

エラーコード....627

サブシステムファイル....669
C および C++言語の使用上の注意....670

プロセス識別番号 670

Measure データファイルツール（MEASFT） 672

INFO コマンド 672
構文 672
例 673
SPLIT コマンド 675
構文 675
SPLIT コマンドの使用上の注意 679
例 680
HELP コマンド 683
構文 683
MEASFT のエラーメッセージと警告メッセージ 683

新規情報と更新内容の古いエントリー 714

H06.25/J06.14 RVU に関する新規情報と更新内容（523324-017） 714
H06.24/J06.13 RVU の新規情報と更新内容（523324-016） 716
H06.23/J06.12 RVU に関する新規情報と更新内容（523324-015） 716
CLIM 接続デバイスのパーティション番号のサポート 716
ロックページカウンターのサポート 717
その他 717

本書について

このマニュアルでは、Measure パフォーマンスモニターのコマンドおよびプログラミングインターフェイスについて説明します。

対象となるリリースバージョンアップデート（RVU）

本書は、改訂版で別途明示されるまで、L15.02 以降のすべての L シリーズの RVU、J06.03 以降のすべての J シリーズの RVU、H06.03 以降のすべての H シリーズの RVU、G06.03 以降のすべての G シリーズ RVU、D40.00 以降のすべての D シリーズの RVU を対象とします。

対象読者

このマニュアルは、HPE NonStop システムのパフォーマンスを監視するシステムアナリストおよびパフォーマンス管理スペシャリストを対象としています。

新規情報と更新内容

867649-193 の新規情報と更新内容

- Measure がジャーナルセグメントを構築する方法に関する情報について、**OSS ジャーナルセグメント**および **SQL ジャーナルセグメント**が更新されました。
- **PROCESSH** の説明にコード例に関する情報が追加されました。
- **SERVERNET エンティティのエンティティ指定構文**が更新され、構文を訂正されました。
- 参照が一般的な I/O に当てはまるように、「InfiniBand」という用語が削除されました。
- **エラーコード**：3272 および 3310 が更新され、3351 および 3352 が追加されました。

867649-002 の新規情報と更新内容

- L16.05 の**すべての PVU および RVU での LIST entity-type の使用上の注意**(83 ページ)および **L シリーズの PVU および RVU での LISTALL entity-type の使用上の注意**(117 ページ)が更新されました。
- カウンター **FILE-BUSY-TIME**(299 ページ)の説明が更新されました。

867649-001 の新規情報と更新内容

- J06.20 の**すべての PVU および RVU での LIST entity-type の使用上の注意**(83 ページ)および **L シリーズの PVU および RVU での LISTALL entity-type の使用上の注意**(117 ページ)が更新されました。
- カウンター **FILE-BUSY-TIME**(299 ページ)の説明が更新されました。

523324-023 の新規情報と更新内容

- **MEASCOM コマンド**(36 ページ)
  - 以下のセクションが更新され、list-option の COUNT、FIRST、および RECORD-NUM が追加されました。
    - **LIST entity-type**(75 ページ)
    - **LISTALL entity-type**(115 ページ)
    - **RESET REPORT**(130 ページ)
    - **SET REPORT**(137 ページ)
    - **SHOW REPORT**(148 ページ)
  - 以下のセクションが更新され、COUNT、FIRST、および RECORD-NUM の使用上の注意が追加されました。

  - **すべての PVU および RVU での LIST entity-type の使用上の注意**(83 ページ)
  - **すべての PVU および RVU での LISTALL entity-type の使用上の注意**(117 ページ)
- 以下のセクションが追加され、新しいレポート形式の CSV、JSON、および XML が追加されました。
  - **LIST entity-type**(75 ページ)
  - **LISTACTIVE entity-type**(104 ページ)
  - **LISTALL entity-type**(115 ページ)
  - **SET REPORT**(137 ページ)
- **エンティティおよびカウンター**(167 ページ)
- ボリュームごとの同時ファイルオープンの数を増やすことができるよう変更されたため、以下のセクションが更新されました。
  - **DISCOPEN エンティティの DDL レコード（ZMS 形式）**(251 ページ)
  - **DISCOPEN エンティティの共通の ID フィールドの DDL 定義**(254 ページ)
- **PROCESS エンティティの DDL レコード（ZMS 形式）**(360 ページ)および **PROCESS エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義**(370 ページ)のカウンターフィールドの DDL 定義が更新され、新しいカウンターが追加されました。
- 今後カウンターフィールドが追加される可能性があるため、以下のセクションが更新されました。
  - **DISKFILE エンティティの DDL レコード（ZMS 形式）**(261 ページ)
  - **SQLPROC エンティティの DDL レコード（ZMS 形式）**(416 ページ)
- **エラーメッセージ**(597 ページ)

  エラーメッセージ 3128、3129、3130 が追加されました。

  エラーメッセージ 3431、3432、3433、3434、3435、3436、3437、3440、3441 が追加されました。

523324-022R の新規情報と更新内容

Hewlett Packard Enterprise のリファレンスの更新。

523324-022 の新規情報と更新内容

- MEASREACTIVE および MEAS_REACTIVE_のスペルが MEASREADACTIVE および MEAS_READACTIVE_に訂正されました。
- **LISTACTIVE entity-type**(104 ページ)の構文が訂正されました。
- **SET REPORT**(137 ページ)に詳細が追加されました。
- **データファイルのエラーの識別**(175 ページ)および「Legacy 形式の ERROR フィールドの値」が更新され、エラー値 8 が追加されました。
- **DDL ヘッダーフィールドの定義**(179 ページ)の **FORMAT-VERSION**(179 ページ)の説明が更新されました。
- **CPU エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義**(199 ページ)で、**MEM-MB**(200 ページ)フィールドの値に関する詳細が追加されました。
- **CPU エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義**(200 ページ)が更新され、**LINK-LARGE-MSGS**(203 ページ)、**LINK-READLINK-MSGS**(203 ページ)、**READLINKCACHE-ALL**(205 ページ)、**READLINKCACHE-CTRL**(205 ページ)、および **READLINKCACHE-NONE**(205 ページ)の L シリーズの説明が更新されました。
- **DISC**(226 ページ)エンティティタイプの説明が更新されました。
- **DISCOPEN**(249 ページ)エンティティの説明が更新されました。
- **DISCOPEN エンティティの共通のカウンターフィールドの DDL 定義**(255 ページ)で以下のフィールドの説明が更新されました。
  - **CACHE-WRITE-CLEANS**(256 ページ)エンティティタイプの説明が更新されました。
  - **CACHE-WRITE-HITS**(256 ページ)エンティティタイプの説明が更新されました。

    - **DRIVER-INPUT-CALLS**(256 ページ)エンティティタイプの説明が更新されました。
    - **DRIVER-OUTPUT-CALLS**(256 ページ)エンティティの説明が更新されました。
- **すべての DISCOPEN エンティティの使用上の注意**(257 ページ)で、DISCOPEN を使用する場合と DISKFILE を使用する場合の留意事項が更新されました。
- **DISKFILE**(259 ページ)エンティティの説明が更新されました。
- **DISKFILE エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義**(265 ページ)で以下のフィールドの説明が更新されました。
    - **CACHE-WRITE-CLEANS**(265 ページ)エンティティタイプの説明が更新されました。
    - **CACHE-WRITE-HITS**(265 ページ)エンティティタイプの説明が更新されました。
    - **DRIVER-INPUT-CALLS**(266 ページ)エンティティタイプの説明が更新されました。
    - **DRIVER-OUTPUT-CALLS**(266 ページ)エンティティタイプの説明が更新されました。
- **すべての DISKFILE エンティティの使用上の注意**(268 ページ)エンティティタイプの説明の留意事項が更新されました。
- **ENDPOINT エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義**(275 ページ)で、**REMOTE-CLUSTER**(276 ページ)が追加され、**REMOTE-CPU**(276 ページ)が更新されました。
- **FILE エンティティの共通の ID フィールドの DDL 定義**(295 ページ)の **MESSAGE-BYTES**(300 ページ)フィールドの説明が更新されました。
- **SYSTEM-PROCESS**(368 ページ)フィールドの説明が更新されました。
- **All PROCESS エンティティの使用上の注意**(384 ページ)の、CPU のすべての PROCESS エンティティの IPU-BUSY-TIME と CPU-BUSY-TIME の相違に関する情報が更新されました。**すべての CPU エンティティの使用上の注意**(211 ページ)からこのセクションへのポインターが追加されました。
- **ENDPOINT_DESC**(560 ページ)および **SVNET^DESC**(584 ページ)の *ctrl^type* の説明が更新されました。
- **エラーメッセージ**(597 ページ)のエラーメッセージ 3289 の原因と回復処置が更新されました。
- **MEASFT のエラーメッセージと警告メッセージ**(683 ページ)に、以下の新しいエラーメッセージと、各メッセージの説明、原因、結果、および回復処置が追加されました。
    - MEASFT 3041（ERR_SEG_ALLOCATE64）
    - MEASFT 3042（ERR_SEG_RESIZE64）
    - MEASFT 3043（ERR_SIG_HANDLER）
- 新規情報と更新内容の古い項目は**新規情報と更新内容の古いエントリー**(714 ページ)に移されました。

523324-021 の新規情報と更新内容

- **ADD COUNTER**(46 ページ)の H07、J05、L シリーズ以降の PVU の大きい USERDEF カウンター値の使用上の注意が更新されました。
- **対象となるリリースバージョンアップデート（RVU）**(17 ページ)が更新され、L シリーズが追加されました。
- 以下のコマンドが更新され、構文と「L シリーズの PVU および RVU の使用上の注意」のサブセクションに FABRIC エンティティが追加されました。
    - **ADD entity-type**(43 ページ)
    - **ASSUME**(54 ページ)
    - **DELETE entity-type**(56 ページ)
    - **INFO entity-type**(68 ページ)
    - **LIST entity-type**(75 ページ)
    - **LISTALL entity-type**(115 ページ)
    - **LISTACTIVE entity-type**(104 ページ)
- **ADD MEASUREMENT**(48 ページ)が更新され、L シリーズの RVU でサポートされるデータファイルのバージョンを記載するために「L シリーズの PVU および RVU の使用上の注意」が追加されました。
- 以下のコマンドに L シリーズの例が追加されました。
    - INFO（**INFO entity-type の例**(69 ページ)）
    - INFO MEASUREMENT（**INFO MEASUREMENT の例**(73 ページ)）

    - LIST（**LIST entity-type の例**(84 ページ)）
    - LIST PLOT（**LIST PLOT の例**(100 ページ)）
- LISTALL（**LISTALL entity-type の例**(118 ページ)）に新しい例が追加されました。
- タイマーセル機能は L シリーズの RVU では無効であることを指摘するためと、代替カウンターに注意を向けるために、**タイマーセルカウンター**(175 ページ)が更新されました。
- 明確にするために**すべての CPU エンティティの使用上の注意**(211 ページ)が書き換えられました。
- **DEVICE エンティティのエンティティ指定構文**(214 ページ)が更新され、フィールドの説明が変更されました。
- 新しい **FABRIC**(281 ページ)エンティティが追加されました。
- **PROCESS エンティティの DDL レコード（ZMS 形式）**(360 ページ)が更新され、新しいカウンターが追加されました。
- **FLAGS**(366 ページ)および **SYSTEM-PROCESS**(368 ページ)のフィールドの説明が更新されました。
- **OPDISK**(327 ページ)が更新され、オプティカルディスクのサポートされているリリースが明記されました。
- **PROCESS エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義**(370 ページ)が更新され、次の新しいカウンターが追加されました。**INTERRUPTS**(374 ページ)、**X-INTERRUPTS**(381 ページ)、および **Y-INTERRUPTS**(382 ページ)。
- **PROCESSH エンティティのエンティティ指定構文**(388 ページ)および **PROCESSH エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義**(396 ページ)が更新され、TSN/X が追加されました。
- **SERVERNET**(399 ページ)および **SERVERNET エンティティのエンティティ指定構文**(400 ページ)が更新され、L シリーズの RVU が追加されました。
- L シリーズでサポートされているノードクラスに言及するため、**NODE-CLASS-S**(275 ページ)が更新されました。
- **SERVERNET エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義**(405 ページ)が更新され、L シリーズの RVU が追加されました。
- L シリーズの RVU での測定には使用されないことを指摘するため、**PF**(406 ページ)が更新されました。
- **SERVERNET エンティティの DDL レコード（ZMS 形式）**(402 ページ)にフラグフィールドが追加され、**SERVERNET エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義**(405 ページ)に定義が追加されました。
- **CLIM の使用上の注意**(278 ページ)が更新され、L シリーズが追加されました。
- **L シリーズ USERDEF エンティティの使用上の注意**(456 ページ)が **USERDEF**(449 ページ)に追加されました。
- **エンティティのセットの測定**(462 ページ)が更新され、空白と null の入力に関する情報が追加されました。
- タイマーセル機能は L シリーズの RVU では無効であることを指摘するためと、代替カウンターに注意を向けるために、**タイマーセルの使用**(464 ページ)が更新されました。
- **MEAS_ALLOCATE_TIMERCELLS_**(466 ページ)、**MEAS_BUMP_TIMERCELL_**(467 ページ)、**MEAS_DEALLOCATE_TIMERCELLS_**(475 ページ)、および **MEAS_RETRIEVE_TIMERCELLS_**(528 ページ)が更新され、タイマーセル機能は L シリーズの RVU では無効であり、これらのプロシージャーはエラーを返し、ポインターは代替の USERDEF カウンターに位置付けられることが記載されました。
- 以下のプロシージャーが更新され、L シリーズの RVU ではこれらのプロシージャーは FABRIC 記述子をサポートし、SERVERNET エンティティ記述子の GMS、ポート、およびファイバーのフィールドはサポートされなくなったことが記載されました。
    - **MEASCONFIGURE**(469 ページ)
    - **MEASINFO**(481 ページ)
    - **MEASLISTCONFIG**(485 ページ)
    - **MEASREAD**(507 ページ)
    - **MEASREAD_DIFF_**(509 ページ)
    - **MEASREADACTIVE**(515 ページ)
    - **MEAS_READACTIVE_MANY_**(521 ページ)

    - **MEASREADCONF**(524 ページ)
    - **MEASWRITE_DIFF_**(538 ページ)
- **MEASCONFIGURE**(469 ページ)が更新され、FABRIC エンティティ記述子が追加されました。
- **MEASINFO**(481 ページ)、**MEASREADCONF**(524 ページ)、および **MEASSTATUS**(535 ページ)が更新され、エンティティの表に FABRIC エンティティが追加されました。
- **MEASOPEN**(503 ページ)の使用上の注意が更新され、L シリーズの RVU でサポートされているデータファイルのバージョンが記載されました。
- 以下のプロシージャーが更新され、FABRIC エンティティと Legacy 形式のレコードが両方とも指定されると、これらのプロシージャーはエラーを返すことが記載されました。
    - **MEASREAD**(507 ページ)
    - **MEASREAD_DIFF_**(509 ページ)
    - **MEASREADACTIVE**(515 ページ)
    - **MEAS_READACTIVE_MANY_**(521 ページ)
    - **MEASWRITE_DIFF_**(538 ページ)
- **MEASREAD**(507 ページ)が更新され、L シリーズの情報が追加されました。
- **MEASREAD_DIFF_**(509 ページ)が更新され、Measure H07 が追加されました。
- **MEASREADACTIVE**(515 ページ)が更新され、FABRIC エンティティが追加されました。
- **MEASWRITE_DIFF_**(538 ページ)が更新され、Measure H07 が追加されました。
- **DEVICE^CLIM^DESC**(550 ページ)が更新され、GMS は CLIM デバイス（ISC および DEVICE エンティティ）には使用されないことが記載されました。
- **FABRIC_DESC**(562 ページ)が追加されました。
- **CODE^SPACE^DESC**(576 ページ)が更新され、TNS/X が追加されました。
- CLIM と `config^name` の更新の説明を追加するためと、L シリーズのその他の更新を追加するために、**SVNET^DESC**(584 ページ)が更新されました。
- **エラーメッセージ**(597 ページ)に新しいエラーメッセージ 3127、3249、3250、および 3402 が追加され、エラーメッセージ 3231 の「原因」の説明が変更されました。
- **例**(435 ページ)が更新され、L シリーズの情報が追加されました。
- **SPLIT コマンド**(675 ページ)の構文が更新され、L シリーズのエンティティが追加されました。
- コマンドに ENDPOINT エンティティタイプが追加されました。**ENDPOINT**(270 ページ)および **ENDPOINT_DESC**(560 ページ)が追加されました。

523324-020 の新規情報と更新内容

- **DISC**(109 ページ)の **discname-path** と **DISC エンティティのエンティティ指定構文**(226 ページ)の **path** の説明が更新されました。
- **INFO MEASUREMENT**(71 ページ)コマンドの時間情報の説明が更新されました。

523324-019 の新規情報と更新内容

- **PROCESSH エンティティのエンティティ指定構文**(388 ページ)に、EDIT ファイルを使用した PROCESSH 測定のコード範囲の指定のサポートは、Measure H02 以降の PVU の TNS 実行可能ファイルでは使用できなくなったという注記が追加されました。
- **MEASFT のエラーメッセージと警告メッセージ**(683 ページ)に、次のエラーメッセージと、メッセージの説明、原因、結果、および回復処置が追加されました。MEASFT 2033 (WARN_READ_INVALID_DATA): An invalid record was found in Measure data file.
- 「グループ」ではなく「ユーザー」を示すようになったため、**PROCESS エンティティの DDL レコード（ZMS 形式）**(360 ページ)の creatorid フィールドの 2 番目のサブフィールドが訂正されました。
- **エラーコード**(627 ページ)に以下のエラーメッセージが追加されました。
    - 3412（ERRM^DFFULL）: 使用中のデータファイルが EOF に達しました。
    - 3413（ERRM^DFFAILED）: 現在のデータファイルへの書き込みが失敗しました。
    - 3414（ERRM^LOCKMSGCTRLBUF）: LOCKMEMORY プロシージャーへの呼び出しが失敗しました。

- 3415（ERRM^SQLSPACEOVERFLOW）: SQL 制御ブロック領域が使用できません。
- 3416（ERRM^INVALIDRECLEN）: 構成レコードの単語の長さが極端に長くなっています。

523324-018 の新規情報と更新内容

- **H シリーズ以降の PVU および RVU についての START MEASSUBSYS の使用上の注意**(151 ページ)に START MEASSUBSYS の使用上の注意が追加され、CPU ごとの CID の使用が構成されたしきい値を超えた場合は EMS 警告イベント 4063 を送信するように Measure サブシステムを構成できることが記載されました。
- より明確にするために**最大値カウンター**(172 ページ)が改訂されました。
- **応答時間カウンター**(173 ページ)および **Syslink カウンター**(174 ページ)の表示単位の値が訂正されました。
- ロック待機カウンターについて説明する新しい**ロック待機カウンター**(173 ページ)サブセクションが追加されました。また、（**DISCOPEN**(249 ページ)および **DISKFILE**(259 ページ)エンティティの）個々のすべての LOCKWAIT-TIME カウンターのカウンタータイプの説明が改訂され、これらの各カウンターのタイプがロック待機カウンタータイプであることが記載されました。
- （**LINE**(309 ページ)および **TERMINAL**(439 ページ)エンティティの）RESPONSE-TIME カウンターのカウンタータイプの説明が改訂され、これらの各カウンターのタイプが応答時間カウンタータイプであることが記載されました。

古い項目については**新規情報と更新内容の古いエントリー**(714 ページ)を参照してください。

表記規則

一般的な構文の表記

このリストは、このマニュアルにおける構文表現の表記法をまとめたものです。

大文字

大文字は、キーワードと予約語を示します。これらの項目は表示されているとおりに正確に入力してください。大カッコで囲まれていない項目は必須です。次に例を示します。

MAXATTACH

斜体文字

斜体文字は、フォントとは関係なく、ユーザーが指定した変数項目を示します。大カッコで囲まれていない項目は必須です。次に例を示します。

file-name

コンピュータータイプ

コンピュータータイプの文字は、

- C およびオープンシステムサービス（OSS）のキーワード、コマンド、予約語を示します。これらの項目は表示されているとおりに正確に入力してください。大カッコで囲まれていない項目は必須です。次に例を示します。

  ```
  cextdecs.h
  ```

  ヘッダーファイルを使用します。
- コンピューターが表示するテキスト文字列のフォントです。次に例を示します。

```
Last Logon: 14 May 2006, 08:02:23
```

- コンピューターコードのリスト。次に例を示します

```
if (listen(sock, 1) < 0)
{
perror("Listen Error");
exit(-1);
}
```

太字

例の太字のテキストは、ユーザーが端末から入力したものであることを表します。次に例を示します。

```
ENTER RUN CODE

?123
CODE RECEIVED:      123.00
```

ユーザーは、入力後、Return キーを押す必要があります。

[ ] 大カッコ

大カッコはオプションの構文項目を囲みます。次に例を示します。

```
TERM [\system-name.]$terminal-name

INT[ERRUPTS]
```

大カッコで囲まれた項目のグループは、1 つの項目を選択する（または何も選択しない）リストです。リスト内の項目については、各項目を中カッコで囲んで縦に並べるか、すべての項目をまとめて中カッコで囲んで各項目を縦線で区切って横に並べることができます。次に例を示します。

```
FC [ num  ]
   [ -num]
   [ text]

K [ X | D ] address
```

{ } 中カッコ

中カッコで囲まれた項目のグループは、1 つの項目を選択する必要のあるリストです。リスト内の項目については、各項目を中カッコで囲んで縦に並べるか、すべての項目をまとめて中カッコで囲んで各項目を縦線で区切って横に並べることができます。次に例を示します。

```
LISTOPENS PROCESS { $appl-mgr-name }
                  { $process-name  }

ALLOWSU { ON | OFF }
```

| 縦線

縦線は、大カッコまたは中カッコで囲まれた水平リスト内の項目を区切ります。次に例を示します。

```
INSPECT { OFF | ON | SAVEABEND }
```

… 省略記号

一対の大カッコまたは中カッコの直後の省略記号は、カッコで囲まれた一連の構文項目を任意の回数反復できることを示します。次に例を示します。

```
M address [ , new-value ]…

 - ] {0|1|2|3|4|5|6|7|8|9}…
```

1 つの構文項目の直後の省略記号は、その構文項目を任意の回数反復できることを示します。次に例を示します。

```
"s-char…"
```

区切り記号

カッコ、カンマ、セミコロン、およびこれまでに説明したもの以外の記号は、表記どおりに入力する必要があります。次に例を示します。

```
error := NEXTFILENAME ( file-name ) ;

LISTOPENS SU $process-name.#su-name
```

大カッコや中カッコなどの記号を囲む引用符は、その記号を表記どおりに入力しなければならないことを示します。次に例を示します。

```
"[" repetition-constant-list "]"
```

項目間のスペース

項目のいずれかがカッコやカンマなどの区切り記号である場合を除き、項目間に示されているスペースは必須です。次に例を示します。

```
CALL STEPMOM ( process-id ) ;
```

2 つの項目間にスペースがない場合、スペースを入れることはできません。この例では、ピリオドと他の項目との間にスペースを入れることができません。

```
$process-name.#su-name
```

行間のスペース

コマンドの構文が長すぎて 1 行に収まらない場合、後続の各行は 3 つのスペースでインデントし、空白行で前の行から切り離します。この空白行は、後続の行内の項目と、縦に並んだ選択リスト内の項目とを区別します。次に例を示します。

```
ALTER [ / OUT file-spec / ] LINE

   [ , attribute-spec ]…
```

!i および!o

プロシージャーコールで、!i の表記は入力パラメーター（呼び出されたプロシージャーにデータを渡すパラメーター）に続き、!o の表記は出力パラメーター（呼び出し側のプログラムにデータを返すパラメーター）に続きます。次に例を示します。

```
CALL CHECKRESIZESEGMENT (  segment-id                    !i
                         , error        ) ;              !o
```

!i,o

プロシージャーコールで、!i,o の表記は入力/出力パラメーター（呼び出されたプロシージャーにデータを渡し、呼び出し側のプログラムにデータを返すパラメーター）に続きます。次に例を示します。

```
error := COMPRESSEDIT ( filenum ) ;                    !i,o
```

!i:i

プロシージャーコールで、!i:i の表記は、バイト単位で文字列の長さを指定する、対応したパラメーターを持つ入力文字列パラメーターに続きます。次に例を示します。

```
error := FILENAME_COMPARE_ (  filename1:length         !i:i
                            , filename2:length ) ;     !i:i
```

!o:i

プロシージャーコールで、!o:i の表記は、バイト単位で出力バッファーの最大長を指定する、対応した入力パラメーターを持つ入力出力バッファーパラメーターに続きます。次に例を示します。

```
error := FILE_GETINFO_ (  filenum                      !i
                        , [ filename:maxlen ] ) ;      !o:i
```

メッセージの表記

このリストは、本書で示されているメッセージの表記規則をまとめたものです。

太字

例の太字のテキストは、ユーザーが端末から入力したものであることを表します。次に例を示します。

```
ENTER RUN CODE

?123
CODE RECEIVED:      123.00
```

ユーザーは、入力後、Return キーを押す必要があります。

斜体以外

斜体以外の英字、数字、および句読記号は、表記どおりに正確に表示されるか返されるテキストを表します。次に例を示します。

```
Backup Up.
```

斜体

斜体のテキストは、値が表示されるか返される変数項目を表します。次に例を示します。

```
p-register

process-name
```

[ ] 大カッコ

大カッコは、常に表示されるとは限らない項目を囲みます。次に例を示します。

```
Event number = number [ Subject = first-subject-value ]
```

大カッコで囲まれた項目のグループは、表示される可能性のあるすべての項目のリストで、実際には、いずれかが表示される可能性もあれば、何も表示されない可能性もあります。リスト内の項目につい

ては、各項目を中カッコで囲んで縦に並べるか、すべての項目をまとめて中カッコで囲んで各項目を縦線で区切って横に並べることができます。次に例を示します。

```
proc-name trapped [ in SQL | in SQL file system ]
```

{ } 中カッコ

中カッコで囲まれた項目のグループは、表示される可能性のあるすべての項目のリストで、実際にはいずれかが表示されます。リスト内の項目については、各項目を中カッコで囲んで縦に並べるか、すべての項目をまとめて中カッコで囲んで各項目を縦線で区切って横に並べることができます。次に例を示します。

```
obj-type obj-name state changed to state, caused by
{ Object | Operator | Service }

process-name State changed from old-objstate to objstate
{ Operator Request. }
{ Unknown.          }
```

| 縦線

縦線は、大カッコまたは中カッコで囲まれた水平リスト内の項目を区切ります。次に例を示します。

```
Transfer status: { OK | Failed }
```

% パーセント記号

パーセント記号は、10 進表記以外の数字の前に置かれます。%表記は 8 進数の前に置かれます。%B 表記は 2 進数の前に置かれます。%H 表記は 16 進数の前に置かれます。次に例を示します。

```
%005400

%B101111

%H2F

P=% p-register E=% e-register
```

管理プログラミングインターフェイスの表記

このリストは、本書で枠で囲まれて記載されているプログラムコマンド、イベントメッセージ、およびエラーリストで使用される表記規則をまとめたものです。

大文字

大文字は定義ファイルの名前を示します。これらの名前は表示されているとおりに正確に入力してください。次に例を示します。

```
ZCOM-TKN-SUBJ-SERV
```

小文字

小文字の単語は、データ定義言語（DDL）のキーワードなど、表記の一部である単語です。次に例を示します。

```
token-type
```

!r

トークンまたはフィールド名の後の!r 表記は、トークンまたはフィールドが必須であることを示します。次に例を示します。

```
ZCOM-TKN-OBJNAME      token-type ZSPI-TYP-STRING.          !r
```

!o

トークンまたはフィールド名の後の!o 表記は、トークンまたはフィールドはオプションであることを示します。次に例を示します。

```
ZSPI-TKN-MANAGER      token-type ZSPI-TYP-FNAME32.         !o
```

関連情報

Measure オペレーターメッセージの一覧については、*オペレーターメッセージマニュアル*を参照してください。

発行履歴

部品番号	製品バージョン	発行日
867649-193	Measure H06 Measure J04 Measure L15	2017 年 3 月
867649-002	Measure L15	2016 年 6 月
867649-001	Measure H06 Measure J04	2016 年 5 月
523324-022R	なし	2015 年 11 月
523324-022	Measure H06 Measure J04 Measure L15	2015 年 8 月
523324-021	Measure H06 Measure J04 Measure L15	2014 年 2 月
523324-020	Measure H06 Measure J04	2013 年 11 月
523324-019	Measure H06 Measure J04	2013 年 8 月

表は続く

部品番号	製品バージョン	発行日
523324-018	Measure H06 Measure J04	2013 年 2 月
523324-017	Measure H06 Measure J04	2012 年 8 月
523324-016	Measure H06 Measure J04	2012 年 2 月
523324-015	Measure H06 Measure J04 [1]	2011 年 8 月

[1] Measure H06/J04 PVU は、当初は H06.22/J06.11 RVU でリリースされました。マニュアルの変更はありませんでした。

Measure の概要

Measure パフォーマンス監視は、システムリソースに関するパフォーマンス統計を収集するツールです。システムコンポーネント、ネットワークコンポーネント、自身のビジネスアプリケーションからデータを収集できます。このデータを使用して、システムのバランスおよび調整、ボトルネックの検出、ワークロードのバランス、およびキャパシティプランニングを行えます。複数のユーザーが同時に Measure セッションを実行して、各ユーザーが個別に測定を構成および取得できます。

動作の概要

この図は、Measure パフォーマンスモニターの主要コンポーネントである、Measure サブシステム、ユーザーインターフェイス、およびサブシステムインストゥルメンテーションを示しています。

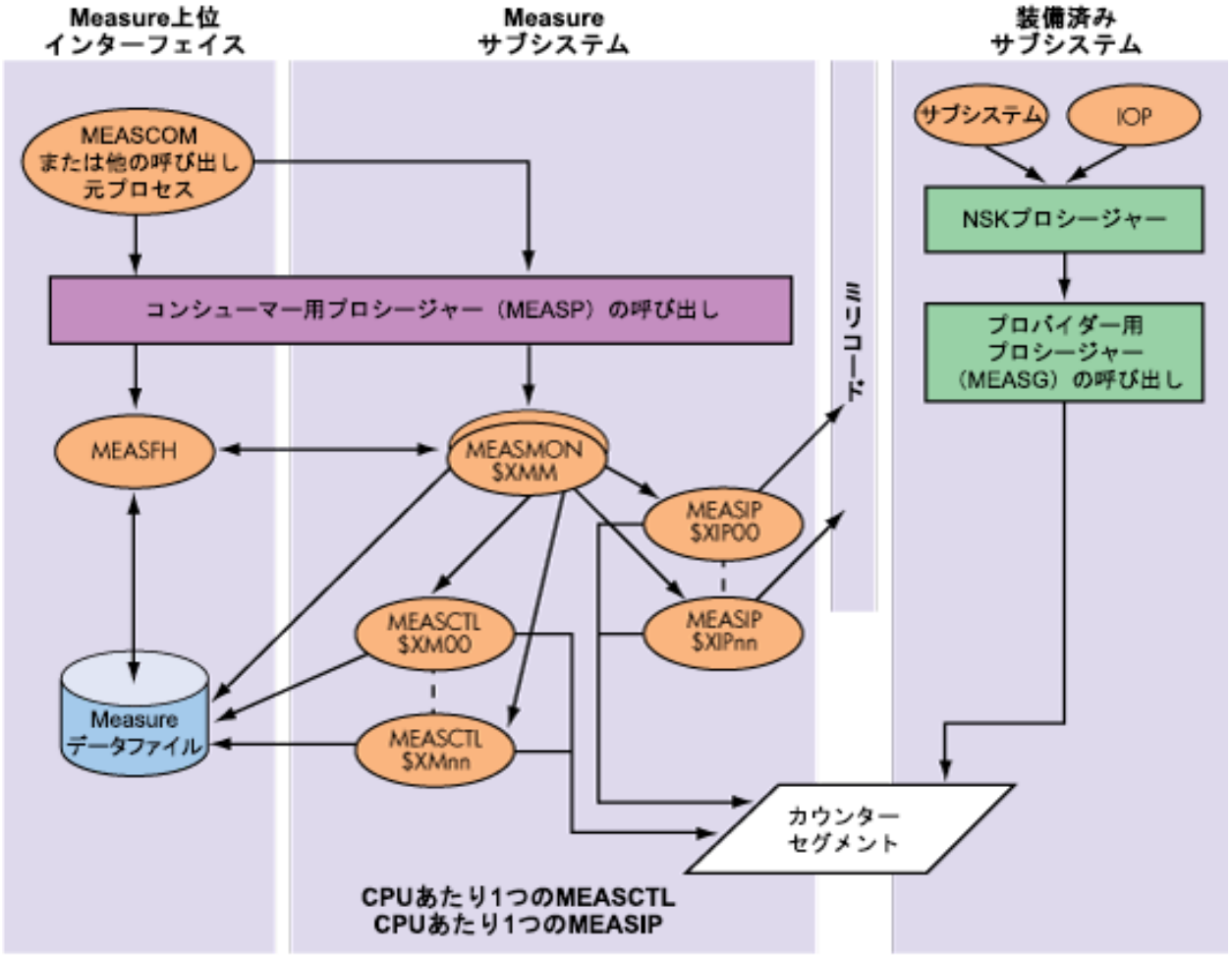


図 1: Measure パフォーマンスモニターの動作の概要

注記:

NSMA システムの場合、この図は、IPU ごとに 1 つの MEASIP プロセスを示します。MEASIP プロセス名の形式は「$XP *nna*」で、*nn* は CPU 番号（00～15）であり、*a* は IPU 番号（IPU 0 の場合は「A」から始まる）を示す英字です。

Measure サブシステム

測定を行うには、Measure サブシステムが実行されている必要があります。サブシステムを起動するには、スーパーグループユーザー（ユーザーグループ 255 のメンバー）である必要があります。このサブシステムのパフォーマンスへの影響は最小限に抑えられています。起動してから、実行を継続することができます。一部のインストールでは、オペレーターがシステムロード操作の一環としてサブシステムを起動します。

ユーザーインターフェイス

Measure パフォーマンスモニターには 2 つのインターフェイスがあります。

- コマンドインターフェイスは、端末で入力するかファイルから入力できるコマンドのセットです。各コマンドはプロシージャーを呼び出し、各プロシージャーは、測定の設定、測定の開始、収集したデータの表示などの機能を実行します。コマンドインターフェイスの説明については、**MEASCOM コマンド**(36 ページ)および**エンティティおよびカウンター**(167 ページ)を参照してください。
- プログラムインターフェイスは、呼び出し可能なプロシージャーのセットです。これらのプロシージャーをアプリケーションから呼び出すことで、Measure のすべての機能にアクセスできます。たとえば、測定を設定し、測定を開始してから停止して、分析のためにファイルにデータを書き込むアプリケーションを作成できます。プログラムインターフェイスの説明については、**Measure 呼び出し可能プロシージャー**(458 ページ)を参照してください。

コマンドインターフェイスを示すサンプルの測定は、このセクションの後半で示しています。ただし、最初にエンティティとカウンターという 2 つの基本的な用語を理解する必要があります。

エンティティおよびカウンター

エンティティは、測定対象となるシステムリソースです。Measure パフォーマンスモニターは、**エンティティおよびカウンター**にあるエンティティのタイプを認識します。

本書では、エンティティのクラス（プロセスやディスクなど）はエンティティタイプと呼ばれます。クラス内の個々の特定のエンティティの識別子はエンティティ指定と呼ばれます。したがって、FILE エンティティタイプ、\NY.$SALES.Q1.JAN はエンティティ指定です。

各エンティティタイプは、カウンターの定義済みセットに関連付けられています。カウンターは、特定のタイプのパフォーマンスデータを収集するプログラム構造です。たとえば CPU エンティティタイプには、測定された各 CPU 内で発生するスワップの数をカウントする SWAPS カウンターがあります。

サポートされているエンティティタイプごとに関連付けられているシステムソフトウェアには、Measure プロシージャーへの呼び出しが組み込まれています。測定可能なイベントが発生すると、制御するソフトウェアは、適切なカウンターをインクリメントするプロシージャーを呼び出します。

独自のカウンターを定義し、それらのカウンターをインクリメントするプログラム内の Measure プロシージャーコールを使用して、カスタムインストゥルメンテーションを測定に追加できます。たとえば、プログラムが特定のループを実行するたびにインクリメントされるカウンターを作成できます。カスタム測定を実行するプロセスは、エンティティタイプ USERDEF によってユーザー定義であることが識別されます。

測定の 3 つのステップ

測定の 3 つのステップは、次のとおりです。

手順

1. 測定を設定する
2. 測定を実施する
3. 測定を検査する

次の 3 つのサブセクションで、コマンドインターフェイスを使用した測定セッションの例について説明します。この例では、入力するコマンドは太字で示されています。Measure の画面表示は標準のフォントで示されています。

コマンドインターフェイスではなくプログラムインターフェイスを使用する測定セッションも基本的な操作は同じですが、操作はプログラム内のプロシージャーコールによって実行されます。

注記:

本書に示す MEASCOM リストは単なる例であり、予告なく変更されることがあります。出力の解析には使用できません。解析の場合は、SET REPORT FORMAT STRUCTURED コマンドを使用してください。

手順 1. 測定を設定する

測定の設定では、測定するエンティティを決定します。

サンプルの測定の最初の 3 行は、次のとおりです。

```
21> MEASCOM
1+ ADD CPU 5
2+ ADD CPU 6
```

1 行目では、MEASCOM（Measure コマンドインターフェイス）を起動します。プラス記号（+）は、MEASCOM コマンドプロンプトです。次の 2 行は、測定項目を設定しています。ここでは、CPU 5 および 6 で測定を行うよう Measure に指示しています。これらのコマンドでは、CPU がエンティティタイプ、5 および 6 がエンティティ指定です。

手順 2. 測定を実施する

サンプルの測定を開始するには、次のコマンドを使用します。

```
3+ START MEASUREMENT MYDATA
```

MYDATA というファイル名は出力ファイルです。このファイルが Measure パフォーマンスモニターの測定データの保存先となります。測定を開始するたびに、新規または既存の測定ファイルを指定する必要があります。

この測定では、多くのデフォルト値を使用します。測定の開始時刻、測定期間、および測定間隔などのオプションを指定できます。測定間隔を指定すると、Measure は測定の開始時と終了時だけでなく、指定された間隔でデータファイルにカウンターを書き込みます。たとえば、カウンター値を 30 分間隔で記録することもできます。これにより、測定を調べる場合に合計だけでなく連続する期間の結果も確認できます。

手順 3. 測定を検査する

サンプルの測定を停止して測定の結果を表示するには、次のコマンドを使用します。

```
4+ STOP MEASUREMENT MYDATA
5+ LIST CPU *

Cpu  5 Cyclone      Initial Lock Pgs 2048      Mem Pages  8192
Memory MB 16        PCBs    483                Page Size  2048
Local System \TSII From 17 Jul 1993, 15:06:38 For 30 Minutes

Cpu-Busy-Time        34.83 % Swaps                0.34
Cpu-Qtime             0.39 # Cpu-Qlen-Max          58 #
Mem-Qtime             0.02 # Mem-Qlen-Max          13 #
Dispatches           83.27   Intr-Busy-Time       1.45 %
Process-Ovhd          0.02 % Send-Busy-Time       0.61 %
Disc-IOs              9.56   Cache-Hits          17.42
```

```
Transactions                  Response-Time
Page-Requests                 Page-Scans
Ending-Free-Mem       4092    Ending-UCME
Ending-UDS             500    Ending-SDS            1500
Ending-UCL             100    Ending-SCL            2000

++

Cpu  6 Cyclone      Initial Lock Pgs 2048      Mem Pages  8192
Memory MB 16        PCBs    483                Page Size  2048
Local System \TSII From 17 Jul 1993, 15:06:38 For 30 Minutes

Cpu-Busy-Time        16.40 % Swaps                 0.58
Cpu-Qtime             0.22 # Cpu-Qlen-Max           58 #
Mem-Qtime             0.03  Mem-Qlen-Max           13 #
Dispatches           35.14   Intr-Busy-Time        0.61 %
Process-Ovhd          0.05 % Send-Busy-Time        0.17 %
Disc-IOs              5.34   Cache-Hits           10.24
Transactions                  Response-Time
Page-Requests                 Page-Scans
Ending-Free-Mem       4092    Ending-UCME
Ending-UDS             500    Ending-SDS            1500
Ending-UCL             100    Ending-SCL            2000
```

LIST コマンドは、測定結果を画面に表示します。測定対象の CPU ごとにレポートが表示されます。カウンター値はパーセンテージまたは合計として表示されます。

その他の出力オプションも存在します。測定間隔を指定した場合は、最終値のほかに、各間隔で取得されたカウンター値を調べることができます。解釈されなかった値（パーセンテージに変換されていない値）を表示できます。また、詳細レポートと簡略レポートのどちらかを選択できます。

ほとんどのエンティティタイプでは、測定中にもカウンター値を表示できます。

測定が完了したら、1 つまたは複数のカウンター値をプロットとして表示できます。プロットを使用してグラフィカルビューを作成すれば、値をより簡単に分析できるようになります。プロットするカウンターを指定してプロットを作成するには、次のコマンドを使用します。

```
6+ ADD PLOT CPU-BUSY-TIME
7+ LIST PLOT

  0:::::+:::20.0:::::+:::40.0:::::+:::60.0:::::+ ...
A ************************|
B ************|
  0:::::+:::20.0:::::+:::40.0:::::+:::60.0:::::+ ...

      Min value = 16.036              Max value = 34.827

      A: CPU-BUSY-TIME                Cpu 5
      B: CPU-BUSY-TIME                Cpu 6
```

結果として生成されるプロットには、測定対象の 2 つ CPU の CPU-BUSY-TIME カウンター値が表示されます。このプロットは棒グラフの形式で、各カウンター値は横 1 行のアスタリスクで表されます。2 軸形式を使用して、1 つの軸がカウンター値を表し、もう 1 つの軸が時間間隔を表すようにすることもできます。どちらのタイプのプロットでも、プロットの向き（横または縦）、値の目盛り、および時間枠を変更できます。

測定の開始と停止の際の考慮事項

測定中のシステムリソースの消費は全般的にあまり顕著でなくても、測定が開始または停止したときや間隔レコードが書き込まれるときに、CPU およびディスクの活動が急増することが認識されることがあります。

この問題を防ぐには、以下の手順を使用して、測定が停止するか間隔レコードを書き込むと同時に別の測定が開始するような構成を避けてください。

1. MEASCOM セッションを開始し、STATUS MEASSUBSYS コマンド（**INFO コマンドを使用して、アクティブな測定が間隔データを書き込む時間を判別する**(33 ページ)のプロンプト 1）を入力して、システム上で実行中（アクティブ）の測定の一覧を表示します。

   この例で、アクティブな測定は MDFGUEST、DOFF3、および MCP です。
2. 測定を追加し（例 1 のプロンプト 2、3、および 4）、INFO *コマンドを入力して（例 1 のプロンプト 5）、間隔が構成されている測定を表示します。

   例 1 で、INFO * コマンドの出力は、MDFGUEST と MCPU の両方が 10 秒の間隔で構成されていること、およびそれらの開始時刻（最初のサンプリング時刻）が MDFGUEST では 14:00:57、MCPU では 11:55:35 であることを示しています。それ以降の各サンプリング時刻は開始時刻からの時間間隔の倍数であるため、MDFGUEST は秒の 1 の位が 7 になる時刻、MCPU は秒の 1 の位が 5 になる時刻に、間隔データをそれぞれ書き出すことが予測できます。

   しかし、間隔が 17 秒などといった奇数で、開始時刻が数時間前である場合はどうなるでしょうか。

   このような場合、アクティブな測定で MEASCOM セッションを開始し（**LISTALL コマンドを使用して、アクティブな測定が間隔データを書き込む時間を判別する**(34 ページ)のプロンプト 1）、FROM オプションで最近の時刻を指定して LISTALL コマンドを入力して（例 2 のプロンプト 2）、最近のサンプリング時刻を表示します。
3. START MEASUREMENT コマンドで FROM オプションを使用して、新しい測定が、他のアクティブな測定とは異なる開始時刻かつ異なる間隔で開始するように構成します。

   たとえば例 1 で、間隔データが MDFGUEST および MCPU と同時に書き込まれないようにするには、秒の 1 の位が 1 になる時刻に新しい測定を開始し、INTERVAL の値を 10 に指定することが必要な場合もあります。

INFO コマンドを使用して、アクティブな測定が間隔データを書き込む時間を判別する

```
$DATA PERF 31> MEASCOM
MEASURE Performance Monitor - T9086J04 - (01FEB11) - \PROD1
(C) Copyright 1986-2010 Hewlett-Packard Development Company, L.P.
1+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements =  3
  $DATA.PERF.MDFGUEST
  $SYSTEM.MEASURE.DOFF3
  $GUEST.USER1.MCPU

Number of Active MEASCTL Processes =  4
  in CPU(s): 0, 1, 2, 3
Number of Active MEASIP Processes =   8
  in CPU(s): 0, 1, 2, 3

OSS OFF, SQL OFF, Skew copy time OFF, Lock CID table ON, CIDs = 64K/CPU
2+ ADD MDFGUEST
3+ ADD $SYSTEM.MEASURE.DOFF3
4+ ADD $GUEST.USER1.MCPU
5+ INFO *
Add measurement $DATA.PERF.MDFGUEST
Data collected from system \PROD1, MEASURE release version H06.
From 13 Apr 2011, 14:00:57,  Interval   10 Seconds
```

```
-- Add Disc $GUEST-M

Add measurement $SYSTEM.MEASURE.DOFF3
Data collected from system \PROD1, MEASURE release version H06.
From  7 Apr 2011,  8:52:59
Counter data records suppressed
-- Add Cpu *
-- Add Process *
-- Add File * (GUARDIAN)
-- Add Disc $*
-- Add Tmf *
-- Add Servernet * (*, *, 0, *, 0)

Add measurement $GUEST.USER1.MCPU -- Current Data File --
Data collected from system \PROD1, MEASURE release version H06.
From 13 Apr 2011, 11:55:35,  Interval   10 Seconds
-- Add Cpu *

7+ EXIT
```

LISTALL コマンドを使用して、アクティブな測定が間隔データを書き込む時間を判別する

```
$DATA PERF 32> TIME
April 13, 2011 14:23:45
$DATAS PERF 33> meascom
MEASURE Performance Monitor - T9086J04 - (01FEB11) - \PROD
(C) Copyright 1986-2010 Hewlett-Packard Development Company, L.P.
1+ ADD $GUEST.USER1.MCPU
2+ LISTALL CPU 1,FROM 14:24,FORMAT BRIEF
Cpu  1 NSE-M          Init Lock Pgs     19          Mem Pages  1048576
Memory MB 16384       PCBs           12100          Pg Size    16384  Bytes
IPUs    2
Format Version:  H06  Data Version:  H06  Subsystem Version:  2
Local System \PROD1  From  13 Apr 2011, 14:23:55   For    10 Seconds
------------------------------------------------------------------------------
...
Cpu  1 NSE-M          Init Lock Pgs     19          Mem Pages  1048576
Memory MB 16384       PCBs           12100          Pg Size    16384  Bytes
IPUs    2
Format Version:  H06  Data Version:  H06  Subsystem Version:  2
Local System \PROD1  From  13 Apr 2011, 14:24:05   For    10 Seconds
------------------------------------------------------------------------------
...
Cpu  1 NSE-M          Init Lock Pgs     19          Mem Pages  1048576
Memory MB 16384       PCBs           12100          Pg Size    16384  Bytes
IPUs    2
Format Version:  H06  Data Version:  H06  Subsystem Version:  2
Local System \PROD1  From  13 Apr 2011, 14:24:15   For    10 Seconds
------------------------------------------------------------------------------
...
Cpu  1 NSE-M          Init Lock Pgs     19          Mem Pages  1048576
Memory MB 16384       PCBs           12100          Pg Size    16384  Bytes
IPUs    2
Format Version:  H06  Data Version:  H06  Subsystem Version:  2
Local System \PROD1  From  13 Apr 2011, 14:24:25   For    10 Seconds
------------------------------------------------------------------------------
...
```

```
Cpu  1 NSE-M          Init Lock Pgs     19          Mem Pages  1048576
Memory MB 16384       PCBs           12100          Pg Size    16384  Bytes
IPUs    2
Format Version:  H06  Data Version:  H06  Subsystem Version:  2
Local System \PROD1  From  13 Apr 2011, 14:24:35   For    10 Seconds
-------------------------------------------------------------------------------
...
3+ EXIT
```

MEASCOM コマンド

この章では、Measure コマンドインターフェイス（MEASCOM）の使用方法と、MEASCOM の各コマンドについて説明します。詳しくは、**MEASCOM の使用に関する一般的な情報**(39 ページ)を参照してください。

MEASCOM コマンドの概要

表 1: MEASCOM コマンド

コマンド	機能
サブシステム制御コマンド	
START MEASSUBSYS(150 ページ)	Measure サブシステムを起動します。
STATUS MEASSUBSYS(157 ページ)	Measure サブシステムのステータスを表示します。
STOP MEASSUBSYS(160 ページ)	Measure サブシステムを停止します。
測定コマンド	
ADD entity-type(43 ページ)	測定するエンティティを選択します。
ADD COUNTER(46 ページ)	ユーザー定義カウンターを選択します。
DELETE entity-type(56 ページ)	選択された測定対象からエンティティを削除します。
DELETE COUNTER(58 ページ)	ユーザー定義カウンターを削除します。
INFO entity-type(68 ページ)	測定対象として選択したエンティティを表示します。
INFO COUNTER(70 ページ)	ユーザー定義カウンターに関する情報を表示します。
SQLCATALOG(149 ページ)	部分修飾の ANSI SQL 名を展開するためのデフォルトのカタログを指定します。
SQLSCHEMA(149 ページ)	部分修飾の ANSI SQL 名を展開するためのデフォルトのスキーマを指定します。
START MEASUREMENT(153 ページ)	測定を開始します。
STATUS MEASUREMENT(159 ページ)	測定に関する情報を表示します。

表は続く

コマンド	機能
STOP MEASUREMENT(161 ページ)	測定を停止します。
表示コマンド	
ADD MEASUREMENT(48 ページ)	調査する測定データファイルを選択します。
DELETE MEASUREMENT(59 ページ)	選択した検査対象から測定データファイルを削除します。
INFO MEASUREMENT(71 ページ)	測定データファイルに関する情報を表示します。
LIST entity-type(75 ページ)	選択したエンティティに関するレポートを表示します。
LIST EXTNAMES(94 ページ)	Measure の OSS PATHID の値を OSS パス名にマッピングするか、Guardian ファイル名または MID を ANSI SQL 名および Guardian ファイル名にマッピングするために、レコードのキーシーケンスファイルを作成するか、キーシーケンスファイルに追記します。
LIST OSSNAMES(95 ページ)	Measure の OSS PATHID の値を OSS パス名にマップするか、Guardian ファイル名または Measure の MID の値を同等の OSS ファイルパス名にマップするために、レコードのキーシーケンスファイルを作成するか、キーシーケンスファイルに追記します。
LISTACTIVE entity-type(104 ページ)	アクティブなエンティティに関するレポートを表示します。
LISTALL entity-type(115 ページ)	選択したエンティティに関するレポートを間隔レコードごとに表示します。
LISTENAME(123 ページ)	Guardian ファイル名と作成バージョンシリアル番号（CRVSN）を対応する外部名（ANSI SQL 名または OSS パス名）に変換します。
LISTGNAME(123 ページ)	OSS ファイルパス名を対応する Guardian ファイル名と作成バージョンシリアル番号（CRVSN）に変換します。
LISTPNAME(125 ページ)	Guardian ファイル名と作成バージョンシリアル番号（CRVSN）を同等の OSS ファイルパス名に変換します。
OSSPATH(127 ページ)	OSS ファイルパス名の展開に使用するデフォルトのディレクトリを宣言します。
PAGESIZE(128 ページ)	二重プロンプト（++）の前に表示される出力行の数を設定します。
RESET REPORT(130 ページ)	レポート形式のオプションをデフォルト値にリセットします。
SET REPORT(137 ページ)	レポート形式のオプションを設定します。
SHOW REPORT(148 ページ)	現在のレポート形式のオプションを表示します。

表は続く

コマンド	機能
プロットコマンド	
ADD PLOT(51 ページ)	プロットするカウンターを選択します。
DELETE PLOT(60 ページ)	選択したプロット対象からカウンターを削除します。
INFO PLOT(74 ページ)	プロット対象として選択されているカウンターに関する情報を表示します。
LIST PLOT(97 ページ)	選択したカウンターのプロットを表示します。
RESET PLOT(129 ページ)	プロット形式のオプションをデフォルト値にリセットします。
SET PLOT(133 ページ)	プロット形式のオプションを設定します。
SHOW PLOT(147 ページ)	現在のプロット形式のオプションを表示します。
基本的なコマンド	
ASSUME(54 ページ)	コマンドオブジェクトを選択します。
COMMENTS(55 ページ)	コメントメッセージの表示または非表示を選択します。
ENV(61 ページ)	環境パラメーターを表示します。
EXIT(63 ページ)	MEASCOM セッションを終了します。
FC(64 ページ)	コマンドを変更して実行します。
HELP(65 ページ)	コマンド構文とエラーメッセージを表示します。
HISTORY(67 ページ)	前に入力したコマンドを表示します。
LOG(126 ページ)	コマンドと応答のログをファイルに記録します。
OBEY(127 ページ)	コマンド（OBEY）ファイルを実行します。
OUT(128 ページ)	コマンド応答をファイルにリダイレクトします。
RUN(132 ページ)	Measure を終了せずに別のプロセスを実行します。
SETPROMPT(145 ページ)	環境情報を MEASCOM プロンプトに表示します。
SWAPVOL(163 ページ)	スワップファイルのボリュームを指定します。
SYSTEM(163 ページ)	デフォルトのシステムを選択します。
TIME(163 ページ)	指定したシステムの現在の日付と時刻を表示します。
VOLUME(164 ページ)	デフォルトのボリュームを選択します。

表は続く

コマンド	機能
WARNINGS(164 ページ)	警告メッセージの表示または非表示を選択します。
!（感嘆記号）(165 ページ)	コマンドを再実行します。

MEASCOM の使用に関する一般的な情報

MEASCOM コマンドの構文規則

1 行に複数のコマンド

MEASCOM のコマンド行には複数のコマンドを含めることができます。複数のコマンドを区切るには、セミコロン（;）を使用します。

複数行にまたがるコマンド

コマンドを次の行に続けるには、行の末尾にアンパサンド（&）を付加します。MEASCOM では、次の行に継続プロンプト（これもアンパサンド）が表示されます。次の行に入力するテキストは、前の行のコマンドの一部分として扱われます。それぞれの行には 132 文字まで含めることができます。1 つのコマンドは、合計 1100 文字まで、複数行にわたって継続できます。

コマンド内のコメント

コマンドにコメントを埋め込むには、コメントの前後に、ブランクを 1 つ、ハイフンを 2 つ、そしてブランクを 1 つ置きます（ -- ）。（ブランクは推奨されますが必須ではありません。ユーザー定義の名前でハイフンが有効であるため、ブランクによって問題が防止されます。）行の終わりでは、それ以前のすべてのコメントが終わりになります。

コマンドプロンプトと表示

MEASCOM では、新しいコマンドを受け入れることができるときにプラス記号（+）プロンプトが表示され、前の行からのコマンドを続行することができるときにアンパサンド（&）が表示されます。プロンプトにより、入力する行と MEASCOM が生成する行（プロンプトのない行）とが区別されます。

LIST コマンドは、複数のレポートを連続して生成し、レポートを確認するために各レポートの間で一時停止します。二重プロンプト（+ +）は、別のレポートが続くことを示します。次のレポートを表示するには、リターンキーを押します。残りのレポートをすべてスキップして入力プロンプトに戻るには、Ctrl-Y を押します（BREAK のキーシーケンスで、一部のキーボードでは BREAK キーも利用できます）。

一部の Measure オンラインヘルプの表示では、さらに多くの情報があることを示すために二重プロンプト（+ +）が使用されます。この場合も、リターンキーを押すと次の画面が表示され、Ctrl-Y を押すとコマンド入力に戻ります。

Measure G09 以降の PVU では、画面上の Measure エンティティレポートのスクロールを一部制御できます。**PAGESIZE**(128 ページ)によって、別のプロンプトが出される前に表示する行の数を宣言することができます。OSS ファイルのパス名が表示に存在すると、レポート形式の長さが変化します。情報を読み取る前に情報がスクロールして表示されなくなることを防ぐために、PAGESIZE を使用して出力の長さを直接制御します。

ディスクファイル名

ディスクファイルを指定するには、以下のフォームを使用できます。

filename
subvolume.filename

$volume.subvolume.filename
$volume.#filenum

ワイルドカードとしてアスタリスクを使用すると、ファイル名の要素を置き換えることができます。次に例を示します。

- $SR8.QUAL.*は、$SR8 ボリュームのサブボリューム QUAL 内のすべてのファイルを指します。
- $*.QUAL.STATUS は、いずれかのボリュームのサブボリューム QUAL 内の STATUS という名前のファイルを指します。
- $SR8.*.*は、$SR8 ボリューム上のすべてサブボリュームとすべてのファイルを指します。
- $*.*.*は、すべてのボリューム上のすべてサブボリュームのすべてのファイルを指します。

アスタリスクを使用して、ファイル名の要素の一部のみを表すことはできません。たとえば、$SR*をボリューム名として指定することは無効です。

MEASCOM ではファイルシステムのプロシージャーを使用してファイル名を解析します。ファイル名の代わりに論理 DEFINE およびデバイス名も使用できます。たとえば、名前が =_myfile で定義される FILE エンティティの測定を設定するには、次のようにします。

```
1+ ADD FILE =_myfile
```

LDEV 14 に対応する DISC ボリュームの測定を設定するには、次のようにします。

```
2+ ADD DISC $14
```

ファイルシステムが論理ファイル名を物理ファイル名に変換できないか、または指定されたエンティティタイプに対応した物理名でない場合は、エラーが返されます。

注記:

論理デバイス番号は、特定のシステムに固有です。データファイルが 1 つのシステムから別のシステムに移動する場合、そのファイルの任意の既存の論理デバイス番号が無効になります。

G シリーズ以降の RVU を実行しているシステムでは、論理デバイス番号はシステムの起動時に動的に決定されます。したがって、特定のデバイス用の論理番号は、他のデバイスをシステムに追加したりシステムから削除したときに変更される可能性があります。この理由から、G シリーズ以降の RVU を実行しているシステムでは、コマンド（OBEY）ファイルまたはその他の非対話式ファイルで論理デバイス番号を使用しないでください。

コマンドの略語

MEASCOM コマンドを+プロンプトで対話式に入力するとき、コマンド、オブジェクト、属性、およびカウンター名のキーワードを省略することができます。たとえば、STATUS MEASUREMENT と入力する代わりに、STAT MEASU と入力することができます。

- MEASCOM は略語を単語ごとに突き合わせ、完全一致を優先させます。たとえば、LIST は LISTALL の省略形ではなく、常に LIST コマンドとして認識されます。
- MEASCOM は DISK をキーワード DISC の代替スペルとして認識するため、DISK を DISKFILE の略語として使用することはできません。
- コマンド、オブジェクト、および属性の場合、MEASCOM はそれぞれの略語を他のすべてのコマンド、オブジェクト、および属性と比較します。使用する各略語は一意でなければなりません。たとえば、RE は REPORT オブジェクトにも一致するため、RESET コマンドの略語として使用できません。
- カウンター名の場合、MEASCOM は指定されたエンティティタイプの他のカウンター名に対してのみ略語を比較します。たとえば、DISC エンティティには REQUEST-QTIME および REQUEST-QLEN-MAX という名前のカウンターがあります。これらの名前を REQUEST-QT および REQUEST-QL として省略できます。ハイフンはスペースではなく名前の一部として扱われます。REQUEST-QTIME を REQ-QT として省略することはできません。
- MEASCOM に新しいキーワードが追加されるたび、使用する略語を変更することが必要になる場合があります。
- 非対話型コマンド入力で略語を使用することはできません（つまり、MEASCOM コマンドを TACL プロンプトから実行する場合）。

- 非対話型コマンドの入力時に使用されるコマンド（OBEY）ファイルまたは入力ファイルで略語を使用することはできません。略語は RVU ごとに異なる場合があるため、この制限によってコマンドファイルが古くなって使用できなくなることを防ぎます。
- HELP コマンドでトピックを省略することはできません。たとえば、HELP ADD TER ではなく HELP ADD TERMINAL と入力する必要があります。

MEASCOM セッションの実行

コマンドインターフェイスまたはプログラムインターフェイスを介して測定を行うには、Measure サブシステムが実行中である必要があります。多くのインストールでは、サブシステムをシステムのロード操作の一環として起動してから、継続的に実行したままにしてあります。

Measure サブシステムの起動

サブシステムを起動できるのは、スーパーグループユーザー（255,*n*）のみです。

MEASCOM を使用してサブシステムを起動するには、TACL のプロンプトで次のコマンドを入力します。

```
> MEASCOM START MEASSUBSYS
```

サブシステムをプログラムで起動するには、MEASMONCONTROL プロシージャーを使用します。説明については、**MEASMONCONTROL**(499 ページ)を参照してください。

MEASCOM の起動と停止

MEASCOM は対話形式でも非対話形式でも使用できます。以下のそれぞれの使用例では、Measure サブシステムがすでに実行されていることを前提としています。

- MEASCOM を対話形式で使用するには、コマンドインタープリターのプロンプトでコマンド MEASCOM を入力します。システムが MEASCOM プロセスを起動し、プロセスがプラス記号（+）を使用して入力を求めるプロンプトを表示したら、コマンドを対話形式で入力します。セッションの最後に、EXIT と入力するか Ctrl-Y キーを押すと、オペレーティングシステムのコマンドインタープリターに戻ります。次に例を示します。

```
11> MEASCOM
MEASURE Performance Monitor - T9086H06 - (01AUG11) - \SYSTEM
(C) Copyright 1986-2010 Hewlett-Packard Development Company, L.P.
1+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements =  0

Number of Active MEASCTL Processes =  4
  in CPU(s): 0, 1, 2, 3
Number of Active MEASIP Processes =   4
  in CPU(s): 0, 1, 2, 3

OSS OFF, SQL OFF, Skew copy time OFF, Lock CID table ON, CIDs = 64K/CPU
2+ EXIT
```

- MEASCOM を非対話形式で使用するには、MEASCOM と入力し、続けて 1 つまたは複数の Measure のコマンドを入力します。MEASCOM プロセスが起動され、指定されたコマンドを実行してから、コマンドインタープリターに制御を返します。次に例を示します。

```
11> MEASCOM STATUS MEASSUBSYS
MEASURE Performance Monitor - T9086H06 - (01AUG11) - \SYSTEM
(C) Copyright 1986-2010 Hewlett-Packard Development Company, L.P.
Number of Active (or Configured) Measurements =  0

Number of Active MEASCTL Processes =  4
  in CPU(s): 0, 1, 2, 3
Number of Active MEASIP Processes =   4
```

```
  in CPU(s): 0, 1, 2, 3

OSS OFF, SQL OFF, Skew copy time OFF, Lock CID table ON, CIDs = 64K/CPU
```

キーワード MEASCOM の後のコマンドを入力する代わりに、Measure のコマンドが含まれている入力ファイルを指定できます。次に例を示します。

```
35> MEASCOM /IN $MEAS.CMDS.MCONFIG /
```

MCONFIG は、以下のコマンドが含まれているファイルです。

```
ADD CPU *
ADD DISC *
START MEASUREMENT $DATA.PERF.MAY04
```

MEASCOM コマンドの詳細な構文については、**MEASCOM**(42 ページ)を参照してください。

カスタムの起動ファイルの作成

コマンド（OBEY）ファイルに類似した MEASCOM コマンドのファイルを作成して、MEASCOM が呼び出されるたびに実行するようにすることができます。ファイルは MEASCOM コマンドのみを含む、MEASCSTM という名前の編集ファイルでなければなりません。

MEASCOM は現在のサブボリュームで MEASCSTM ファイルを検索します。ファイルが見つからない場合、MEASCOM はデフォルトサブボリュームを検索します。この機能により、さまざまな測定環境および分析環境について異なる MEASCSTM ファイルを設定することができます。

この例では、MEASCSTM ファイルを示します。

```
COMMENTS SUPPRESS 2000
WARNINGS SUPPRESS 3013
SETPROMPT VOLUME
SET REPORT RATE OFF
SET REPORT LOADID TEST
```

MEASCOM

MEASCOM コマンドは、Measure コマンドインターフェイス（MEASCOM）を起動することによって測定セッションを開始します。

MEASCOM の構文

```
MEASCOM [ / [ IN filename ] [ , OUT listfile ] / ]
    [ command [ ; command ] ... ]
```

filename

MEASCOM がコマンドを読み取る既存のディスクファイル、ディスク以外のデバイス、またはプロセスです。省略すると、*filename* は、デフォルトでコマンドインタープリター（通常は端末）の現在の入力ファイルに指定されます。

listfile

MEASCOM が出力を書き込むディスクファイル、ディスク以外のデバイス、またはプロセスです。省略すると、*listfile* は、デフォルトでコマンドインタープリター（通常は端末）の現在の出力ファイルに指定されます。

MEASCOM は、*listfile* に次のようにコマンド出力を書き込みます。

- *listfile* が既存のディスクファイルの場合、MEASCOM はファイルに出力を追加します。
- *listfile* がテープファイルの場合、MEASCOM は、そのファイルに書き込みを行った後で連続する 2 つのファイルマークを書き込んでから、そのファイルを閉じます。

- *listfile* がラインプリンターまたはプロセスの場合、MEASCOM は、ファイルを開いた後でページ排出を書き込んでから、ファイルに書き込みを行います。
- *listfile* が存在しない場合、MEASCOM が新しいディスクファイルとして作成します。

MEASCOM が端末に書き込む文字数は、1 行あたり 80 文字未満です。

command

Measure のコマンドです。省略すると、MEASCOM はコマンドを *filename* から読み取ります。コマンドを指定すると、MEASCOM は、それらを実行してから、オペレーティングシステムのコマンドインタープリターに制御を返します。

関連コマンド

コマンド	機能
EXIT	MEASCOM セッションを終了します

MEASCOM の使用上の注意

MEASFH からデータを取得しようとしているときに複数の呼び出しが必要な場合、MEASCOM は各呼び出しの前に BREAK がないか確認します。

MEASCOM の例

Measure コマンドインターフェイスを起動するには、Measure サブシステムのステータスを確認し、コマンドインターフェイスを停止します。

```
11> MEASCOM
MEASURE Performance Monitor - T9086H06 - (01AUG11) - \SYSTEM
(C) Copyright 1986-2010 Hewlett-Packard Development Company, L.P.
1+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements =  0

Number of Active MEASCTL Processes =  4
  in CPU(s): 0, 1, 2, 3
Number of Active MEASIP Processes =   4
  in CPU(s): 0, 1, 2, 3

OSS OFF, SQL OFF, Skew copy time OFF, Lock CID table ON, CIDs = 64K/CPU
2+ EXIT
```

ADD entity-type

ADD *entity-type* コマンドは、測定するエンティティタイプ（たとえば CPU）を指定します。

いくつかのエンティティタイプを同時に測定するには、測定を開始する前に各エンティティタイプの ADD *entity-type* コマンドを入力します。

ADD entity-type の構文

```
ADD [ entity-type ] entity-spec [ , entity-spec ] ...
```

entity-type

次のいずれかです。

```
CLUSTER     DISCOPEN  LINE    PROCESS    SYSTEM
CONTROLLER  DISKFILE  NETLINE PROCESSH   TERMINAL
CPU         ENDPOINT  OPDISK  SERVERNET  TMF
DEVICE      FABRIC    OSSCPU  SQLPROC    USERDEF
DISC        FILE      OSSNS   SQLSTMT
```

entity-type キーワードは、ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして指定された場合はオプションです。

entity-spec

測定対象のエンティティを識別します。エンティティ指定はエンティティタイプごとに固有です。構文については、**エンティティおよびカウンター**(167 ページ)にある指定したエンティティタイプの説明を参照してください。

関連コマンド

コマンド	機能
DELETE *entity-type*	エンティティを構成から削除します
INFO *entity-type*	構成内のエンティティを表示します
START MEASUREMENT	構成を定義した後に測定を開始します

すべての PVU および RVU での ADD entity-type の使用上の注意

エンティティのサブセットを測定に含めたり除外したりする方法の説明は、**DELETE entity-type の例**(57 ページ)にある DELETE *entity-type* コマンドを参照してください。

L シリーズの PVU および RVU での ADD entity-type の使用上の注意

L シリーズの PVU および RVU では、 ADD SERVERNET コマンドは、GMS、ポート、およびファイバー仕様をサポートしません。L シリーズの RVU の SERVERNET エントリーでは、CLIM に関連する IPC および RIPC ノードクラスのみが利用できます。

H シリーズ以降の PVU および RVU についての ADD entity-type の使用上の注意

- H06.23/J06.12 以降の RVU では、CLIM に接続されたディスク用の ADD DISC コマンドは、オプションの DISC 仕様パラメーターとしてパーティション番号をサポートします。パーティション番号は、パーティション分割されたディスク内のパーティションに割り当てられている番号です。**ADD entity-type の例**(45 ページ)も参照してください。
- Measure H03/J01 以降の PVU では、ADD コマンドは、CLIM を使用する SERVERNET、DEVICE、および DISC エンティティ指定を可能にします。構文と例については、**SERVERNET**(399 ページ)、**DEVICE**(214 ページ)、および **DISC**(226 ページ)エンティティを参照してください。
- Measure H01 以降の PVU では、ADD コマンドは、ANSI SQL 名を使用する DISCOPEN、DISKFILE、および FILE のエンティティ指定を可能にします。構文については、**DISCOPEN**(249 ページ)、**DISKFILE**(259 ページ)、および **FILE**(284 ページ)エンティティを参照してください。

ADD entity-type の例

- 名前が$DISC1 の CLIM 接続ディスク内でパーティションを測定する構成を定義するには、以下のようにします。

```
+ ADD DISC $DISC1 (3, X, C1002531, 308, 1)
```

- ANSI SQL 名を使用してエンティティを追加するには、以下のようにします。

```
+ ADD FILE 'TABLE CATALOG_12.SCHEMA_34.TABLE_56'
+ ADD DISCOPEN 'SCHEMA CATALOG_12.SCHEMA_34'
+ ADD DISKFILE 'CATALOG CATALOG_12'
```

> **注記:**
>
> 追加されたテーブルがドロップされて同じ測定中に後で再作成される場合、そのテーブルの最初のインスタンスのみが測定されます。

- G シリーズで使用されるシンボルファイル TSYSCLR の代わりにシステム DLL の INITDLL および MCPDLL を使用して H シリーズの測定を追加するには、以下のようにします。

```
+ ADD PROCESSH $SYSTEM.SYS01.TSYSDP2 (2, *) ($SYSTEM.SYS01.TSYSDP2)
+ ADD PROCESSH $SYSTEM.SYS01.TSYSDP2 (2, *) ($SYSTEM.SYS01.INITDLL)
+ ADD PROCESSH $SYSTEM.SYS01.TSYSDP2 (2, *) ($SYSTEM.SYS01.MCPDLL)

+ ADD PROCESSH $SYSTEM.SYS01.TSYSDP2 (11, *) ($SYSTEM.SYS01.TSYSDP2)
+ ADD PROCESSH $SYSTEM.SYS01.TSYSDP2 (11, *) ($SYSTEM.SYS01.INITDLL)
+ ADD PROCESSH $SYSTEM.SYS01.TSYSDP2 (11, *) ($SYSTEM.SYS01.MCPDLL)
+ ADD PROCESSH ALLTIME 1 ($SYSTEM.SYS00.INITDLL)
+ ADD PROCESSH ALLTIME 1 ($SYSTEM.SYS00.MCPDLL)
+ ADD PROCESSH ALLTIME 1 (SL $SYSTEM.SYS00.TSL)
+ ADD PROCESSH ALLTIME 8 ($SYSTEM.SYS00.INITDLL)
+ ADD PROCESSH ALLTIME 8 ($SYSTEM.SYS00.MCPDLL)
+ ADD PROCESSH ALLTIME 8 (SL $SYSTEM.SYS00.TSL)
```

- すべての CPU、すべてのディスク、およびすべてのプロセスを測定する構成を定義するには、以下のようにします。

```
+ ADD CPU *
+ ADD DISC *
+ ADD PROCESS *
```

- NWREG.ACCOUNTS ファイルがプロセス 6,234 によってオープンされるとき、そのファイル上で実行される論理（FILE）と物理（DISCOPEN）の両方の I/O 操作を測定するには、以下のようにします。

```
+ ADD FILE $*.NWREG.ACCOUNTS (6,234)
+ ADD DISCOPEN $*.NWREG.ACCOUNTS (6,234)
```

- 特定のオブジェクトファイルを実行中のすべてのプロセスを測定に追加し、プロシージャー（code-range）実行データを収集するには、以下のようにします。

```
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.TCPIP)
```

- 特定のコード範囲の測定を追加し、EDIT（コード 101）ファイルを介して分かりやすい名前を指定するには、範囲または範囲オフセットのいずれかを指定できます。この例では、コード範囲を定義するオフセットの使用を示しています。

```
OBJECT $a.b.prog1
CSS^6100^LEVEL3                  130206
FIRST^LOCATION                  +000000
PROCESS^EVENT                   +000401
PROCESS^EVENT^1                 +001401
PROCESS^EVENT^2                 +002401
PROCESS^EVENT^3                 +003401
```

```
PROCESS^EVENT^4                 +004401
PROCESS^EVENT^5                 +005401
L4^PROTOCOL                     +006577
CMI^PROTOCOL                    +006743
```

ADD COUNTER

ADD COUNTER コマンドは、ユーザー定義カウンターを指定のプロセスから現在の構成に追加します。

ユーザー定義カウンターを構成する前に、そのカウンターに関連付けられているプロセスを構成する必要があります。プロセスを構成するには、ADD USERDEF コマンド（ADD *entity-type* コマンドで、エンティティタイプが USERDEF）を使用します。

カウンタータイプの詳細は、**カウンター値の解釈**(168 ページ)を参照してください。

ADD COUNTER の構文

```
ADD [ COUNTER ] counter-name, PROCESS process-spec, counter-type [ , ARRAY
size ]
```

COUNTER

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして COUNTER を指定した場合はオプションです。

counter-name

カウンター名です。カウンター名には 1～16 個の英字、数字、ハイフン、および下線を含めることができます。最初の文字は、英字でなければなりません。

PROCESS *process-spec*

カウンターを保持するプロセスの名前です。このプロセス仕様は、ADD USERDEF コマンドのプロセス仕様と正確に一致しなければなりません。

プロセス仕様として\MAIN.$ACCTS.BILLING.*などのワイルドカードテンプレートを使用することができます。ただし、テンプレートによって示されるすべてのプロセスは、同じカウンターを変更する必要があります。

counter-type

作成するカウンターのタイプです。D シリーズおよび G シリーズの RVU では、*counter-type* は以下のいずれかである必要があります。

ACCUM	累積カウンター（長さ= 3 ワード）
FACCUM	64 ビットの累積カウンター（長さ= 5 ワード）
BUSY	ビジーカウンター（長さ= 6 ワード）
QUEUE	キューカウンター（長さ= 7 ワード）

H シリーズ以降の RVU では、*counter-type* は以下のいずれかである必要があります。

ACCUM	累積カウンター（長さ= 2 ワード）
FACCUM	64 ビットの累積カウンター（長さ= 4 ワード）
BUSY	ビジーカウンター（長さ= 4 ワード）

表は続く

QUEUE	キューカウンター（長さ= 8 ワード）
QBUSY	キュービジーカウンター（長さ= 8 ワード）
TCELLBUSY	ビジーカウンター、タイマーセルで維持（長さ= 4 ワード）
TCELLQUEUE	キューカウンター、タイマーセルで維持（長さ= 8 ワード）
TCELLQBUSY	キュービジーカウンター、タイマーセルで維持（長さ= 8 ワード）

ARRAY *size*

カウンター配列の長さ（1～127）です。配列インデックスは 0 で開始されるため、サイズが 1 の場合は 2 要素の配列が作成され、サイズが 2 の場合は 3 要素の配列が作成されるといったようになります。

注記:

H07、J05、L シリーズ以降の PVU のカウンター配列の長さは 1 から 511 です。

デフォルトでは、カウンター名は 1 つのカウンター値（配列長 0）に関連付けられています。また、カウンター名をカウンター値の配列に関連付けることができます。

関連コマンド

コマンド	機能
ADD USERDEF	1 つまたは複数のユーザー定義のカウンターを変更するプロセスを構成に追加します
DELETE COUNTER	ユーザー定義のカウンターを構成から削除します
INFO COUNTER	構成内のユーザー定義のカウンターを表示します

ADD COUNTER の使用上の注意

- ユーザー定義のカウンターをアプリケーションに配置するには、適切な時点で MEASCOUNTERBUMPINIT および MEASCOUNTERBUMP プロシージャーを呼び出すようにソースコードを変更します。これらのプロシージャーの説明については、**MEASCOUNTERBUMP**(473 ページ)および **MEASCOUNTERBUMPINIT**(474 ページ)を参照してください。

  ユーザー定義カウンターから情報を収集するには、以下のようにします。

  1. ADD USERDEF コマンドを使用して、カウンターを変更するプロセスを構成に追加します。
  2. ADD COUNTER コマンドを使用して、カウンターを構成に追加します。ADD USERDEF コマンドで指定されているように *entity-spec* を正確に指定することで、カウンターを変更する各プロセスを識別します。
- ユーザー定義のカウンターのレコードは、長さを 1000 ワードまでに指定できます。1000 ワードの制限を超えない限り、各プロセスは最大 85 個の名前付きユーザーカウンターを保持することができ、各カウンターは最大 128 個のカウンター値の配列とすることができます。配列内の各エンティティはカウンターとみなされ、1000 ワードの制限に対してカウントされます。長いカウンター配列がある場合、カウンターレコードがオーバーフローする可能性があります。

各 USERDEF エンティティは、合計配列要素が 256 個に制限されます。配列要素の数を計算するには、配列ごとの要素数（ARRAY *size*）に、定義される配列ごとに 1 を加算します。次に例を示します。

```
+ ADD COUNTER COUNTER1, PROCESS MEASTEST, ACCUM, ARRAY 20
+ ADD COUNTER COUNTER2, PROCESS MEASTEST, BUSY,  ARRAY 20
```

この例では、配列要素の合計数は 42（20 + 20 + 1 + 1）です。配列要素の合計数が 256 を超えると、Measure はエラー 3251（構成エラー）を返します。

- H07、J05、L シリーズ以降の PVU では、ユーザー定義カウンターのレコードは、長さを最大 8000 ワードとすることができます。8000 ワードの制限を超えない限り、各プロセスは最大 600 個の名前付きユーザーカウンターを保持することができ、各カウンターは最大 512 個のカウンター値の配列とすることができます。配列内の各エンティティはカウンターとみなされ、8000 ワードの制限に対してカウントされます。長いカウンター配列がある場合、カウンターレコードがオーバーフローする可能性があります。

  各 USERDEF エンティティは、合計配列要素が 1600 個に制限されます。配列要素の数を計算するには、配列ごとの要素数（ARRAY *size*）に、定義される配列ごとに 1 を加算します。次に例を示します。

  ```
  + ADD COUNTER COUNTER1, PROCESS MEASTEST, ACCUM, ARRAY 20
  + ADD COUNTER COUNTER2, PROCESS MEASTEST, BUSY,  ARRAY 20
  ```

  この例では、配列要素の合計数は 42（20 + 20 + 1 + 1）です。配列要素の合計数が 1600 を超えると、Measure はエラー 3251（構成エラー）を返します。
- 複数のプロセスがカウンターを保持できます。ただし、あるプロセスについて測定できるユーザー定義カウンターは 1 セットだけです。つまり、プロセス内のユーザー定義カウンターのセットを使用して測定を開始した場合は、最初の測定が停止するまで、その同じプロセス内で別のユーザー定義カウンターのセットを使用して別の測定を開始することはできません。
- H07、J07 および L シリーズでは、同じプロセス内で複数の USERDEF 測定を同時に行うことが許可されます。プロセスのセカンダリ USERDEF 測定を指定すると、特定のカウンターの INDEX 値は以前指定したものと同じでなければなりません。外部レコード形式は変更されません。

ADD COUNTER の例

この例では、現在の構成に 2 つのユーザー定義カウンターを追加します。これらは累積カウンターである TRANSACTIONS とビジーカウンターである WAITING です。カウンターは両方とも BILLING プロセスによって変更されます。

```
+ ADD USERDEF BILLING
+ ADD COUNTER TRANSACTIONS, PROCESS BILLING, ACCUM
+ ADD COUNTER WAITING, PROCESS BILLING, BUSY
```

ADD MEASUREMENT

ADD MEASUREMENT コマンドによって、測定データファイルに MEASCOM からアクセスできるようになるため、データファイルからレポートまたはプロットを生成することができます。LIST、LISTALL、または LISTACTIVE コマンドを使用してデータを表示する前に、ADD MEASUREMENT を発行する必要があります。

ADD MEASUREMENT では、ファイルのファイル処理プロセス（MEASFH）を作成することによってデータファイルがアクセス可能になります。MEASFH プロセスは、その測定データファイル用のすべての I/O を実行します。MEASFH は、ファイルの ADD MEASUREMENT を最初に発行したときに作成され、データファイルへのアクセスを削除するか、Measure セッションを終了するまでアクティブなままになります。

アクセスできるデータファイルは一度に 1 つだけですが、あるデータファイルへのアクセスを削除しなくても、別のファイルにアクセスすることができます。たとえば、FILE1 に対して ADD MEASUREMENT コマンドを発行し、FILE1 からのデータを表示して、FILE2 に対して ADD MEASUREMENT コマンドを発行することができます。FILE2 が現在アクセス可能なデータファイルになり、FILE1 へのアクセスは一

時停止されます（削除はされません）。この機能を使用して、関連付けられた MEASFH プロセスを停止したり起動したりせずにデータファイル間を移動することができます。

ADD MEASUREMENT の構文

```
ADD [ MEASUREMENT ] data-file [ , MEASFH measfh ]
```

MEASUREMENT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして MEASUREMENT を指定した場合はオプションです。

data-file

測定データを含むディスクファイル（ローカルまたはリモート）です。現在アクティブなデータファイルを指定することができます。

MEASFH *measfh*

ご使用のシステムのデフォルト以外の MEASFH プロセスを指定します。*measfh* はオブジェクトファイル名です。以下の場合は代替の MEASFH プロセスを指定してください。

- Measure サブシステムの以前の製品バージョンで作成された測定データファイルを処理する必要がある場合。MEASFH および測定データファイルはリリースに依存します。たとえば、D40 MEASFH は D40 データファイルと一緒に使用する必要があります。
- リモートシステム上のデータファイルを指定する場合。リモートシステム上の MEASFH を使用する必要はありませんが、ネットワークを介して I/O を行うよりも、リモートデータファイルへの I/O をリモート MEASFH に実行させる方がはるかに高速です。

MEASFH オブジェクトファイルの名前は通常、$SYSTEM.SYS*nn*.MEASFH です。システムのサブボリューム名を検出するには、STATUS *コマンドを使用します。SYS*nn* は OSIMAGE ファイルを含むサブボリュームです。

注記:

G シリーズの RVU を実行しているシステムで D シリーズ以前の RVU からのデータファイルを調べるには、適切な MEASFH とともに、D*nn* 製品バージョンの MEASCOM を使用する必要があります。さまざまなデータファイルとの互換性をできるだけ保つには、常に最新の D*nn* 製品バージョンの MEASCOM を使用してください。

関連コマンド

コマンド	機能
DELETE MEASUREMENT	データファイルを閉じて、その現在の MEASFH プロセスを削除します
INFO MEASUREMENT	アクセス可能なデータファイルと、レポートとプロットに使用するファイル（現在のデータファイル）を表示します

すべての PVU および RVU での ADD MEASUREMENT の使用上の注意

- MEASFH では、ファイルの索引および外部カウンターレコードを作成するために、データファイルのサイズの 3 倍のディスクスペースが必要です。このディスクスペースは、MEASCOM スワップボリュームと同じディスク上または SWAPVOL コマンドを使用してスワップファイルに指定されたボリューム上のいずれかになります。
- ユーザー ID はデータファイルに対する読み取りアクセス権が必要です。

- 同じセッション中に多くの測定にアクセスしようとしすぎた場合、SEGMENT_ALLOCATE_はエラー5（無効なセグメントサイズ）を返します。エラー番号とその定義の完全なリストについては、*Guardian プロシージャーコールリファレンスマニュアル*を参照してください。
- ADD MEASUREMENT コマンドは、ADD MEASUREMENT が所定のデータファイルに初めて発行されたときに MEASFH 句を処理します。セッション中に同じデータファイルで別の MEASFH プロセスを使用するには、DELETE MEASUREMENT コマンドを使用して現在の MEASFH プロセスを削除してから、ADD MEASUREMENT コマンドを再発行して新しい MEASFH プロセスを指定してください。L シリーズの RVU 用の MEASFH06 については、**L シリーズの PVU および RVU での ADD MEASUREMENT の使用上の注意**(50 ページ)を参照してください。

L シリーズの PVU および RVU での ADD MEASUREMENT の使用上の注意

Measure は、L シリーズの RVU の H06 以前のデータファイルバージョンの解析をサポートしていません。L シリーズの RVU での測定解析の場合、Measure は、Measure H06/J04 以降の PVU で生成された H06 データファイルバージョンと、Measure L01 以降の PVU で生成された H07 データファイルバージョンをサポートします。

L シリーズの RVU で H06 データファイルの解析をサポートするには、H06 データファイルバージョンをサポートしている TNS/X バージョンの MEASFH（MEASFH06 と呼ばれます）を、個別の製品番号（T9186）を通じて利用できます。間違ったデータファイルか MEASFH バージョンのいずれかが指定された場合、エラー 3290（ERR^VERSION^NOT^SUPPORTED）が返されます。

H シリーズ以降の PVU および RVU についての ADD MEASUREMENT の使用上の注意

H06.07 から始まる J シリーズ以降の RVU では、MEASFT ユーティリティを使用して、プロセッサー番号やエンティティタイプなどの分割条件に従って測定データファイルを分割することができます。測定構成と OSS または ANSI ジャーナルデータは両方のファイルに追加されます。

MEASFT を使用して既存の G シリーズのデータファイルを分割することもできます。このような場合、最初に H シリーズの RVU を実行するシステムに G シリーズのデータファイルを移動する必要があります。MEASFT を実行してデータファイルを分割した後、新しいデータファイルを分析のために G シリーズの RVU に戻すか、オブジェクトファイルを指定する MEASFH パラメーターを使用して、MEASFT を実行したシステム上で適切な G シリーズバージョンの MEASFH を実行します。

MEASFT ユーティリティについて詳しくは、**Measure データファイルツール（MEASFT）**(672 ページ)を参照してください。

G シリーズの PVU および RVU についての ADD MEASUREMENT の使用上の注意

- Measure G05 より前では、分析のために開くことができる測定データファイルのサイズには、127.5 MB の上限が存在していました。Measure G05 以降の PVU では、測定データファイルのサイズは 1 GB です。大きいサイズを指定するには、FILESIZE オプションを使用します。詳しくは、**START MEASUREMENT の構文**を参照してください。

  大規模データファイルの解析のために MEASCOM を起動する前に、スワップファイルを格納するための十分なディスクスペース（たとえばデータファイルのサイズの 2 倍）があることを確認します。ディスクのスワップ領域に$SYSTEM を使用しないでください。スワップは、システムパフォーマンスと他のアプリケーションのメモリ可用性の妨げとなる可能性があります。MEASCOM が開始された $VOLUME に十分なスワップスペースが存在しない場合、MEASCOM セグメント（SEGMENT_allocate エラー 21）がオーバーフローしてトラップが発生することがあります。
- G シリーズの RVU を実行しているシステムでは、収集間隔がある場合に MEASCTL がデータレコードを書き込む方法が変更されるため、一部のデータファイルは D シリーズの RVU を実行するシステムでの同等の測定用のファイルよりも小さくなります。この相違は大規模な測定で最も顕著になり、特にすべての FILE エンティティおよび DISCOPEN エンティティを測定する測定で最も目立ちます。

ADD MEASUREMENT の例

- 2 つの測定データファイルを MEASCOM からアクセス可能にするには、以下のようにします（データファイル$DATA.MEAS.MAY04 は現在のデータファイルです）。

```
+ ADD MEASUREMENT $DATA.MEAS.MAY03
+ ADD MEASUREMENT $DATA.MEAS.MAY04
```

  後で以下のコマンドを入力すると、MAY03 が現在のデータファイルになります。

```
+ ADD MEASUREMENT $DATA.MEAS.MAY03
```

- 遠隔測定データファイルを現在のデータファイルとして指定するには、以下のようにします。

```
+ ADD MEASUREMENT \NY.$METS.MEAS.MAY06, &amp;
& MEASFH \NY.$SYSTEM.SYS14.MEASFH
```

ADD PLOT

ADD PLOT コマンドは、現在プロット定義にカウンターを追加します。

プロット定義がすでに開始されている場合、2 番目のデータファイルからのカウンターを追加すると、MEASCOM が警告メッセージを表示し、現在のプロット定義を削除し、新しいデータファイル用の新しいプロット定義を開始します。

ADD PLOT の構文

PROCESSH および USERDEF を除くすべてのエンティティタイプについて、構文は次のとおりです。

```
ADD [ PLOT ] counter [ ( record-number ) ]
```

PLOT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして PLOT を指定した場合はオプションです。

counter

LIST コマンドによって生成された最後のレポートから選択されるカウンター名です。レポートに複数のエンティティが表示される場合、ADD PLOT は、各エンティティからの指定されたカウンターをプロット定義に追加します。1 つのプロットに 26 個のカウンターをプロットできます。

record-number

プロットされる単一レコードの序数です（1～200）。

PROCESSH エンティティタイプの場合、構文は以下のとおりです。

```
ADD [ PLOT ] counter [ ( code-space ) ]
```

PLOT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして PLOT を指定した場合はオプションです。

counter

次のいずれかです。

procedure-name	個別のプロシージャーをプロットに追加します。
CODE-RANGE	すべてのコードサンプルを追加します。高速化、非高速化、およびネイティブモードのオブジェクトコード実行のための測定は、1 つのプロット項目にまとめられます。CODE-RANGE を使用するとき、その時点で実行されているコードに応じて、カウンターの一部が存在しない場合があります。
ACCEL-BUSY-SAMPLES	高速化されたコードサンプルを追加します。
TNS-BUSY-SAMPLES	TNS コードサンプルを追加します。
TNSR-BUSY-SAMPLES	TNS/R ネイティブコードサンプルを追加します。（G シリーズのみ）
TNSR-BUSY-TIME	TNS/R ネイティブコード領域を追加します。（G シリーズのみ）
NATIVE-BUSY-SAMPLES	ネイティブコードサンプルを追加します。（G シリーズまたは H シリーズ）
NATIVE-BUSY-TIME	ネイティブコード領域を追加します。（G シリーズまたは H シリーズ）
IPU**n**-BUSY-TIME	IPU ビジー時間コード領域を追加します。（J シリーズ以降）ここで、*n* は IPU 番号です。
IPU**n**-QTIME	IPU キュー時間コード領域を追加します。（J シリーズ以降）ここで、*n* は IPU 番号です。
IPU**n**-DISPATCHES	IPU ディスパッチを追加します。（J シリーズ以降）ここで、*n* は IPU 番号です。

code-space

コード領域指定です。高速化コードおよび TNS コードの場合は以下のようになります。

空き容量	仕様	高速化	TNS
ユーザーコード	UC[.*n*]、*n* は次の範囲内です。	0～31	0～15
ユーザーライブラリ	UL[.*n*]、*n* は次の範囲内です。	0～31	0～15
システムコード	SC[.*n*]、*n* は次の範囲内です。	0～31	0
システムライブラリ	SL[.*n*]、*n* は次の範囲内です。	0～31	0～31

n は、指定したタイプのすべてのコード領域になります。

適用できるコード領域の数には追加の制限事項がいくつか適用されます。詳しくは、**PROCESSH**(387 ページ)を参照してください。

TNS/R ネイティブコードの場合：

空き容量	仕様
ユーザーコード	UCR
ユーザーライブラリ	ULR
システムコード	SCR
システムライブラリ	SLR

H シリーズの RVU の場合、*code-space* 仕様は適用されません。指定する任意の値は受け入れられますが、無視されます。

USERDEF エンティティタイプの場合、構文は以下のとおりです。

```
ADD [ PLOT ] counter [ ( array-index ) ]
```

PLOT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして PLOT を指定した場合はオプションです。

counter

ユーザー定義カウンターの名前です。

array-index

配列内の項目のインデックス番号です（0～127）。

注記:

H07、J05、L シリーズ以降の PVU の配列内の項目のインデックス番号は 0～511 の範囲で指定できます。

関連コマンド

コマンド	機能
DELETE PLOT	プロット定義からカウンターを削除します
INFO PLOT	プロット定義を表示します
LIST PLOT	プロットを表示します

ADD PLOT の使用上の注意

LISTACTIVE コマンドによって表示されるデータをプロットすることはできません。

ADD PLOT の例

- プロット定義に 2 つのカウンターを追加するには、以下のようにします。

```
+ LIST CPU 1
        .
        .   (CPU レポートがここに表示されます)
        .
+ ADD PLOT CPU-BUSY-TIME
```

```
+ LIST FILE $DATA1.ACCT.BOOKS
        .
        .   (ファイルレポートがここに表示されます)
        .
+ ADD PLOT FILE-BUSY-TIME
```

- この例では、LIST コマンドによって表示される各ファイルから FILE-BUSY-TIME カウンターを追加します。カウンターは、LIST コマンドがカウンターを表示するのと同じ順序でプロットに追加されます。この例で、26 個を超えるファイルが表示される場合、最大の FILE-BUSY-TIME 値を持つ 26 ファイルからのカウンターがプロット定義に追加されます。

```
+ LIST FILE *, BY FILE-BUSY-TIME
        .
        .   (ファイルレポートがここに表示されます)
        .
+ ADD PLOT FILE-BUSY-TIME
```

- この例では、ACCEL-BUSY-TIME、TNS-BUSY-SAMPLES、および TNSR-BUSY-TIME カウンターを（カウンターの合計である CODE-RANGE を使用して）オブジェクトファイル $DATA.APPL.UPDKEYS から追加します。TIME-BASE OFF を指定してプロットを生成すると、3 つのカウンターが 1 行にプロットされます。例については、**LIST PLOT**(97 ページ)

を参照してください。

```
+ LIST PROCESSH $DATA.APPL.UPDKEYS
        .
        .   (PROCESSH レポートがここに表示されます)
        .
+ ADD PLOT CODE-RANGE
```

- この例では、C0-LS-SENDS カウンターの昇順で OSSCPU レコードを表示します。ここで、C0 はソケット操作に含まれる他の CPU の CPU 番号です。

```
+ LIST OSSCPU *, BY C0-LS-SENDS ( ASCENDING )
```

- この例では、CPU 1 の OSSCPU レコードを表示し、次に C6-LS-AWAKES カウンターの値のプロットを表示します。ここで、C6 はソケット操作に含まれる他の CPU の CPU 番号です。

```
+ LIST OSSCPU 1
+ ADD PLOT C6-LS-AWAKES
+ LIST PLOT
```

ASSUME

ASSUME コマンドは、デフォルトのコマンドオブジェクトを設定します。ほとんどの MEASCOM コマンドは、アクションとオブジェクトで構成されています。オブジェクト名はオプションです。オブジェクト名が省略されると、MEASCOM は最後に出された ASSUME コマンドによって指定されたデフォルトのコマンドオブジェクトを使用します。最初のデフォルトのコマンドオブジェクトは MEASUREMENT です。

ASSUME の構文

```
ASSUME object
```

object

次のいずれかです。

```
COUNTER  MEASSUBSYS  MEASUREMENT  PLOT  REPORT  entity-type
```

entity-type

次のいずれかです。

```
CLUSTER     DISCOPEN   LINE       PROCESS    SYSTEM
CONTROLLER  DISKFILE   NETLINE    PROCESSH   TERMINAL
CPU         ENDPOINT   OPDISK     SERVERNET  TMF
DEVICE      FABRIC     OSSCPU     SQLPROC    USERDEF
DISC        FILE       OSSNS      SQLSTMT
```

ASSUME の例

ASSUME コマンドを使用してコマンドラインを短くするには、以下のようにします。

```
+ ADD $DATA.MEAS.MAY04    -- MEASUREMENT がデフォルトのオブジェクト
+ ASSUME REPORT
+ SET FORMAT BRIEF
+ SET TOTALS INCLUDE
+ ASSUME CPU
+ LIST 2
    .
    .  (CPU 2 用のレポートがここに表示されます)
    .
+ LIST 4
    .
    .   (CPU 4 用のレポートがここに表示されます)
    .
+ LIST 6
    .
    .   (CPU 6 用のレポートがここに表示されます)
    .
```

COMMENTS

COMMENTS メッセージは、表示されるコメントメッセージを指定します。コメントは、COMMENT という単語と 2000~2999 の範囲の数字で識別されます。

COMMENTS の構文

```
COMMENTS { DISPLAY  { ALL | comm-num [, comm-num ] ... } }
         { SUPPRESS { ALL | comm-num [, comm-num ] ... } }
```

DISPLAY

表示されるコメントメッセージを指定します。

ALL

1 つまたは複数の後続の COMMENTS SUPPRESS コマンドによって指定されたものを除き、すべてのコメントを表示します。COMMENTS DISPLAY ALL がデフォルトです。

comm-num

コメントの番号です（2000～2999）。

SUPPRESS

抑制するコメントメッセージを指定します。

ALL

1 つまたは複数の後続の COMMENTS DISPLAY コマンドによって指定されたものを除き、すべてのコメントを抑制します。

関連コマンド

コマンド	機能
WARNINGS	表示される警告メッセージを指定します。

COMMENTS の使用上の注意

Measure には番号 2000 の 1 つのコメントのみが含まれます。

COMMENTS の例

コメント 2000 を抑制するには、以下のようにします。

```
+ COMMENTS SUPPRESS 2000
```

DELETE entity-type

DELETE *entity-type* コマンドは、指定したタイプのエンティティを現在の構成から削除します。

DELETE *entity-type* コマンドを ADD *entity-type* *コマンドと組み合わせて使用します。DELETE *entity-type* により、特定のリソースを測定から除外することができます。

DELETE entity-type の構文

```
DELETE [ entity-type ] entity-spec [ , entity-spec ] ...
```

entity-type

次のいずれかです。

```
CLUSTER     DISCOPEN   LINE      PROCESS     SYSTEM
CONTROLLER  DISKFILE   NETLINE   PROCESSH    TERMINAL
CPU         ENDPOINT   OPDISK    SERVERNET   TMF
DEVICE      FABRIC     OSSCPU    SQLPROC     USERDEF
DISC        FILE       OSSNS     SQLSTMT
```

entity-type キーワードは、ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして指定された場合はオプションです。

entity-spec

測定対象のエンティティを識別します。エンティティ指定は各エンティティタイプに固有です。構文については、**エンティティおよびカウンター**(167 ページ)にある指定したエンティティタイプの説明を参照してください。

関連コマンド

コマンド	機能
ADD *entity-type*	エンティティを構成に追加します
INFO *entity-type*	構成を表示します

すべての PVU および RVU での DELETE entity-type の使用上の注意

PROCESSH エンティティで、測定からエンティティ全体を削除することはできますが、個々のコード領域指定子を削除することはできません。

L シリーズの PVU および RVU での DELETE entity-type の使用上の注意

L シリーズの PVU および RVU では、DELETE SERVERNET コマンドは、GMS、ポート、およびファイバー仕様をサポートしません。L シリーズの RVU の SERVERNET エントリーでは、CLIM に関連する IPC および RIPC ノードクラスのみが利用できます。

H シリーズ以降の PVU および RVU についての DELETE entity-type の使用上の注意

- Measure H03/J01 以降の PVU では、DELETE コマンドは、CLIM を使用する SERVERNET、DEVICE、および DISC エンティティ指定を可能にします。構文と例については、**SERVERNET**(399 ページ)、**DEVICE**(214 ページ)、および **DISC**(226 ページ)エンティティを参照してください。
- Measure H01 以降の PVU では、DELETE コマンドは、ANSI SQL 名を使用する DISCOPEN、DISKFILE、および FILE のエンティティ指定を可能にします。構文については、**DISCOPEN**(249 ページ)、**DISKFILE**(259 ページ)、および **FILE**(284 ページ)エンティティを参照してください。

DELETE entity-type の例

- ANSI SQL 名を使用してエンティティを削除するには、以下のようにします。

```
+ DELETE FILE 'TABLE CATALOG_12.SCHEMA_34.TABLE_56'
+ DELETE DISCOPEN 'TABLE CATALOG_12.SCHEMA_34.TABLE_56 PARTITION PART_78'
+ DELETE DISKFILE 'SCHEMA SCHEMA_34'
```

- CPU 6 を除くすべての CPU を測定する構成を定義するには、以下のようにします。

```
+ ADD CPU *
+ DELETE CPU 6
+ INFO CPU
Add cpu *
Delete cpu 6
```

- ACCT.BILLING 以外の ACCT サブボリュームのすべてのプログラムで TRANSACTIONS カウンターを測定するには、以下のようにします。

```
+ ADD USERDEF ACCT.*
+ DELETE USERDEF ACCT.BILLING
+ ADD COUNTER TRANSACTIONS, PROCESS ACCT.*, TYPE ACCUM
```

- エンティティのサブセットを測定から除外するには、最初に ADD コマンドを使用してセットを追加し、次に DELETE コマンドを使用してサブセットを削除します。たとえば、CPU 0 を除くすべての CPU を測定する場合は、次のように入力します。

```
+ ADD CPU *
+ DELETE CPU 0
+ START MTEST
```

- $SYSTEM を除くすべてのディスクのすべてのファイルを測定し、ディスク$SYSTEM のサブボリューム SYS00 内のファイルのみ測定するには、$SYSTEM を除く個々のディスクを追加し、次に $SYSTEM.SYS00 サブボリュームを追加します。

```
+ ADD FILE $DISC1.*.*
+ ADD FILE $DISC2.*.*
+ ADD FILE $DISC3.*.*
+ ADD FILE $DISC4.*.*
+ ADD FILE $DISC5.*.*
+ ADD FILE $SYSTEM.SYS00.*
```

注記:

削除された項目のサブセットを追加することはできません。ワイルドカードを使用してすべてのファイルを指定し、次に$SYSTEM ボリュームを削除してから、$SYSTEM ボリュームの SYS00 サブボリュームを追加し直すということはできません（つまり、次の例は動作**しません**）。

```
+ ADD FILE $*
+ DELETE FILE $SYSTEM.*.*
+ ADD FILE $SYSTEM.SYS00.*
```

システム上のディスクが多すぎるため別々に追加できない場合、`ADD FILE $*`コマンドを使用してすべてのファイルを測定してから、LIST コマンドを使用して必要なファイルを表示します。

DELETE COUNTER

DELETE COUNTER コマンドは、現在の構成から 1 つまたは複数のユーザー定義カウンターを削除します。

DELETE COUNTER の構文

```
DELETE [ COUNTER ] counter-name [ , counter-name ] ...
```

COUNTER

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして COUNTER を指定した場合はオプションです。

counter-name

削除されるカウンターの名前です。現在の構成からすべてのユーザー定義カウンターを削除するには、アスタリスク（*）を使用します。

注意:

DELETE COUNTER は、カウンターが異なるプロセスによって変更される場合でも、指定された名前を持つ構成済みのすべてのカウンターを削除します。

関連コマンド

コマンド	機能
ADD COUNTER	ユーザー定義のカウンターを構成に追加します
INFO COUNTER	構成内のユーザー定義のカウンターを表示します

DELETE COUNTER の例

ユーザー定義カウンター TRANSACTIONS を削除するには、次のようにします（両方の TRANSACTIONS カウンターは 2 つの異なるプロセスによって変更されますが、両方とも削除されます）。

```
+ INFO COUNTER *
Add counter TRANSACTIONS, process $PERF.QUOTAS.BILLING, accum
Add counter WAITING, process $PERF.QUOTAS.BILLING, busy
Add counter TRANSACTIONS, process $PERF.QUOTAS.INCOMING, accum
+ DELETE COUNTER TRANSACTIONS
+ INFO COUNTER *
Add counter WAITING, process $PERF.QUOTAS.BILLING, busy
```

DELETE MEASUREMENT

DELETE MEASUREMENT コマンドは、MEASCOM でアクセスできる測定データファイルのセットから、1 つまたは複数の測定データファイルを削除します。

DELETE MEASUREMENT の構文

```
DELETE [ MEASUREMENT ] data-file [ , data-file ] ...
```

MEASUREMENT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして MEASUREMENT を指定した場合はオプションです。

data-file

削除する測定データファイルの名前です。すべてのアクセス可能なデータファイルを指定するには、アスタリスク（*）を使用します。

関連コマンド

コマンド	機能
ADD MEASUREMENT	測定データファイルにアクセスできるようにします

DELETE MEASUREMENT の使用上の注意

使用していないデータファイルへのアクセスを中止するには、このコマンドを使用します。測定を削除すると、測定に関連付けられているファイル処理（MEASFH）を停止し、データファイルのために維持されている一時的構造を解放することでシステムリソースが節約されます。

DELETE MEASUREMENT の例

2 つのデータファイルを MEASCOM によってアクセスできるようにし、その後、$DATA.MEAS.MAY03 をアクセス不可にするには、次のようにします。

```
+ ADD MEASUREMENT $DATA.MEAS.MAY03
+ ADD MEASUREMENT $DATA.MEAS.MAY04
                  .
                  .
                  .
+ DELETE MEASUREMENT $DATA.MEAS.MAY03
```

DELETE PLOT

DELETE PLOT コマンドは、現在のプロット記述からカウンターを削除します。

DELETE PLOT の構文

```
DELETE [ PLOT ] counter [ ( char ) ] [ , counter [ ( char ) ]    ... ]
```

PLOT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして PLOT を指定した場合はオプションです。

counter

PROCESSH と USERDEF を除くすべてのエンティティタイプでは、COUNTER は削除するカウンターの名前です。プロット記述のすべてのカウンターを指定するには、アスタリスク（*）を使用します。

PROCESSH エンティティタイプでは、*counter* はプロシージャー名です。すべての PROCESSH プロシージャーを削除するには、*counter* に CODE-RANGE を指定します。

USERDEF エンティティタイプでは、*counter* はユーザー定義のカウンター名です。すべてのユーザー定義カウンターを削除するには、*counter* に NAME を指定します。

char

カウンターを記述するために以前指定されていた文字です。*counter* という名前のすべてのカウンターを指定するには、アスタリスク（*）を使用します。

現在のプロット記述に同じ名前を持つ複数のカウンターが含まれている場合、*char* を指定する必要があります。

関連コマンド

コマンド	機能
ADD PLOT	プロット記述にカウンターを追加します
INFO PLOT	プロット記述を表示します
LIST PLOT	プロットを表示します

DELETE PLOT の例

2 つの CPU-BUSY-TIME カウンターのいずれかを削除するには、次のようにします（2 つ以上のカウンターが同じ名前を持つ場合、警告メッセージが表示されます）。

```
+ INFO PLOT *
Add measurement $SPOOL.QUOTAS.DATAGO
--A-- List Cpu 6, &
        To 20 Aug 1994, 14:29:08
      Add plot CPU-BUSY-TIME
--B-- List Cpu 7, &
        To 20 Aug 1994, 14:29:08
      Add plot CPU-BUSY-TIME
+ DELETE PLOT CPU-BUSY-TIME
MEAS 3057 Ambiguous DELETE command;
   specify which occurrence to delete.
+ DELETE PLOT CPU-BUSY-TIME (A)
+ INFO PLOT *
Add measurement $SPOOL.QUOTAS.DATAGO
--B-- List Cpu 7, &
        To 20 Aug 1994, 14:29:08
      Add plot CPU-BUSY-TIME
```

ENV

ENV コマンドでは、1 つまたはすべての MEASCOM セッション環境パラメーターの値が表示されます。

Measure H01 以降の PVU では、ENV コマンドは **SQLCATALOG**(149 ページ)および **SQLSCHEMA**(149 ページ)コマンドの現在の設定を表示します。

Measure G09 以降の PVU では、ENV コマンドは OSSPATH および PAGESIZE の現在の設定も表示します。

ENV の構文

```
ENV [ / OUT filename / ] [ env-param ]
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は編集ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

ENV コマンドを実行した後、MEASCOM は現在の OUT ファイル（通常、端末）への出力の書き込みを再開します。

OUT オプションは、ログファイルの内容には影響しません。ログファイルについて詳しくは、**LOG**(126 ページ)を参照してください。

env-param

次のいずれかです。

ASSUME	デフォルトのコマンドオブジェクトを表示します
COMMENTS	COMMENTS コマンドの現在の設定を表示します
LOG	デフォルトのログファイルの名前を表示します

表は続く

OSSPATH	現在のデフォルトの OSSPATH を表示します
OUT	デフォルトの出力ファイルの名前を表示します
PAGESIZE	現在の PAGESIZE 設定が表示されます
SETPROMPT	現在のプロンプトテキストを表示します
SQLCATALOG	部分的に修飾された ANSI SQL 名を解決するために使用される現在のデフォルトのカタログ値を表示します
SQLSCHEMA	部分的に修飾された ANSI SQL 名を解決するために使用される現在のデフォルトのスキーマ値を表示します
SWAPVOL	MEASCOM スワップボリュームと異なる場合にスワップボリューム名を表示します。
SYSTEM	デフォルトシステムの名前を表示します
VOLUME	デフォルトのボリュームおよびサブボリュームの名前を表示します
WARNINGS	WARNINGS コマンドの現在の設定を表示します

env-param を省略すると、すべての環境パラメーター値が表示されます。

ENV の使用上の注意

MEASCOM では、環境パラメーターを設定するために使用するコマンドの形式で環境情報を表示します。この方法により、/OUT *filename*/を使用してコマンド（OBEY）ファイルを作成できます。このコマンドファイルを使用して、特定の環境パラメーターのセットを復元することができます。

ENV の例

- 環境パラメーターを表示するには、次のようにします（明示的に設定した場合、OUT、LOG、および SWAPVOL パラメーターがこの表示に含められます）。

```
+ ENV
  System    \BUYER
  Volume     $DATA.MEAS
  Assume     MEASUREMENT
  Setprompt Assume
  Comments   Display All
  Warnings   Display All
```

- Measure H01 以降の PVU で、ANSI SQL 名を含む環境パラメーターを表示するには、次のようにします。

```
+ ENV
  System      \BUYER
  Volume      $DATA.MEAS
  OSSpath     "/This/is/the/path/"
  SQLCATALOG 'CATALOG_12'
  SQLSCHEMA  'SCHEMA_34'
  Assume      MEASUREMENT
  Setprompt   Assume
  Pagesize    24
```

```
  Comments    Display All
  Warnings    Display All
```

たとえば、ANSI SQL テーブル名を次のように指定します。

```
'TABLE table_56'
```

MEASCOM は先頭にデフォルトのカタログおよびスキーマを付加するため（設定されている場合）、完全修飾名は次のようになります（上記の値のとき）。

```
'TABLE CATALOG_12.SCHEMA_34.table_56'
```

- Measure G09 以降の PVU で、OSSPATH の現在のデフォルトパス、および PAGESIZE の現在の設定をそれぞれ表示するには、次のようにします。

```
+ ENV
  System    \BUYER
  Volume     $DATA.MEAS
  OSSpath    "/This/is/the/path/"
  Assume     MEASUREMENT
  Setprompt Assume
  Pagesize   24
  Comments   Display All
  Warnings   Display All

+ ENV OSSPATH
  OSSpath    "/This/is/the/path"
```

EXIT

EXIT コマンドにより MEASCOM セッションが終了します。

EXIT の構文

```
EXIT
```

関連コマンド

コマンド	機能
MEASCOM	MEASCOM セッションを開始します。

EXIT の使用上の注意

- MEASCOM は、停止または異常終了するときに完了コードを返します。完了コードは、次のとおりです。

コード	意味
0	正常な自発的終了です。エラーや警告は発行されませんでした。
1	正常な自発的終了です。警告が発行されました。
2	異常な自発的終了です。エラーメッセージが発行されました。
3	異常な早期終了です。MEASCOM でエラー状態が発生し、エラーが原因で早期に終了しました。

コードが 0 の場合、メッセージは表示されませんが、TACL プロンプトが再び表示されます。コードが 1、2、または 3 の場合、説明メッセージが表示されます。

- MEASCOM を呼び出すときに IN ファイルを指定する場合、ファイルの最後に達することは EXIT コマンドと同等です。

EXIT の例

この例では、自発的な異常終了についての EXIT 表示を示します。

```
+ EXIT
CPU time 0:00:02.430
2: Process terminated with fatal errors or diagnostics
```

FC

FC コマンドは TACL FC コマンドのように動作します。FC によって、コマンドを履歴バッファーから取得し、コマンドを表示してユーザーが変更できるようにし、変更されたコマンドを実行します。FC の完全な説明については、*TACL Reference Manual* を参照してください。

FC の構文

```
FC [ number  ]
   [ -number ]
   [ text    ]
```

number

取得するコマンドが含まれる履歴バッファー内の行数です。

-number

取得するコマンドに到達するために現在の行から減算する履歴バッファーの行数です。

text

テキスト文字列です。このテキスト文字列で始まる最後に入力されたコマンドが取得されます。

オプションを指定しない場合は、入力された最後のコマンドが取得されます。

関連コマンド

コマンド	機能
HISTORY	前に入力したコマンドを表示します
!	前のコマンドに変更を加えずに、すぐに再実行します

FC の例

- 履歴バッファーの 15 行目のコマンドを取得するには、以下のようにします。

  ```
  21+ FC 15
  ```

- 現在の行より 3 行前のコマンドを取得するには、以下のようにします。

  ```
  24+ FC -3
  ```

- テキスト文字列 SE で始まる最後に入力されたコマンド（たとえば、SET REPORT FORMAT BRIEF コマンド）を取得するには、以下のようにします。

  ```
  25+ FC SE
  ```
- 入力された最後のコマンドを取得するには、以下のようにします。

  ```
  28+ FC
  ```

HELP

HELP コマンドは、コマンド、オブジェクト、エンティティ、およびエラーメッセージに関するクイックリファレンス情報を表示します。

HELP の構文

```
HELP [ / OUT filename / ] [ command-name         ]
                         [ command-name object ]
                         [ command-name ENTITY ]
                         [ command-name entity ]
                         [ object              ]
                         [ entity               ]
                         [ counter              ]
                         [ entity COUNTERS      ]
                         [ error-number         ]
                         [ ABBREVIATIONS        ]
                         [ ALL                  ]
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は編集ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

HELP コマンドを実行した後、MEASCOM は現在の OUT ファイル（通常、端末）への書き込みを再開します。

OUT オプションは、ログファイルの内容には影響しません。

command-name

指定されたコマンドの簡単な説明を提供します （例：HELP ADD）。

command-name object

指定されたコマンドの完全な構文を示します （例：HELP ADD MEASUREMENT）。

***command-name* ENTITY**

オブジェクトがエンティティタイプであるコマンドの一般構文に関する情報を提供します（例：HELP ADD ENTITY）。

command-name entity

指定したエンティティタイプの構文の詳細情報を提供します （例：HELP ADD CPU）。

object

各属性の値を含む、指定したオブジェクトの属性について説明します （例：HELP REPORT）。

entity

指定したエンティティタイプの簡単な説明を提供します （例：HELP CONTROLLER）。

counter

指定したカウンターの簡単な説明を提供し、この名前のカウンターを使用するエンティティを識別します （例：HELP RESPONSE-TIME）。

***entity* COUNTERS**

指定したエンティティのカウンターの値を示します （例：HELP CPU COUNTERS）。

error-number

指定した Measure エラーメッセージに関する原因、結果、および回復処置の情報を提供します（例：HELP 3100）。

ABBREVIATIONS

アルファベット順にすべての Measure キーワード（コマンド、オブジェクト、およびエンティティ）を示します。このリストを使用して、キーワードを一意に省略するために必要な文字数を確認します。

ALL

すべての Measure キーワード（コマンド、オブジェクト、および属性）をカテゴリごとに示し、詳細なヘルプトピックのガイドを提供します。パラメーターを付けずに HELP を実行した場合は、HELP ALL を実行した場合と同じになります。

HELP の使用上の注意

- ヘルプトピックを省略することはできません。たとえば、HELP START MEASU ではなく、HELP START MEASUREMENT を入力する必要があります。
- 二重プロンプト（++）は、トピックに詳細情報を表示できることを示します。その他の情報を表示するには、Return キー（または任意のその他のキー）を押します。Ctrl-Y を押すか、二重プロンプトで BREAK と入力して、コマンドエントリーに戻ります。

HELP の例

- SET コマンドに関する情報を取得するには、次のように入力します。

```
+ HELP SET
The SET command can operate on the PLOT and REPORT objects.
SET's function is to change the value of the attribute
specified to the value entered.  Each time MEASCOM is
started, the PLOT and REPORT objects are initialized to
default values.  These values can be changed by using the SET
command, or can be temporarily changed by setting the
attribute on the LIST (LISTALL, and LISTACTIVE) command.  In
this case the value is changed only for the duration of the
command.

For specific command syntax perform:
  HELP SET PLOT
  HELP SET REPORT
```

- SET PLOT コマンドの構文を取得するには、次のように入力します。

```
+ HELP SET PLOT
SET [ PLOT ] <plot attribute> <value> [, <plot attribute>
<value> ]...

See HELP PLOT for <plot attribute> and <value>.
```

- READY-TIME カウンターを使用するエンティティを識別し、カウンターの意味の説明を取得するには、次のように入力します。

```
+ HELP READY-TIME
CPU Report:
  Time processes were executing or waiting on the ready list.
  Counter type: Queue

Process Report:
  Time this process was executing or waiting on the ready list.
  Counter type: Busy
```

- Measure エラー 3014 に関する情報を表示するには、次のように入力します。

```
+ HELP 3014
Cause:    A three-digit sequence-id is added to LOADID to
          designate each interval of the measurement, when
          "LISTALL" is used. It is suppressed if 3 characters
          are not available for use.
Result:   Sequencing is suppressed.
Recovery: Use 5 or fewer characters for LOADID.
```

HISTORY

HISTORY コマンドは、最も新しく入力された 1 つまたは複数のコマンドを表示します。

HISTORY の構文

```
HISTORY [ number ]
```

number

表示する履歴バッファーの行数（0～999）を指定します。数を指定しない場合、履歴バッファーの最後の 10 行が表示されます。バッファーに含まれる行数が指定した行数より少ない場合、HISTORY は既存の行を表示します。

関連コマンド

コマンド	機能
FC	前に入力したコマンドを表示、変更、および実行します
!	前に入力したコマンドを再実行します

HISTORY の例

この例は、HISTORY コマンドおよび FC コマンドの両方を表示します。HISTORY コマンドは、履歴バッファー内の最後の 5 つの行（28～32 行）を示します。FC コマンドは、履歴バッファーから 31 行目を再表示します。FC を使用した後で、コマンドを再実行するか、コマンドを編集してから再実行できます。

```
32+ HISTORY 5

28+ LIST PROCESS (0,*), RATE OFF
29+ INFO MEASUREMENT *
30+ LIST DISC $SYSTEM
31+ LIST CPU *, BY CPU-BUSY-TIME
32+ HISTORY 5

33+ FC 31
```

```
33+ LIST CPU *, BY CPU-BUSY-TIME
33 ...
```

INFO entity-type

INFO *entity-type* コマンドは、現在の構成における、指定したタイプのエンティティをすべて表示します。

INFO entity-type の構文

```
INFO [ / OUT filename / ] [ entity-type ]
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

INFO コマンドを実行した後、MEASCOM は現在の OUT ファイル（通常、端末）への書き込みを再開します。OUT オプションは、ログファイルには影響しません。

entity-type

次のいずれかです。

```
CLUSTER     DISCOPEN   LINE      PROCESS     SYSTEM
CONTROLLER  DISKFILE   NETLINE   PROCESSH    TERMINAL
CPU         ENDPOINT   OPDISK    SERVERNET   TMF
DEVICE      FABRIC     OSSCPU    SQLPROC     USERDEF
DISC        FILE       OSSNS     SQLSTMT
```

entity-type キーワードは、ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして指定された場合はオプションです。

関連コマンド

コマンド	機能
ADD *entity-type*	エンティティを構成に追加します
DELETE *entity-type*	エンティティを構成から削除します

すべての PVU および RVU での INFO entity-type の使用上の注意

- INFO 表示にエンティティが追加対象および削除対象の両方として表示されている場合、削除操作が優先されます。
- MEASCOM は、構成を定義するために使用するコマンドのリストとして情報を表示します。この機能では、/OUT *filename*/を使用して、現在の測定構成を後から復元できるコマンド（OBEY）ファイルを作成できます。
- INFO PROCESSH コマンドの出力は、現在の PROCESSH 測定構成を定義するために使用されるコマンドのリストになります。/OUT *filename*/を使用して、指定した PROCESSH 構成を格納するコマンド（OBEY）ファイルを作成できます。

L シリーズの PVU および RVU での INFO entity-type の使用上の注意

L シリーズの PVU および RVU では、INFO コマンドは、SERVERNET エンティティの GMS、ポート、およびファイバー指定をサポートしません。

H シリーズの PVU および RVU での INFO entity-type の使用上の注意

- H06.23/J06.12 以降の RVU では、測定にパーティション番号を指定した場合、INFO DISC コマンドは CLIM 接続ディスクのパーティション番号を表示します。パーティション番号は、パーティション分割されたディスク内のパーティションに割り当てられている番号です。**INFO entity-type の例**(69 ページ)を参照してください。
- Measure H01 以降の PVU では、INFO *entity-type* コマンドは、測定構成に ANSI SQL 名を使用した場合、ANSI SQL 名を使用して DISCOPEN、DISKFILE、および FILE エンティティ指定を表示します。構文については、**DISCOPEN**(249 ページ)、**DISKFILE**(259 ページ)、および **FILE**(284 ページ)エンティティを参照してください。

G シリーズの PVU での INFO entity-type の使用上の注意

Measure G12 以降の PVU では、TNS コードファイルでのみコード領域識別子が表示されます（高速化 TNS コードファイルを含む）。

INFO entity-type の例

- 以下の INFO DISC コマンドの表示には、$DISC1 という名前の CLIM 接続ディスクのパーティション番号（**太字**で強調表示）が示されます。

```
+ INFO DISC

Add Disc $DISC1 (3, X, C1002531, 308, 1)
```

- 現在の構成におけるすべての CPU エンティティを表示するには、次のように入力します。

```
+ INFO CPU
Add cpu *
Delete cpu 6
```

- 次の例は、G シリーズでの例です。コード領域識別子が表示されていないことに注意してください。

```
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP (UCR $SYSTEM.SYS00.TCPIP)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP (ULR $SYSTEM.SYS00.ZCRESRL)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP (ULR $SYSTEM.SYS00.ZINETSRL)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP (ULR $SYSTEM.SYS00.ZLANCSRL)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP (ULR $SYSTEM.SYS00.ZCRTLSRL)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP (SCR $SYSTEM.SYS00.TSYSCLR)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP (SLR $SYSTEM.SYS00.TSYSCLR)
+ INFO PROCESSH
Add Processh $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.TCPIP)
Add Processh $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.ZCRESRL)
Add Processh $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.ZINETSRL)
Add Processh $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.ZLANCSRL)
Add Processh $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.ZCRTLSRL)
Add Processh $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.TSYSCLR)
```

- 次に、L シリーズ以前の INFO SERVERNET コマンドの表示例を示します。

```
+ info servernet
Add Servernet 2 (*, 100, 2, 5, 3, 1)
```

- 次に、L シリーズの INFO SERVERNET コマンドの表示例を示します。GMS、ポート、ファイバー指定は表示されません。

```
+ info servernet
Add Servernet 2, C1002531
```

INFO COUNTER

INFO COUNTER コマンドは、1 つまたは複数のユーザー定義のカウンターについて、カウンターの名前とタイプ、およびカウンターを修正するプロセスの名前を表示します。

INFO COUNTER の構文

```
INFO [ / OUT filename / ] [ COUNTER ]
     counter-name [ , counter-name ] ...
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

INFO コマンドを実行した後、MEASCOM は現在の OUT ファイル（通常、端末）への出力の書き込みを再開します。

OUT オプションは、ログファイルの内容には影響しません。

COUNTER

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして COUNTER を指定した場合はオプションです。

counter-name

カウンターの名前、またはすべてのカウンターを示すアスタリスク（*）です。

関連コマンド

コマンド	機能
ADD COUNTER	ユーザー定義のカウンターを構成に追加します
DELETE COUNTER	ユーザー定義のカウンターを構成から削除します

すべての PVU および RVU での INFO COUNTER の使用上の注意

MEASCOM は、カウンターを定義するために使用するコマンドのリストとして情報を表示します。この機能では、/OUT *filename*/を使用して、現在の一連のカウンターを後から復元できるコマンド（OBEY）ファイルを作成できます。

ユーザー定義のカウンター情報を保存するときに、現在の USERDEF 構成情報も保存できます。ユーザー定義の構成情報を保存するには、OUT パラメーターとともに INFO USERDEF コマンドを使用します。

G シリーズの PVU での INFO COUNTER の使用上の注意

Measure G12 以降の PVU では、TNS コードファイルでのみコード領域識別子が表示されます（高速化 TNS コードファイルを含む）。

INFO COUNTER の例

現在の構成におけるすべてのユーザー定義カウンターを表示するには、次のように入力します。

```
+ INFO COUNTER *
Add counter TRANSACTIONS, process $PERF.QUOTAS.BILLING, accum
Add counter WAITING, process $PERF.QUOTAS.BILLING, busy
Add counter TRANSACTIONS, process $PERF.QUOTAS.INCOMING, accum
```

INFO MEASUREMENT

INFO MEASUREMENT コマンドは、MEASCOM で現在アクセスできる 1 つまたは複数の測定データファイルに関する次の情報を表示します。

- ファイル名。ADD MEASUREMENT コマンドとして表示されます。
- 測定の開始時間、停止時間、および収集間隔。停止時間は、測定の開始時に停止時間を指定した場合にのみ、現在アクティブな測定について表示されます。TIME 分割データファイルの場合、表示される開始時間と停止時間は元のデータファイルのものです。TIME 分割データファイルの期間情報を取得するには、MEASFT INFO コマンドを使用する必要があります。
- 測定中のシステムデータ領域使用量に関する情報。エンティティタイプごとに、次の情報が含まれます。
  - エンティティタイプ名
  - そのタイプのエンティティで同時に測定されているエンティティの最大数
  - そのタイプのエンティティに割り当てられるシステムデータ領域内の単語の最大数

  これらの値は、現在アクティブなデータファイルについては表示されません。分割データファイルの場合、これらの値は元のデータファイルの値になります。
- 測定構成。ADD コマンドおよび DELETE コマンドを含むコメントとして表示されます。分割データファイルの場合、これらの値は元のデータファイルの値になります。
- データファイル内の OSS ジャーナルセグメントと、MEASCOM セッション内の OSS ジャーナルセグメントの現在のステータス（Measure G09 以降の PVU のみ）。
- データファイル内の ANSI SQL ジャーナルセグメントと、MEASCOM セッション内の ANSI SQL ジャーナルセグメントの現在のステータス（Measure G11 以降の PVU のみ）。

INFO MEASUREMENT の構文

```
INFO [ / OUT filename / ] [ MEASUREMENT ] data-file
     [ , data-file ] ...
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

INFO コマンドを実行した後、MEASCOM は現在の OUT ファイル（通常、端末）への出力の書き込みを再開します。

OUT オプションは、ログファイルの内容には影響しません。

MEASUREMENT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとしてこれを指定した場合はオプションです。

data-file

アクセス可能な測定データファイルの名前です。すべてのアクセス可能なデータファイルを指定するには、アスタリスク（*）を使用します。

関連コマンド

コマンド	機能
ADD MEASUREMENT	MEASCOM がデータファイルにアクセスできるようにします

すべての RVU での INFO MEASUREMENT の使用上の注意

- INFO MEASUREMENT 表示では、現在のデータファイル（レポートの生成時に MEASCOM が使用するファイル）の名前には「Current Data File」というコメントが続きます。別のファイルを現在のデータファイルにするには、ADD MEASUREMENT コマンドを使用します。
- INFO MEASUREMENT コマンドの出力は、構成を定義するために使用されるコマンドのリストになります。/OUT *filename*/を使用して、コマンド（OBEY）ファイルを作成できます。続いてこのファイルを編集して、新しい構成ファイルを作成できます。
- OSS ファイルパス名が測定の指定で使用される場合、INFO MEASUREMENT レポートがフォーマットされる時点で、指定されたパス名が変換可能になっている必要があります。なっていない場合、フォーマットエラーが発生します。この変換は、測定中のシステム、または OSS ファイルパス名から取得できます。最後の例はこの動作を強調しています。

H シリーズの PVU および RVU での INFO MEASUREMENT の使用上の注意

- H06.23/J06.12 以降の RVU では、測定でパーティション番号が指定された場合、INFO MEASUREMENT コマンドによって CLIM 接続ディスクのパーティション番号が表示されます。パーティション番号は、パーティション分割されたディスク内のパーティションに割り当てられている番号です。
- Measure H02 以降の PVU では、INFO MEASUREMENT コマンドで生成される出力に、測定でカウンターデータレコードが抑制されているかどうかが表示されます。
- Measure H01 以降の PVU では、INFO MEASUREMENT コマンドは、ANSI SQL 名が測定構成に使用されていた場合、ANSI SQL 名を使用して DISCOPEN、DISKFILE、および FILE エンティティ指定を表示します。エンティティ指定については、**DISCOPEN**(249 ページ)、**DISKFILE**(259 ページ)、または **FILE**(284 ページ)を参照してください。

G シリーズの PVU での INFO MEASUREMENT の使用上の注意

- Measure G12 以降の PVU では、TNS コードファイルでのみコード領域識別子が表示されます（高速化 TNS コードファイルを含む）。
- Measure G11 以降の PVU では、ANSI SQL ジャーナルセグメントの構築が試みられた場合、そのステータスが表示されます （Measure は、**START MEASSUBSYS**(150 ページ)の使用上の注意で説明しているように、START MEASUREMENT コマンドで明示的に要求するか、ジャーナルセグメントの構築をデフォルトにすることによって要求した場合、ジャーナルセグメントの構築を試みます）。ANSI SQL 名セグメントの状態は次のとおりです。

  - SQL journal segment attached
  - SQL journal segment under construction
  - SQL journal segment error: nnnn

  ジャーナルセグメントの構築は測定が停止した後で行われます。構築が完了したときに、ファイルの現在のオープナーには通知されません。データファイルを開いている MEASCOM セッションは、ANSI SQL を使用して ANSI SQL 名参照を解決できます。ジャーナルセグメントにアクセスするには、DELETE MEASUREMENT コマンドを実行してから、元の ADD MEASUREMENT コマンドを繰り返します。
- Measure G09 以降の PVU では、Measure がデータファイルで OSS ジャーナルセグメントの構築を試みた場合、そのステータスが表示されます （Measure は、**START MEASSUBSYS**(150 ページ)の使用上の注意で説明しているように、START MEASUREMENT コマンドで明示的に要求するか、ジャーナルセグメントの構築をデフォルトにすることによって要求した場合、ジャーナルセグメントの構築を試みます）。OSS ジャーナルセグメントの状態は次のとおりです。

  - OSS journal segment attached
  - OSS journal segment under construction
  - OSS journal segment error: nnnn

  ジャーナルセグメントの構築は測定が停止した後で行われます。構築が完了したときに、ファイルの現在のオープナーには通知されません。データファイルを開いている MEASCOM セッションは、OSS ファイルシステムネームサーバーを使用して、OSS ファイルパス名参照を解決できます。ジャーナル

セグメントにアクセスするには、DELETE MEASUREMENT コマンドを実行してから、元の ADD MEASUREMENT コマンドを繰り返します。

L シリーズの PVU での INFO MEASUREMENT の使用上の注意

L シリーズの RVU では、ENDPOINT データを含む L シリーズの RVU 上で作成された測定の場合、INFO MEASUREMENT は GMS、ポート、またはファイバー指定を表示しません。ただし、GMS、ポート、およびファイバー指定は、H または J シリーズの RVU で作成された測定では表示されます。

INFO MEASUREMENT の例

- 次の例は、測定に対してカウンターデータレコードの抑制が指定された INFO MEASUREMENT レポートを表示します。Measure H02 以降の PVU では、この抑制機能が有効です。

```
+ INFO MEASUREMENT MEASXX
Add measurement $PERF.GREG.MEASXX -- Current Data File --
Data collected from system \YOSPRD, MEASURE release version H02.
From 16 Aug 2007, 13:52:32
Counter data records suppressed
-- Add Cpu 0
```

- 次の例は、ANSI SQL 名が含まれている INFO MEASUREMENT レポートを表示します。

```
+ INFO MEASUREMENT $GUEST.MEASURE.DEMO
Add measurement $GUEST.MEASURE.DEMO -- Current Data File --
Data collected from system \DEMO, MEASURE release version H01.
From 26 Feb 2003, 15:59:54
-- Add File 'TABLE CATALOG_12.SCHEMA_34.TABLE_56'
-- Add Discopen 'TABLE CATALOG_12.SCHEMA_34.TABLE_56'
-- Add Diskfile 'TABLE CATALOG_12.SCHEMA_34.TABLE_56'
```

- 次の例は、現在アクセス可能なすべてのデータファイルに関する情報を表示します。現在のファイルはアクティブなデータファイルであるため、INFO MEASUREMENT コマンドは、測定の停止時間やファイルの内容を表示できません。

```
+ INFO MEASUREMENT *
Add measurement $DATA.CAPTURE.CPUPROC - Current Data File --
Data collected from system \MEASURE, Measure release version G11.2
From 29 Aug 2003,  07:00:00
-- Add cpu *
-- Add process *
Add measurement $DATA.CAPTURE.NEWPH
Data collected from system \MEASURE, Measure release version G11.2
From 28 Aug 2003,  20:15:00,  To 28 Aug 2003,  20:30:00
Process                525 Entities          115500 Words
Processh               525 Entities          435317 Words
-- Add Processh $SYSTEM.SYS00.TSYSDP2 ($SYSTEM.SYS00.TSYSDP2)
-- Add Processh $SYSTEM.SYS00.MEASCOM ($SYSTEM.SYS00.MEASCOM)
```

- OSS 測定データファイルの場合、次の例はアクティブな測定を表示します。

```
+ INFO MEASUREMENT *
Add measurement $SPOOL.MEASDATA.MDATA
Data collected from system \DEV, MEASURE release version G09
From 17 Nov 2000,11:19:18
OSS journal segment attached
Cpu                      6 Entities            1992 Words
Process                909 Entities          156348 Words
-- Delete Process system-processes
-- Add CPU *
-- Add Process "/bin"
```

- 次の例は、OSS ファイルパス名の変換エラー（「*OSSPath*」で表示）を伴う INFO MEASUREMENT レポートを表示します。

```
+ INFO MEASUREMENT *
Add measurement $SPOOL.MEASDATA.MDATA -- Current Data File --
Data collected from system \DEV, MEASURE release version G09.
From 17 Nov 2000, 11:19:15,  To 17 Nov 2000, 11:19:18
Cpu                              6 Entities               1992 Words
Process                    909 Entities             156348 Words
-- Add Cpu *
-- Add Process "*OSSPath*"
```

- 次の例は、L シリーズ RVU に関する INFO MEASUREMENT レポートを表示します。測定に SERVERNET エンティティ（以下に ENDPOINT と表示）が存在する場合、GMS、ポート、およびファイバーの指定は、L シリーズ RVU では許可されていないので表示されません。

```
+ info measurement mdfxx
Add measurement $SYSTEM.KMAZAWA.MDFXX -- Current Data File --
Data collected from system \COGNAC1, MEASURE release version H07.
From 5 Sept 2014, 11:28:14,  To 5 Sept 2014, 11:29:49
Endpoint                 68 Entities            6256 Words
Fabric                   32 Entities            2688 Words
-- Add Cpu 3
-- Add Process 3, *
-- Add Endpoint * (IPC )
-- Add Endpoint 3 (RIPC)
-- Add Endpoint *, SCLIM001
-- Add Fabric * (CPU)
-- Add Fabric * (STORAGE, SCLIM000)
-- Add Fabric 0 (IP, NCLIM000, COMVQP10)
```

INFO PLOT

INFO PLOT コマンドでは、現在のプロット定義を表示します。

INFO PLOT の構文

```
INFO [ / OUT filename / ] [ PLOT ] counter [ (char) ]
     [ , counter [ (char) ] ] ...
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

INFO コマンドを実行した後、MEASCOM は現在の OUT ファイル（通常、端末）への出力の書き込みを再開します。

OUT オプションは、ログファイルの内容には影響しません。

PLOT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして PLOT を指定した場合はオプションです。

counter

PROCESSH と USERDEF を除くすべてのエンティティタイプで、*counter* はカウンター名です。すべてのカウンターを指定するには、アスタリスク（*）を使用します。

- PROCESSH エンティティタイプでは、*counter* はプロシージャー名です。プロット内のすべてのプロシージャーを表示するには、CODE-RANGE として *counter* を指定します。
- USERDEF エンティティタイプでは、*counter* はユーザー定義のカウンター名です。プロット内のすべてのユーザー定義のカウンターを表示するには、NAME として *counter* を指定します。

char

表示するプロットされたカウンターのプロット文字か、*counter* という名前のすべてのカウンターを指定するためのアスタリスク（*）です。省略した場合、デフォルトのアスタリスク（*）が使用されます。

関連コマンド

コマンド	機能
ADD PLOT	プロット定義にカウンターを追加します
DELETE PLOT	定義からカウンターを削除します
LIST PLOT	プロットを表示します

INFO PLOT の使用上の注意

INFO PLOT の出力は、カウンターをプロット定義に追加するために使用されるコマンドのリストです。/OUT *filename*/を使用して、指定したプロット定義を格納するコマンド（OBEY）ファイルを作成できます。

INFO PLOT の例

現在のプロット定義を表示するには、次のように入力します。

```
+ INFO PLOT *

Add measurement $SPOOL.QUOTAS.DATAGO
--A-- List Cpu 6, &
         To 20 Aug 1994, 14:29:08
       Add plot CPU-BUSY-TIME
--B-- List Cpu 7, &
         To 20 Aug 1994, 14:29:08
       Add plot CPU-BUSY-TIME
```

LIST entity-type

LIST *entity-type* コマンドは、現在のデータファイル（ADD MEASUREMENT コマンドで最後に指定されたファイル）からデータを読み取り、*entity-spec* パラメーターに含められたエンティティごとにレポートを表示します。

二重プロンプト（++）は、画面上のレポートに別のレポートが続くことを示します。次のレポートを表示するには Return キー（または他の任意のキー）を押します。後続のレポートを無視するには、Ctrl+Y キーを押すか二重プロンプトで BREAK と入力します。

- 前のコマンドラインで複数のコマンドを入力していた場合は、BREAK により現在のコマンドが中断され、次のコマンドが実行されます。
- 複数のコマンドを入力していない場合は、BREAK により現在のコマンドが中断され、MEASCOM プロンプトに戻ります。

LIST entity-type の構文

```
LIST [ / OUT filename / ] [ entity-type ] entity-spec
     [ , list-option ] ...
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

LIST コマンドを実行した後、MEASCOM は現在の OUT ファイル（通常、端末）への出力の書き込みを再開します。

OUT オプションは、ログファイルの内容には影響しません。

entity-type

次のいずれかです。

CLUSTER	DISCOPEN	LINE	PROCESS	SYSTEM
CONTROLLER	DISKFILE	NETLINE	PROCESSH	TERMINAL
CPU	ENDPOINT	OPDISK	SERVERNET	TMF
DEVICE	FABRIC	OSSCPU	SQLPROC	USERDEF
DISC	FILE	OSSNS	SQLSTMT	

entity-type キーワードは、ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして指定された場合はオプションです。

FILE および PROCESS エンティティの特別な留意事項については、**すべての PVU および RVU での LIST entity-type の使用上の注意**(83 ページ)を参照してください。

entity-spec

測定対象のエンティティを識別します。エンティティ指定はエンティティタイプごとに固有です。構文については、**エンティティおよびカウンター**(167 ページ)にある指定したエンティティタイプの説明を参照してください。

list-option

次のいずれかです。

BY *item-name* [(ASCENDING) | (DESCENDING)]

昇順または降順でレポートをソートします。

item-name は次のいずれかです。

```
{ counter  }
{ identification-item }
```

指定する値は次の通りです。

counter はカウンター名です。デフォルトでは、カウンター項目は降順でソートされます。

PROCESSH、DISC、CPU、および USERDEF のエンティティタイプに関する特別な留意事項は次のとおりです。

- PROCESSH では、*counter* はプロシージャー名です。
- Legacy 形式の DISC キャッシュカウンターでは、考慮するキャッシュブロックのサイズを指定するために *counter* の前に C0-、C1-、C2-、または C3- を付ける必要があります。ZMS 形式の DISC キャッシュカウンターでは、*counter* の前に C*n*- を付ける必要があります。ここで *n* は 0 から 7 の値です。
- CPU では、カウンターが指定されていない場合、CPU が番号順に表示されます。
- USERDEF では、有効なカウンター名は *name*、*value*、*type*、および *index* です。

identification-item は CPU-NUM や PROCESS-NAME などの識別項目です。指定したエンティティの有効などの識別項目でも使用できます。デフォルトでは、識別項目は昇順でソートされます。

CR-NAME-LEN { SHORT | LONG }

プロシージャー（code-range）名の表示される長さを制御します。デフォルトは SHORT です。

SHORT

プロシージャー（code-range）名を 32 文字で切り捨てます（必要な場合）。プロシージャーのマングル名は、表示前にデマングルされません。

LONG

プロシージャー（code-range）名を SHORT 形式で表示し、32 文字より長い場合は次の行に全体を表示します。プロシージャーのマングル名は、表示用にデマングルされません。

CR-NAME-FORM { STANDARD | DEMANGLED | BOTH }

プロシージャー（code-range）のデマングル名（該当する場合）を表示するかどうかを制御します。デフォルトは STANDARD です。

STANDARD

プロシージャー（code-range）名をコードファイル（マングル）または EDIT ファイル（デマングル）で指定した形式で表示します。

DEMANGLED

必要に応じて、表示前にプロシージャー（code-range）名をデマングルします。

BOTH

プロシージャー（code-range）名を、STANDARD と DEMANGLED の両方の形式で表示します。

CR-NAME-QUAL { UNQUALIFIED | QUALIFIED }

code-range 出力の従来の行の直後に続く行の Code-Range Name 列で、オブジェクトファイル名修飾子を付けて（該当する場合）、プロシージャー（code-range）名を表示するかどうかを制御します。デフォルトは UNQUALIFIED です。

UNQUALIFIED

修飾子のない Legacy 形式でプロシージャー名を表示します。

QUALIFIED

関連付けられたコードの Guardian または OSS オブジェクトファイル名とともにプロシージャー名を表示します。この機能は、さまざまなオブジェクトファイル内にある似た名前のプロシージャーを区別する場合に役立ちます。Guardian ファイル名では、関連付けられた CRVSN が名前の末尾に付加されています。

COUNT *record-count*

表示する選択したレコードの数（10 進数）を指定します。たとえば、FIRST が 24 で COUNT が 6 の場合、レコード 24 から 29 が表示されます。指定しない場合、デフォルトはすべてのレコードです。指定した場合、値は正の整数である必要があります。

DOTS { ON | OFF }

レポートの表示で、ラベルと数値の間に接続のドットを含めるかどうかを指定します。STYLE が ZMS の場合にのみ有効です。デフォルト設定は、OFF です。

ON

フォーマットされた数値にラベルを接続するドットを表示します。

OFF

フォーマットされた数値にラベルを接続するドットを表示しません。

FIRST *record-num*

開始レコード番号（10 進数）を指定します。指定しない場合、デフォルトは 1 です。指定した場合、値は正の整数である必要があります。

FOR *duration*

レポートウィンドウの期間を指定します。FOR と TO を同時に使用することはできません。省略した場合、測定が終了したときとエンティティの測定が終了したときのいずれか早い時点で、ウィンドウは終了します。

duration

次のいずれかの形式の時間間隔です。

n SECOND[S]
n MINUTE[S]
n HOUR[S]

指定する値は次の通りです。*n* は 1～9999 の範囲の整数です。

FORMAT { BRIEF | CSV | JSON | NORMAL | STRUCTURED | XML }

レポートの形式を設定します。デフォルトは NORMAL です。

BRIEF

エンティティタイプごとに最もよく使用されるカウンターを含む簡略レポートを表示します。

簡略形式のレポートに含まれるカウンターについて詳しくは、**エンティティおよびカウンター** (167 ページ)で、指定されたエンティティタイプの DDL レコードを参照してください。簡略形式のレポートに含まれるカウンターは、各 DDL レコードで**太字**で記されています。

CSV

現在のサブボリュームで、エンティティタイプに基づいた名前と CSV のサフィックスを持つ「C」ディスクファイル（タイプ 180）に、コマンドからの出力を CSV 形式で書き込みます。ファイルの最大サイズは 1TB です。ファイルがすでに存在している場合、MEASCOM はそのファイルを上書きします。ファイルが存在しない場合、MEASCOM が作成します。

JSON

現在のサブボリュームで、エンティティタイプに基づいた名前と JSN のサフィックスを持つ「C」ディスクファイル（タイプ 180）に、コマンドからの出力を JSON 形式で書き込みます。ファイルの最大サイズは 1TB です。ファイルがすでに存在している場合、MEASCOM はそのファイルを上書きします。ファイルが存在しない場合、MEASCOM が作成します。

NORMAL

すべてのカウンターを表示します。

STRUCTURED

Enform を使用した後の調査のために構造化ファイルにレポートを書き込みます。1 つのレコードは、単一のエンティティのすべてのカウンターから構成されます（DDL レコード形式については、**エンティティおよびカウンター**(167 ページ)にある指定したエンティティタイプの説明を参照してください）。

同じエンティティタイプのエンティティに関するレポートは同じファイルに書き込まれ、そのファイルはエンティティタイプに応じた名前が付けられます。ファイルがすでに存在する場合、MEASCOM がデータをそのファイルに追加します。ファイルが存在しない場合、MEASCOM が作成します。（FORMAT 属性を変更すると構造化ファイルは閉じられます）。

XML

現在のサブボリュームで、エンティティタイプに基づいた名前と XML のサフィックスを持つ「C」ディスクファイル（タイプ 180）に、コマンドからの出力を XML 形式で書き込みます。ファイルの最大サイズは 1TB です。ファイルがすでに存在している場合、MEASCOM はそのファイルを上書きします。ファイルが存在しない場合、MEASCOM が作成します。

FROM [[*start-date* ,] *start-time-of-day*]

レポートウィンドウの開始を指定します。FROM を省略すると、測定の開始時刻か、エンティティの測定の開始時刻のいずれか遅い方が使用されます。

start-date

次のいずれかの形式で表した、レポートウィンドウが開始する日付です。

```
{ [d]d mmm[ yyyy]}
{ mmm [d]d[ yyyy] }
```

指定する値は次の通りです。

dd は日であり、1～31 の範囲の数値です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT です。

yyyy は西暦です。有効な年は 1984～2047 です。

date を省略した場合、測定の開始日が使用されます。

start-time-of-day

次の形式で表した、レポートウィンドウが開始する時刻です。

```
hh:mm[:ss]
```

指定する値は次の通りです。

hh は時間です（0～23）。

mm は分です（0~59）。

ss は秒です（0~59）。

time-of-day を省略した場合、測定の開始時間が使用されます。

IF { *item-name operation value* }

レポートに含めるデータレコードを決定します。指定した条件を満たすレコードだけが含められます。

item-name

次のいずれかです。

```
{ counter             }
{ identification-item }
```

指定する値は次の通りです。

counter はカウンター名です。これらのエンティティタイプに関する特別な留意事項は次のとおりです。

- PROCESSH エンティティタイプでは、*counter* はプロシージャー名です。
- DISC エンティティタイプのキャッシュカウンターでは、考慮するキャッシュブロックのサイズを指定するために *counter* の前に C0-、C1-、C2-、または C3- を付ける必要があります。

identification-item は、INT または INT(32)タイプの数値エンティティ識別項目です。使用できる識別項目を確認するには、MEASDDLS ファイルか、**MEASCONFIGURE**(469 ページ)にある各エンティティ記述子の説明を参照してください。

文字列項目、固定長項目、USERDEF エンティティ、および FILE エンティティに関する特別な留意事項は次のとおりです。

- IF 句では、PROCESS-NAME などの文字列項目と、FROM-TIMESTAMP などの固定長項目を使用できません。この規則の 1 つの例外が USERDEF エンティティであり、このエンティティでは、カウンターが *name* に等しい場合は文字列を使用できません。次に例を示します。

  ```
  + LIST USERDEF *, IF NAME = MYCOUNTER
  ```

- IP-ADDR 値は FILE 測定のエンティティ記述子として使用できませんが、レポートを表示する選択オプションとしてこの値を使用できます。詳しくは、**IP-ADDR**(288 ページ)、**OSS の OSS AF_INET および AF_INET6 ストリームソケットのオープン**(307 ページ)、および **OSS の OSS AF_INET および AF_INET6 データグラムソケットのオープン**(308 ページ)を参照してください。

operation

次のいずれかです。

>（大きい）	=（等しい）
<（未満）	<>（等しくない）

value

0～2147482.999 の範囲の数値です。Measure G09 以降の PVU では、0～999999999999 の範囲です。*item-name* が時間カウンターである場合、*value* は秒単位で表す必要があります。たとえば、26 ミリ秒は**.026** として入力する必要があります。

LOADID *loadid*

このコマンドで生成されるレコードの *loadid* フィールドに配置される名前を指定します。LOADID オプションは、構造化レポートにのみ適用されます。

loadid

1～8 文字の長さの英数字の文字列です。文字列には、英字、数字、キャレット（^）、ハイフン（-）、およびアンダースコア（_）を含められます。最初の文字は、英字でなければなりません。

RATE { ON | OFF }

カウンター値を表示する方法を指定します。RATE 属性は、構造化ファイルには影響がありません。デフォルトは ON です。

ON

解釈されたカウンター値（1 秒あたりの数とビジー率）を表示します。

OFF

解釈されていないカウンター値を表示します。

RECORD-NUM { ON | OFF }

各レコードのレコード番号を表示するかどうかを指定します。デフォルト設定は、OFF です。

ON

各レコードのレコード番号を表示します。

OFF

各レコードのレコード番号を表示しません。

STYLE { LEGACY | ZMS }

ZMS 形式のインターフェイスを使用するか、Measure G11 より前の PVU と互換性のあるレガシーインターフェイスを使用して、表示または構造化データをフォーマットするかどうかを設定します。G シリーズの RVU ではデフォルトは LEGACY で、J シリーズ以降の RVU ではデフォルトは ZMS です。

LEGACY

表示および構造化レコードで Legacy 形式を使用します。

ZMS

表示および構造化レコードで ZMS 形式を使用します。

TO [*end-date* ,] *end-time-of-day*]

レポートウィンドウの終了を指定します。FOR と TO を同時に使用することはできません。省略した場合、測定が終了したときとエンティティの測定が終了したときのいずれか早い時点で、ウィンドウは終了します。

end-date

レポートウィンドウが終了する日付で、*start-date* と同じ形式です。

end-date を省略した場合、測定の終了日が使用されます。

end-time-of-day

レポートウィンドウの終了時間で、*start-time-of-day* と同じ形式です。

end-time-of-day を省略した場合、測定の終了時間が使用されます。

TOLERANCE { ON | OFF }

レポートに含まれている測定を特定する場合に適用される許容値を示します。

ON

（デフォルト）FROM および TO の範囲を、測定間隔の半分をプラスまたはマイナスして境界を定めた範囲として解釈します。たとえば、指定した FROM 時刻が 8:00 であり、測定間隔が 30 分の場合、実際の FROM 時刻は 7:45 に早まります。同様に、指定した TO 時刻が 10:00 の場合は、実際の TO 時刻は 10:15 にまで延長されます。

OFF

指定どおり正確に FROM と TO の範囲を適用します。

TOTALS { INCLUDE | ONLY | SUPPRESS }

TOTALS が表示されるかどうかを指定します。デフォルトは SUPPRESS です。

INCLUDE

プロセスごとの合計と総合計の両方が表示されることを示します。総合計が表示されるときに、一連の合計は大カッコ（[…]）で囲まれ、総計されているコードを実行していたプロセス数が個々の code-range の結果の前に置かれ、これも同様に大カッコで囲まれます（つまり[Totals across 3 processes]）。

ONLY

PROCESSH および USERDEF を除くすべてのエンティティについて、最終の TOTALS レポートのみを表示します。1 つのエンティティレポートのみを生成した場合は、TOTALS レポートは表示されません。

PROCESSH エンティティの場合は、集計した PROCESSH データのみが表示されます。

注記:

PROCESSH 測定が（ADD PROCESSH コマンドで）構成されたときにコードファイル指定が与えられたかどうかに応じて、コードファイルごとか、プロシージャーごとのどちらかに基づいて PROCESSH データを収集できます。したがって、表示された合計は、コードファイル全体を含む場合も、コードファイル内の一連のコード範囲それぞれを含む場合もあります。

SUPPRESS

エンティティレポートのみを表示します （レポートに 1 つのエンティティの測定のみが含まれている場合、TOTALS 属性が無視されます）。

PROCESSH の場合、SUPPRESS はプロセスごとのデータのみが表示される（集計データは表示されない）ことを示します。

SUPPRESS は USERDEF レポートには影響しません。

ZERO-REPORTS { INCLUDE | SUPPRESS }

すべてのゼロの値を含むレコードが表示されるかどうかを指定します。デフォルトは SUPPRESS です。

INCLUDE

すべてのカウンターの値がゼロの場合でも、エンティティレポートを表示します。

SUPPRESS

すべてのカウンターの値がゼロの場合、エンティティレポートを表示しません。

注記:

スナップショットカウンターは、特定の時点で取得された測定を返します。ゼロ以外のスナップショットカウンターの値は、必ずしも測定間隔中のアクティビティを示すわけではありません。スナップショットカウンターの値は通常、ZERO-REPORTS 句のレコードを評価するときに考慮されません。

ZERO-VALUES { INCLUDE | SUPPRESS }

0.005 未満の値をゼロまたは空白として表示するかどうかを指定します。デフォルトは SUPPRESS です。

INCLUDE

カウンターの値が 0.005 未満の場合にゼロを表示します。

SUPPRESS

カウンターの値が 0.005 未満の場合に空白を表示します。

ZERO-VALUES 属性は、構造化ファイルには影響がありません。

関連コマンド

コマンド	機能
LISTACTIVE	アクティブな測定を表示します
SET REPORT	レポート形式を指定します

すべての PVU および RVU での LIST entity-type の使用上の注意

- L16.05/J06.20 以降の RVU では、COUNT、FIRST、および RECORD-NUM を使用するときに、各外部レコードは、PROCESSH と USERDEF を除くすべてのエンティティの 1 つのレコードとして数えられます。PROCESSH と USERDEF では、各プロセスが 1 つのレコードとして数えられます。
- FILE および PROCESS のエンティティは、開いているファイルの名前を表示します。プロセスなど、ファイル名構文に準拠していない名前を持つリソースをファイルに含められます。これらのエンティティのレポートを表示するには、ファイル名ではなくリソースの CPU、PIN、およびファイル番号を指定します。例については、**LIST entity-type の例**(84 ページ)を参照してください。
- FROM、FOR、または TO 句を使用して、測定データの一部分だけを表示するレポートウィンドウを指定した場合、LIST コマンドはレポートを生成します。
  - 収集間隔なしでデータが収集された場合、指定したレポートウィンドウ内で開始および停止したエンティティのみが、レポートに含められます。
  - 収集間隔を使用してデータが収集された場合、指定するエンティティごとに、MEASCOM は、レポートウィンドウの開始に最も近い時点で書き込まれたレコードと、レポートウィンドウの終了に最も近い時点のレコードを読み取ります。MEASCOM は続いて、2 つのレコードの差異からレポートを作成します。両方のレコードは、レポートウィンドウ内に存在するか、ウィンドウの開始または終了時間の 1 つの収集間隔内に存在する必要があります。
- レポート出力ファイルが構造化ファイルである場合、格納されたレポート（BY *item-name*）を生成できません。REPORT オブジェクトの FORMAT オプションは、出力形式を指定します。
- リストオプション CR-NAME-LEN、CR-NAME-FORM、および CR-NAME-QUAL は、PROCESSH エンティティにのみ有効です。
- 処理可能な形式に対応する外部レコードを取得するには、**TEMPLATE-VERSION**(180 ページ)を参照してください。

L シリーズの PVU および RVU での LIST entity-type の使用上の注意

- L シリーズの RVU 上に作成された測定の場合、LIST コマンドは、SERVERNET エンティティの GMS、ポート、またはファイバー指定をサポートしません。これらの指定は、H および J シリーズの RVU 上で作成された測定では引き続きサポートされます。
- L シリーズの RVU 上に作成された測定の場合、Adpt（アダプター）フィールドには「CPUnn-MITE」ではなく「HCA-1」が表示され、SERVERNET エンティティでは GMS、ポート、およびファイバーのフィールドは表示されません。

H シリーズ以降の PVU および RVU での LIST entity-type の使用上の注意

- H06.25/J06.14 以降の RVU で、MEASFT SPLIT BY TIME コマンドを使用して分割したデータファイルのレポートを生成する場合は、LIST *entity-type* コマンドで（TIME 分割データファイルの期間内で）FROM および TO オプションを指定して、元のデータファイルに対して行われた Measure のレポートと、TIME 分割データファイルに対して行ったレポートとの忠実性を確保する必要があります。
- H06.23/J06.12 以降の RVU では、CLIM 接続ディスクに対する LIST DISC コマンドは、オプションの DISC 指定パラメーターとしてパーティション番号をサポートします。パーティション番号は、パーティション分割されたディスク内のパーティションに割り当てられている番号です。IF および BY 句でパーティション番号を指定する PARTITION の使用もサポートされます。

  さらに、CLIM 接続ディスクのパーティション番号が測定で指定されていると、そのパーティション番号が LIST DISC コマンドで表示されます。**LIST entity-type の例**(84 ページ)を参照してください。
- Measure H04/J02 以降の PVU は、LIST コマンドは、レポート出力ファイルが構造化ファイルである場合、測定データファイルのサイズに応じて形式 1 または形式 2 の構造化ファイルを作成します。
- Measure H03/J01 以降の PVU では、LIST コマンドは、CLIM を使用する SERVERNET、DEVICE、および DISC エンティティ指定を可能にします。構文と例については、**SERVERNET**(399 ページ)、**DEVICE**(214 ページ)、および **DISC**(226 ページ)エンティティを参照してください。
- Measure H01 以降の PVU では、LIST コマンドは、ANSI SQL 名を使用する DISCOPEN、DISKFILE、FILE、および SQLSTMT のエンティティ指定を可能にします。構文については、**DISCOPEN**(249 ペ

ージ)、**DISKFILE**(259 ページ)、**FILE**(284 ページ)、または **SQLSTMT**(421 ページ)エンティティを参照してください。

G シリーズの PVU および RVU での LIST entity-type の使用上の注意

- Measure G12 以降の PVU では、TNS コードファイルでのみコード領域識別子が表示されます（高速化 TNS コードファイルを含む）。
- G シリーズの RVU では、ZMS 形式のレポートモードの場合でも、IF 句は、ZMS 形式のレコードおよびレポートにのみ表示されるフィールドを参照できません。

LIST entity-type の例

- 次の LIST DISC コマンドの表示例では、$DISC1 という名前の CLIM 接続ディスクのパーティション番号（**太字**で強調表示）が示されます。

```
+ LIST DISC $DISC1
Disc $DISC1-P    (147 GB)          PID 3,304      Pool
Cfg   $DISC1
SAC   C1002531
SvNet X     LDEV      221   LUN 308    Partition  1
Format Version:  H06  Data Version:  H06  Subsystem Version:  2
Local System \MEASURE  From  25 Jan 2011, 11:44:50   For  11.1 Seconds
------------ Logical Requests
---------------------------------------------
Request-Qtime                            Requests                    0.18 /s
Request-ART                  0.04 ms     Swaps
DefReq-QTime                             DefReqs
Deferred-QTime                           Requests-Blocked
------------ Physical Operations
-----------------------------------------
Device-Qbusy-Time             0.19 %     Disc-Rate                   0.18 /s
Device-AST                   10.39 ms
...
```

- NORMAL レポート形式（NORMAL ではすべてのカウンターが表示されます）を使用して CPU レポートを表示するには、次のように入力します。

```
+ LIST CPU 5
Cpu  5 VLX          Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16        PCBs    491               Page Size  2048
Local System \BUYER From 22 Oct 1994, 12:20:47 For 118 Secs

Cpu-Busy-Time        21.46 % Swaps                 0.21
Cpu-Qtime             0.27 # Cpu-Qlen-Max           62 #
Mem-Qtime             0.01 # Mem-Qlen-Max           26 #
Dispatches           31.80   Intr-Busy-Time        0.49 %
Process-Ovhd          0.02 % Send-Busy-Time        0.12 %
Disc-IOs              2.00   Cache-Hits            4.39
Transactions                 Response-Time
Page-Requests                Page-Scans
Ending-Free-Mem       4092   Ending-UCME
Ending-UDS             500   Ending-SDS            1500
Ending-UCL             100   Ending-SCL            2000
```

- BRIEF 形式（BRIEF は Measure 定義のカウンターのサブセットが表示されます）を使用してすべての CPU レポートを表示するには、次のように入力します。

```
+ LIST CPU *
Cpu  5 VLX          Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16        PCBs    491               Page Size  2048
```

```
Local System \BUYER From 22 Oct 1994, 12:20:47 For 118 Secs

Cpu-Busy-Time        21.46% Swaps                0.21
Ending-Free-Mem       4092  Ending-UCME
++
Cpu  6 VLX           Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16         PCBs    100               Page Size  2048
Local System \BUYER From 22 Oct 1994, 12:20:47 For 118 Secs

Cpu-Busy-Time       18.42% Swaps                0.27
Ending-Free-Mem       4092 Ending-UCME             0
++
Cpu  7 VLX           Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16         PCBs    100               Page Size  2048
Local System \BUYER From 22 Oct 1994, 12:20:47 For 118 Secs

Cpu-Busy-Time           5% Swaps                 0.19
Ending-Free-Mem       4092 Ending-UCME             0
```

- SWAPS カウンターに従って昇順で同じ CPU を表示するには、次のように入力します。

```
+ LIST CPU *, BY SWAPS (ASCENDING)
Cpu  7 VLX           Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16         PCBs    100               Page Size  2048
Local System \BUYER From 22 Oct 1994, 12:20:47 For 118 Secs

Cpu-Busy-Time           5% Swaps                 0.19
Ending-Free-Mem       4092 Ending-UCME             0
++
Cpu  5 VLX           Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16         PCBs    491               Page Size  2048
Local System \BUYER From 22 Oct 1994, 12:20:47 For 118 Secs

Cpu-Busy-Time        21.46% Swaps                0.21
Ending-Free-Mem       4092  Ending-UCME            0
++
Cpu  6 VLX           Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16         PCBs    100               Page Size  2048
Local System \BUYER From 22 Oct 1994, 12:20:47 For 118 Secs

Cpu-Busy-Time        18.42% Swaps                0.27
Ending-Free-Mem       4092  Ending-UCME            0
++
```

- CPU-BUSY-TIME 値で 5 より大きい CPU レポートのみを表示するには、次のように入力します。

```
+ LIST CPU *, IF CPU-BUSY-TIME > 5
Cpu  6 VLX           Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16         PCBs    100               Page Size  2048
Local System \BUYER From 22 Oct 1994, 12:20:47 For 118 Secs

Cpu-Busy-Time        18.42% Swaps                0.27
Ending-Free-Mem       4092  Ending-UCME            0
++
Cpu  5 VLX           Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16         PCBs    491               Page Size  2048
Local System \BUYER From 22 Oct 1994, 12:20:47 For 118 Secs
```

```
Cpu-Busy-Time        21.46% Swaps                0.21
Ending-Free-Mem       4092  Ending-UCME             0
```

- CPU 1 の 2 つのレポート（測定ウィンドウ全体にわたる CPU アクティビティの平均を示したレポートと、より小さなレポートウィンドウでの CPU アクティビティの平均を示したレポート）を表示するには、次のように入力します。

```
+ LIST CPU 1
Cpu  1 VLX          Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16        PCBs    256               Page Size  2048
Local System \BUYER From 5 Nov 1994, 10:29:24 For 3.5 Hours

Cpu-Busy-Time        36.29% Swaps                0.42
Ending-Free-Mem       4092  Ending-UCME             0

+ LIST CPU 1, FROM 12:30, TO 12:45
Cpu  1 VLX          Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16        PCBs    256               Page Size  2048
Local System \BUYER From 5 Nov 1994, 12:30:24 For 15 Minutes

Cpu-Busy-Time        15.05% Swaps                0.05
Ending-Free-Mem       4092  Ending-UCME             0
```

- リモートシステムからの測定データファイルを調べている場合は、データファイル内のファイルを参照するときに完全な指定を使用する必要があります。たとえば、ローカルシステム（\BILLS という名前）に関する FILE レポートを表示するには、次のように入力します。

```
+ LIST FILE *
File Open $NEW01.MEAS.QCPU4A
Device Type  3
Opener Process: CPU  3  PIN   37    File Number   1
Local System \BILLS From 24 Oct 1994, 10:01:08 For 0.1 Seconds

Reads                       45.46
Writes
++
File Open \BILLS.03,037
Device Type  0
Opener Process: CPU  3  PIN   39    File Number   2
Local System \BILLS From 24 Oct 1994, 10:01:08 For 0.2 Seconds

Reads
Writes                      86.28
++
```

一方、\ACCT からのファイルを調べるために、前の例の最初のファイルを表示するには、次のように入力します。

```
+ LIST FILE \BILLS.$NEW01.MEAS.QCPU4A
```

次に、例の 2 番目のファイルを表示するには、ファイル名（MEASCOM で一覧表示）が別のシステムを指しているため、CPU、PIN、およびファイル番号を使用する必要があります。

```
+ LIST FILE \BILLS.$*.*.* (3,39,2)
```

- 収集した PROCESSH サンプルを BRIEF 形式で表示するには、次のように入力します。

```
+ RESET REPORT *
+ SET REPORT FORMAT BRIEF
+ LIST PROCESSH *, RATE OFF, IF CODE-RANGE > 0
Process  1,47                Pri 168        Device Name $SYSTEM
Program  $SYSTEM.SYS46.FUP
Userid   255,255   Creatorid  255,255   Ancestor 1,37 ($Z02M)
```

```
Local System \RAMBLER From  10 Sep 2004, 13:22:08   For  22.2 Seconds
Total samples =         69 #
Code-Map  UC.0          Samples          1 #      1.45 % of total
Code file $SYSTEM.SYS46.FUP:134451329318623
Code-Map  UC.1          Samples          2 #      2.90 % of total
Code file $SYSTEM.SYS46.FUP:134451329318623
Code-Map  NATIVE        Samples         65 #     94.20 % of total
Code file $SYSTEM.SYS46.OSIMAGE:134451333766514
Code-Map  NATIVE        Samples          1 #      1.45 % of total
Code file $SYSTEM.SYS46.OSIMAGE:134451333766514
```

- 収集した PROCESSH サンプルをデフォルトの形式（CR-NAME-LEN SHORT、CR-NAME-FORM STANDARD、および CR-NAME-QUAL UNQUALIFIED を使用）で表示するには、次のように入力します。

```
+ LIST PROCESSH 1,47,BY CODE-RANGE,IF CODE-RANGE > 0
 Process  1,47                 Pri 168        Device Name $SYSTEM
 Program  $SYSTEM.SYS46.FUP
 Userid   255,255   Creatorid  255,255   Ancestor 1,37 ($Z02M)
 Local System \RAMBLER From  10 Sep 2004, 13:22:08   For  22.2 Seconds
 Total samples =         69 #
 Code-Map  UC.0          Samples          1 #      1.45 % of total
 Code file $SYSTEM.SYS46.FUP:134451329318623
                                                      Percent      Percent
 Code-Range Name                Accel        TNS    of Code-Map    of Total
------------------------------ ----------- ----------- -----------
----------
FILE^HAS^FORMAT1                           1 #               100 %
1.45
 Code-Map  UC.1          Samples          2 #      2.90 % of total

 Code file $SYSTEM.SYS46.FUP:134451329318623
                                                      Percent      Percent
Code-Range Name                Accel        TNS    of Code-Map    of Total
------------------------------ ----------- ----------- -----------
----------
 START^LARGE^READ                          1 #                50 %
1.45
 READ^BLOCK                                1 #                50 %
1.45
 Code-Map  NATIVE        Samples         65 #      94.20 % of total

 Code file $SYSTEM.SYS46.OSIMAGE:134451333766514
                                                      Percent      Percent
Code-Range Name                       NATIVE       of Code-Map    of Total
 ------------------------------- ----------- ----------- -----------
 dbio_xsums_validate_                     31 #     47.69 %     44.93 %
 dbio_xsums_generate_                      6 #      9.23 %      8.70 %
 tser__transfer                            3 #      4.62 %      4.35 %
 WAIT                                      3 #      4.62 %      4.35 %
 tip_avtrelease_flags                      2 #      3.08 %      2.90 %
 tip_avtsetup_data                         2 #      3.08 %      2.90 %
 LOCKMEMORY                                2 #      3.08 %      2.90 %
 searchInitIndex__45NSKOrderedArr          1 #      1.54 %      1.45 %
```

- 収集した PROCESSH をデマングル名（CR-NAME-LEN LONG、CR-NAME-FORM DEMANGLED、および CR-NAME-QUAL UNQUALIFIED）で表示するには、次のように入力します。

```
+ SET REPORT CR-NAME-LEN LONG
+ SET REPORT CR-NAME-FORM DEMANGLED
+ SET REPORT CR-NAME-QUAL UNQUALIFIED
+ LIST PROCESSH 1,47,BY CODE-RANGE,IF CODE-RANGE > 0
 Process  1,47                 Pri 168        Device Name $SYSTEM
 Program  $SYSTEM.SYS46.FUP
 Userid   255,255   Creatorid  255,255   Ancestor 1,37 ($Z02M)
 Local System \RAMBLER From  10 Sep 2004, 13:22:08   For  22.2 Seconds

 Total samples =         69 #

 Code-Map  UC.0          Samples          1 #      1.45 % of total
 Code file $SYSTEM.SYS46.FUP:134451329318623

                                                      Percent      Percent
 Code-Range Name                Accel        TNS    of Code-Map    of Total
------------------------------ ----------- ----------- -----------
----------
 FILE^HAS^FORMAT1                         1 #               100 %      1.45
 FILE^HAS^FORMAT1
 Code-Map  UC.1          Samples          2 #      2.90 % of total
 Code file $SYSTEM.SYS46.FUP:134451329318623
                                                        Percent      Percent
  Code-Range Name               Accel        TNS    of Code-Map    of
Total
------------------------------ ----------- ----------- -----------
----------
 START^LARGE^READ                         1 #                50 %      1.45
 START^LARGE^READ
 READ^BLOCK                               1 #                50 %      1.45
 READ^BLOCK

 Code-Map  NATIVE        Samples         65 #     94.20 % of total
 Code file $SYSTEM.SYS46.OSIMAGE:134451333766514

                                             Percent      Percent
                                           Percent      Percent
     Code-Range Name               NATIVE    of Code-Map    of Total
------------------------------ ----------- ----------- -----------
dbio_xsums_validate_                    31 #     47.69 %     44.93 %
dbio_xsums_validate_()
dbio_xsums_generate_                     6 #      9.23 %      8.70 %
dbio_xsums_generate_()
tser__transfer                           3 #      4.62 %      4.35 %
tser__transfer()
WAIT                                     3 #      4.62 %      4.35 %
WAIT()
tip_avtrelease_flags                     2 #      3.08 %      2.90 %
tip_avtrelease_flags()
tip_avtsetup_data                        2 #      3.08 %      2.90 %
tip_avtsetup_data()
LOCKMEMORY                               2 #      3.08 %      2.90 %
LOCKMEMORY()
searchInitIndex__45NSKOrderedArr         1 #      1.54 %      1.45 %
```

```
NSKOrderedArrayOf<NSKM_MOPSTEntry
, unsigned long>::searchInitIndex(unsigned long
, unsigned int) const
```

コード範囲名 `searchInitIndex__45NSKOrderedArr` は `NSKOrderedArrayOf<NSKM_MOPSTEntry, unsigned long>::searchInitIndex(unsigned long, unsigned int) const` になり、どちらもプログラマになじみのあるものです。

- L シリーズの RVU における次の LIST ENDPOINT コマンドの表示例では、IPC ノードクラスが示されます。

```
9+ list endpoint * (ipc)
SAC   IPC-CPU00-TO-CPU03
Adpt  HCA-1
CPU   0    Node Class  IPC                    Other CPU  1    Type 0
Format Version:  H07  Data Version:  H07  Subsystem Version:  1
Local System \COGNAC1  From  5 Sep 2014, 12:20:36  For  99.1 Seconds
------------------------------------------------------------------------
-
Requests                     4.12 /s   Total-IO-Bytes             1,643 /s
IO-Qtime                   882.00 AQL   IO-Qbusy-Time              100.00 %
--------- Data into this CPU --------  --------- Data out from this CPU
----
Read-Requests                2.06 /s   Write-Requests               2.06 /s
Read-Bytes                  72.39 /s   Write-Bytes                 1,336 /s
Read-Cbytes                 29.14 /s   Write-Cbytes               205.08 /s
Read-Qtime                             Write-Qtime                882.00
AQL
Read-Qbusy-Time                        Write-Qbusy-Time           100.00 %
Read-Crequests                         Write-Crequests
------------------------------------------------------------------------
-
Server-Qtime
Retries                                Acks
X-Defrd-Busy-Time                      Y-Defrd-Busy-Time
```

- L シリーズの RVU における次の LIST ENDPOINT コマンドの表示例では、現在の CPU と CLIM 間のアクティビティが示されます。

```
+ list endpoint * (clms)
SAC   SCLIM000
Adpt  SCLIM000
CPU   0    Node Class  CLMS
Format Version:  H07  Data Version:  H07  Subsystem Version:  1
Local System \COGNAC1  From  5 Sep 2014,  12:20:36  For  99.1 Seconds
------------------------------------------------------------------------
-
Requests                     0.35 /s   Total-IO-Bytes              40.40 /s
IO-Qtime                               IO-Qbusy-Time
--------- Data into this CPU --------  --------- Data out from this CPU
----
Read-Requests                          Write-Requests
Read-Bytes                             Write-Bytes
Read-Cbytes                 25.31 /s   Write-Cbytes                15.10 /s
Read-Qtime                             Write-Qtime
Read-Qbusy-Time                        Write-Qbusy-Time
Read-Crequests               0.13 /s   Write-Crequests              0.22 /s
------------------------------------------------------------------------
-
```

```
Server-Qtime
Retries                                 Acks
X-Defrd-Busy-Time                       Y-Defrd-Busy-Time
```

- L シリーズの RVU における次の LIST FABRIC コマンドの表示例では、2 つの CPU 間における特定の仮想チャネル上の I/O アクティビティが示されます。

```
+ list fabric * (cpu)
Virtual Channel IPC_HREP
CPU   0    Type  CPU     Target CPU  1    Target System \BLIBO
Format Version:  H07  Data Version:  H07  Subsystem Version:  1
Local System \BLIBO   From  24 Sep 2013,  3:57:00   For  14.1 Seconds
------------------------------------------------------------------------
-
X-Send-Qlen-Start                       X-Send-Qlen-End
Y-Send-Qlen-Start                       Y-Send-Qlen-End
--------- Data into this CPU --------  --------- Data out from this CPU
----
X-Recvs                       1.98 /s   X-Sends
X-Recv-Bytes                321.54 /s   X-Send-Bytes
Y-Recvs                                 Y-Sends                       1.91 /s
Y-Recv-Bytes                289.71 /s   Y-Send-Bytes                532.24 /s
```

- L シリーズの RVU における次の LIST FABRIC コマンドの表示例では、CPU とストレージ CLIM 間のアクティビティが示されます。

```
+ list fabric 0 (storage)
Virtual Channel MIME                    Endpoint SCLIM000
CPU   0    Type  Storage CLIM
Format Version:  H07  Data Version:  H07  Subsystem Version:  1
Local System \BLIBO   From  17 Sep 2013,  6:42:17   For  13.2 Seconds
------------------------------------------------------------------------
-
X-Send-Qlen-Start                       X-Send-Qlen-End
Y-Send-Qlen-Start                       Y-Send-Qlen-End
--------- Data into this CPU --------  --------- Data out from this CPU
----
X-Recvs                       0.08 /s   X-Sends                       0.30 /s
X-Recv-Bytes                  0.61 /s   X-Send-Bytes                 18.17 /s
Y-Recvs                                 Y-Sends
Y-Recv-Bytes                            Y-Send-Bytes
```

- L シリーズの RVU における次の LIST CPU コマンドの表示例では、CPU と IP CLIM 間のアクティビティが示されます。

```
+ list fabric 0 (ip)
Virtual Channel COMVQP00                Endpoint NCLIM001
CPU   0    Type  IP CLIM
Format Version:  H07  Data Version:  H07  Subsystem Version:  1
Local System \BLIBO   From  17 Sep 2013,  6:42:17   For  13.2 Seconds
------------------------------------------------------------------------
-
X-Send-Qlen-Start                       X-Send-Qlen-End
Y-Send-Qlen-Start                 1 #   Y-Send-Qlen-End                   1 #
--------- Data into this CPU --------  --------- Data out from this CPU
----
X-Recvs                                 X-Sends                       0.45 /s
X-Recv-Bytes                            X-Send-Bytes                 14.54 /s
```

```
Y-Recvs                        0.23 /s   Y-Sends
Y-Recv-Bytes                  14.54 /s   Y-Send-Bytes
```

- L シリーズの RVU における次の LIST CPU コマンドの表示例では、CPU 0 の CPU アクティビティが示されます。

```
+ list cpu 0
Cpu  0 NSX-B          Init Lock Pgs     19          Mem Pages       2088448
Memory MB      32632  PCBs           20100          Pg Size    16384  Bytes
IPUs    4
Format Version:  H07  Data Version:  H07  Subsystem Version:  3
Local System \COGNAC1 From  5 Sep 2014,  12:53:44   For  128 Seconds
------------ Processor
-----------------------------------------------------
Cpu-Busy-Time                50.51 %    Dispatches                  3,015 /s
Cpu-Qtime                     2.03 AQL  Intr-Busy-Time                0.08 %
Comp-Traps                              Process-Ovhd
------------ Memory
-----------------------------------------------------------
Starting-Free-Mem       1,959,430 #    Ending-Free-Mem         1,959,473 #
Swaps                         0.52 /s   Page-Requests                0.89 /s
MM-Page-Scans                22.60 /s   Page-Scans                  22.60 /s
Unsp-Pages-Qtime              5.29 AQL  Ending-UCME
Unsp-Pages-Start                        Unsp-Pages-End                 8 #
Starting-SCL               113,260 #    Ending-SCL                113,258 #
Starting-UCL                25,561 #    Ending-UCL                 25,504 #
Starting-Free-CIDs          63,706 #    Ending-Free-CIDs           63,674 #
Locked-Pages-Start         106,226 #    Locked-Pages-End          106,197 #
------------ Device I/O
-------------------------------------------------
Disc-IOs                      1.40 /s   Cache-Hits                   2.46 /s
Transactions                            Response-Time
------------ Messaging
-------------------------------------------------
Link-Prepush-Msgs             2.77 /s   Readlinkcache-All            3.49 /s
Link-Readlink-Msgs                      Readlinkcache-Ctrl           0.11 /s
Link-Large-Msgs                         Readlinkcache-None
                                        Replyctrlcache-Msgs             3 /s
++
------------ IPUs
-----------------------------------------------------------
IPU 0
Ipu-Busy-Time                 1.24 %    Ipu-Dispatches              1,586 /s
Ipu-Qtime                     0.02 AQL
IPU 1
Ipu-Busy-Time                 0.79 %    Ipu-Dispatches              597.12 /s
Ipu-Qtime                     0.01 AQL
IPU 2
Ipu-Busy-Time               100.00 %    Ipu-Dispatches              161.70 /s
Ipu-Qtime                        1 AQL
IPU 3
Ipu-Busy-Time               100.00 %    Ipu-Dispatches              670.19 /s
Ipu-Qtime                     1.00 AQL
++
------------ Endpoint Transfers Initiated by this Local Processor
-------------------
to/fm     READS INTO CPU     READ BYTES    WRITES FROM CPU    WRITE BYTES
CPU-0            5.43 /s        7,959 /s          8.33 /s        44,132 /s
```

```
CPU-1
CPU-2
CPU-3
CPU-4
CPU-5
CPU-6
CPU-7
CPU-8
CPU-9
CPU-10
CPU-11
CPU-12
CPU-13
CPU-14
CPU-15
See TOTAL for sum over all initiating CPUs: total data into & from each CPU
```

- L16.05 以前の L シリーズの RVU における次の LIST PROCESS コマンドの表示例では、割り込みカウンターがゼロより大きいプロセスが示されます。Interrupts は x-interrupts と y-interrupts の合計で、太字で表示されます。Interrupts カウンター自体は表示されません。

```
+ list process *,if interrupts > 0
Process  0,290                Pri 255        Device Name $SYSTEM
Program  $SYSTEM.SYS11.TSSTOIP (Native)      System Process: IB Storage IP
(17)
Userid   255,255   Creatorid  0,0       Ancestor Unknown
Format Version:  H07  Data Version:  H07  Subsystem Version:  6
Local System \COGNAC1   From  5 Sep 2014,  12:53:44   For  128 Seconds
------------ Processor
-----------------------------------------------------
Cpu-Busy-Time                             Dispatches                   0.83 /s
Ready-Time                                Comp-Traps
Launch-Wait-Time                          Launches
Vsems                                     Ipu-Switches
Ipu-Num                             1 #
Initial-Priority-Start            255 #   Initial-Priority-End          255 #
Current-Priority-Start            255 #   Current-Priority-End          255 #
X-Interrupts                     0.42 /s  Y-Interrupts                 0.41 /s
------------ Memory Management
---------------------------------------------
Mem-Qtime                                 Page-Faults
Pres-Pages-Qtime                11.00 AQL Abs-Segs-Qtime
Pres-Pages-Start                   11 #   Abs-Segs-Start
Pres-Pages-End                     11 #   Abs-Segs-End
Alloc-Seg-Calls                           Ext-Segs-Qtime
Locked-Pages-Start                 11 #   Locked-Pages-End               11 #
------------ Messaging
-----------------------------------------------------
Messages-Sent                             Messages-Received
Msgs-Sent-Qtime                           Recv-Qtime
Sent-Bytes                                Received-Bytes
Sent-Cbytes                               Received-Cbytes
Returned-Bytes                            Reply-Bytes
Returned-Cbytes                           Reply-Cbytes
MQCs-Inuse-Qtime                          MQC-Allocations
Checkpoints                               MQC-Alloc-Failures
------------ File
-----------------------------------------------------------
```

```
File-Open-Calls                         Info-Calls
Open-Close-Wait-Time
------------ Transactions
-------------------------------------------------
Begin-Trans                             Abort-Trans
------------ Open System Services
-----------------------------------------
OSS-TTY-Wait-Time                       OSS-Dev-Null-Ops
OSS-TTY-Reads                           OSS-TTY-Writes
OSS-TTY-Read-Bytes                      OSS-TTY-Write-Bytes
OSSNS-Wait-Time                         OSSNS-Requests
OSSNS-Message-Bytes                     OSSNS-Redirects
OSSNS-DD-Calls


Process  0,273                Pri 255        Device Name $SYSTEM
Program  $SYSTEM.SYS11.NTIMEIP (Native)      System Process: Timer IP (6)
Userid   255,255   Creatorid  0,0       Ancestor Unknown
Format Version:  H07  Data Version:  H07  Subsystem Version:  6
Local System \COGNAC1   From  5 Sep 2014, 12:53:44   For  128 Seconds
------------ Processor
-----------------------------------------------------
Cpu-Busy-Time                0.07 %    Dispatches                269.95 /s
Ready-Time                   0.07 %    Comp-Traps
Launch-Wait-Time                       Launches
Vsems                                  Ipu-Switches
Ipu-Num                         3 #
Initial-Priority-Start        255 #    Initial-Priority-End          255 #
Current-Priority-Start        255 #    Current-Priority-End          255 #
X-Interrupts                269.80 /s
------------ Memory Management
---------------------------------------------
Mem-Qtime                              Page-Faults
Pres-Pages-Qtime            11.00 AQL  Abs-Segs-Qtime               2.00
AQL
Pres-Pages-Start               11 #    Abs-Segs-Start                 2 #
Pres-Pages-End                 11 #    Abs-Segs-End                   2 #
Alloc-Seg-Calls                        Ext-Segs-Qtime
Locked-Pages-Start             11 #    Locked-Pages-End              11 #
------------ Messaging
-----------------------------------------------------
Messages-Sent                          Messages-Received
Msgs-Sent-Qtime                        Recv-Qtime
Sent-Bytes                             Received-Bytes
Sent-Cbytes                            Received-Cbytes
Returned-Bytes                         Reply-Bytes
Returned-Cbytes                        Reply-Cbytes
MQCs-Inuse-Qtime                       MQC-Allocations
Checkpoints                            MQC-Alloc-Failures
------------ File
-----------------------------------------------------------
File-Open-Calls                        Info-Calls
Open-Close-Wait-Time
------------ Transactions
-------------------------------------------------
Begin-Trans                            Abort-Trans
------------ Open System Services
```

```
-----------------------------------------
OSS-TTY-Wait-Time                       OSS-Dev-Null-Ops
OSS-TTY-Reads                           OSS-TTY-Writes
OSS-TTY-Read-Bytes                      OSS-TTY-Write-Bytes
OSSNS-Wait-Time                         OSSNS-Requests
OSSNS-Message-Bytes                     OSSNS-Redirects
OSSNS-DD-Calls
```

LIST EXTNAMES

Measure G11 以降の PVU で、LIST EXTNAMES コマンドは、次のマッピングに使用するレコードのキーシーケンスファイルを作成したり付加します。

- Measure OSS PATHID 値を OSS パス名および Guardian ファイル名に対応させるマッピング
- MID を ANSI SQL 名および Guardian ファイル名に対応させるマッピング

ファイル EXTNAMES は EXTNAME 名レコードを含み、現在のデフォルトのサブボリュームにあると想定されます。

このコマンドには、オプションも修飾子もありません。このコマンドは REPORT FORMAT STRUCTURED オプションのコンテキスト内でのみ実行できます。

LIST EXTNAMES の構文

```
LIST EXTNAMES
```

EXTNAMES ファイルの DDL レコード

MEASDDLS で定義されている EXTNAMES の DDL レコードは以下のとおりです。

```
RECORD extname. FILE is "extnames" KEY-SEQUENCED.
 02 MID.
    03 PATHID              type character 24.
    03 CRVSN               type character 6.
 02 FILE-SYSTEM-NAME       type character 8.
 02 FILE-NAME.
    03 VOLUME              type character 8.
    03 SUBVOL              type character 8.
    03 FILENAME            type character 8.
 02 NAME-TYPE
type                                                                    .
 02 FULL-NAME-LEN          type binary 16 unsigned.
 02 FULL-NAME              type character 1100.
end
```

DDL レコード記述フィールド

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

MID

OSS ファイルまたは SQL/MX ファイルの Measure ID。OSS ファイルと SQL/MX ファイルでは、MID を EXTNAMES ファイルに対するキーとして使用して、対応する Guardian ファイル名と外部形式の OSS パス名または ANSI SQL 名を取得することができます。

MID には、次の 2 つのサブフィールドがあります。

PATHID	OSS ファイルまたは SQL/MX ファイルの内部形式表現。その他のファイルでは、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルまたは SQL/MX ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。その他のファイルでは、このフィールドにはゼロが含まれます。

FILE-SYSTEM-NAME

データファイル内で測定されたノードとは別のノードにファイルが配置されている場合、*file-name* で記述されるファイルの EXPAND システム名。それ以外の場合、このフィールドにはスペースが入ります。

FILE-NAME

OSS ファイルまたは SQL/MX ファイル用の Guardian ローカル内部形式のファイル名です。ファイルの *gname* とも呼ばれます。

NAME-TYPE

次のいずれかの名前のタイプです。0 - 使用しない、1 - OSS、2 - ANSI SQL。

FULL-NAME-LEN

FULL-NAME フィールドに格納された外部形式の OSS パス名または ANSI SQL 名の長さ（バイト数）。Measure G11 以降の PVU で最大値は 1100 です。

FULL-NAME

左詰めにした外部形式の OSS パス名または ANSI SQL 名。このフィールドは 1100 文字として定義されますが、実際の名前の長さは FULL-NAME-LEN で報告されます。残りの文字は定義されません。

LIST EXTNAMES の例

EXTNAMES 構造化出力ファイルを作成するには、次のように入力します。

```
+ SET REPORT FORMAT STRUCTURED
+ LIST EXTNAMES
```

LIST OSSNAMES

Measure G09 以降の PVU では、LIST OSSNAMES コマンドは、Guardian ファイル名または Measure MID 値を作成し、OSS ファイルパス名同等表現に付加します。キーシーケンスファイルには OSSNAMES という名前が付けられ、現在のデフォルトのサブボリュームに存在すると想定されます。

LIST OSSNAMES コマンドにはオプションも修飾子もありません。このコマンドは REPORT FORMAT STRUCTURED コマンドの後でのみ実行できます。

Measure G11 以降の PVU では、LIST OSSNAMES は引き続き使用できますが、OSS ファイルパス名と ANSI SQL 名の両方をサポートする **LIST EXTNAMES**(94 ページ)に置き換えられています。

LIST OSSNAMES の構文

```
LIST OSSNAMES
```

OSSNAMES ファイルの DDL レコード

MEASDDLS で定義されている OSSNAMES の DDL レコードは以下のとおりです。

```
RECORD OSSNAMES.  FILE is "OSSNAMES" KEY-SEQUENCED.
 02 MID.
    03 PATHID                  type character 24.
    03 CRVSN                   type character 6.
```

```
 02 file-system-name           type character 8.
 02 file-name.
    03 volume                  type character 8.
    03 subvol                  type character 8.
    03 filename                type character 8.
 02 pathname-len               type binary 16 unsigned.
 02 pathname                   type character 1024.
*    key is MID duplicates not allowed
end
```

DDL レコード記述フィールド

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

MID

OSS ファイルまたは SQL/MX ファイルの Measure ID。OSS ファイルでは、MID を OSSNAMES ファイルに対するキーとして使用して、対応する Guardian ファイル名と外部形式の OSS パス名を取得することができます。

MID には、次の 2 つのサブフィールドがあります。

PATHID	OSS ファイルまたは SQL/MX ファイルの内部形式表現。その他のファイルでは、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルまたは SQL/MX ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。その他のファイルでは、このフィールドにはゼロが含まれます。

FILE-SYSTEM-NAME

データファイル内で測定されたノードとは別のノードにファイルが配置されている場合、*file-name* で記述されるファイルの EXPAND システム名。それ以外の場合、このフィールドにはスペースが入ります。

FILE-NAME

OSS ファイル用の Guardian ローカル内部形式のファイル名です。ファイルの *gname* とも呼ばれます。

PATHNAME-LEN

PATHNAME フィールドに格納された外部形式の OSS パス名の長さ（バイト数）。Measure G11 以降の PVU では、最大値は 1024 です。

PATHNAME

左詰めにした外部形式の OSS パス名。このフィールドは 1024 文字として定義されますが、実際の名前の長さは PATHNAME-LEN で報告されます。残りの文字は定義されません。

LIST OSSNAMES の使用上の注意

Measure は、次の警告またはエラーメッセージを表示することがあります。

```
3114 OSS journal segment is required in this context and is not available
for access.
```

LIST OSSNAMES の例

OSSNAMES 構造化出力ファイルを作成するには、次のように入力します。

```
+ SET REPORT FORMAT STRUCTURED
+ LIST OSSNAMES
```

LIST PLOT

LIST PLOT コマンドは、1 つまたは複数の ADD PLOT コマンドによる指定に従って、現在指定されているプロットを表示します。

Measure G11 以降の PVU では、ANSI SQL 名がプロットに表示されます。

LIST PLOT の構文

```
LIST [ / OUT filename / ] [ PLOT                                    ]
                          [ , FOR duration                          ]
                          [ , FROM [ start-date, ] start-time-of-day ]
                          [ , INTERVAL interval                     ]
                          [ , SCALE-FROM number                     ]
                          [ , SCALE-TO number                       ]
                          [ , TIME-BASE { ON | OFF }                ]
                          [ , TO [ end-date , ] end-time-of-day     ]
                          [ , VERT-BASE { ON | OFF }                ]
                          [ , WIDE-ITEM { ON | OFF }                ]
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力をその末尾に追加します。

LIST PLOT コマンドを実行した後、MEASCOM はその出力を現在の OUT ファイル（通常、端末）にリダイレクトします。このオプションは、ログファイルの内容には影響しません。

PLOT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして PLOT を指定した場合はオプションです。

FOR *duration*

プロットウィンドウの期間を指定します。FOR と TO を同時に使用することはできません。省略した場合、測定とエンティティの測定のどちらかが先に終了した時点で、ウィンドウは終了します。

duration

次のいずれかの形式の時間間隔です。

```
n SECOND[S]
n MINUTE[S]
n HOUR[S]
```

指定する値は次の通りです。*n* は 1～9999 の範囲の整数です。

FROM [*start-date* ,] *start-time-of-day*]

プロットウィンドウの開始を指定します。FROM を省略すると、測定とエンティティ測定の開始時刻のいずれか遅い方が使用されます。

start-date

次のいずれかの形式で表した、プロットウィンドウが開始する日付です。

```
{ [d]d mmm[ yyyy] }
{ mmm [d]d[ yyyy] }
```

指定する値は次の通りです。

dd は日であり、1～31 の範囲の数字です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT です。

yyyy は西暦です。有効な年は 1984～2047 です。

start-date を省略すると、測定とエンティティの測定の開始日のいずれか遅い方が使用されます。

start-time-of-day

次の形式で表した、プロットウィンドウが開始する時刻です。

```
hh:mm[:ss]
```

指定する値は次の通りです。

hh は時間です（0～23）。

mm は分です（0~59）。

ss は秒です（0~59）。

start-time-of-day を省略した場合、測定の開始時刻が使用されます。

INTERVAL *interval*

プロットでの時間行の時間間隔を指定します。

interval

次のいずれかの形式の時間間隔です。

```
n SECOND[S]
n MINUTE[S]
n HOUR[S]
```

指定する値は次の通りです。*n* は 1～9999 の範囲の整数であり、データの収集時に指定された収集間隔の倍数です。

INTERVAL を省略すると、MEASCOM はデータの収集時に指定された収集間隔を使用します。

SCALE-FROM *number*

データ値軸の下限を設定します。

number

0～999999999999.999 の範囲の数字です。デフォルトは 0 です。

SCALE-TO *number*

データ値軸の上限を設定します。

number

0～999999999999.999 の範囲の数字です。デフォルトは 100 です。

TIME-BASE { ON | OFF }

カウンター値の棒グラフまたは 2 軸プロットを表示します。データを収集したときに時間間隔を指定した場合、デフォルトは TIME-BASE ON です。指定しなかった場合、デフォルトは TIME-BASE OFF です。

ON

カウンター値のプロットを時系列で表示します。

OFF

カウンター値の平均を棒グラフで表示します。

TO [*end-date* ,] *end-time-of-day*]

プロットウィンドウの終了を指定します。FOR と TO を同時に使用することはできません。省略した場合、測定とエンティティの測定のどちらかが先に終了した時点で、ウィンドウは終了します。

end-date

プロットに使用するデータの終了日で、*start-date* と同じ形式です。

end-time-of-day

プロットに使用するデータの終了時刻で、*start-time-of-day* と同じ形式です。

VERT-BASE { ON | OFF }

時間軸の表示を制御します。デフォルトは ON です。

ON

2 軸プロットでは、時間を縦軸に表示します。棒グラフでは、（縦軸を基準にして）棒を横に表示します。

OFF

2 軸プロットでは、時間を横軸に表示します。棒グラフでは、（横軸を基準にして）棒を縦に表示します。

WIDE-ITEM { ON | OFF }

プロットの密度を設定します。デフォルト設定は、OFF です。

ON

2 軸プロットでは、時間軸と最も小さいカウンター値の間の領域を埋めます。棒グラフでは、グラフ上のカウンター値の数に応じて、2～6 文字の幅の棒を表示します。

OFF

2 軸プロットでは、各時間間隔でカウンター値ごとに 1 つのプロット文字を表示します。棒グラフでは、1 文字の幅の棒を表示します。

関連コマンド

コマンド	機能
ADD PLOT	現在のプロット記述にカウンターを追加します
DELETE PLOT	現在のプロット記述からカウンターを削除します
INFO PLOT	現在のプロット記述を表示します

表は続く

RESET PLOT	プロットオプションをデフォルトにリセットします
SET PLOT	プロット形式を指定します
SHOW PLOT	プロット属性の設定を表示します

H シリーズ以降の RVU での LIST PLOT の使用上の注意

- H06.25/J06.14 以降の RVU で、MEASFT SPLIT BY TIME コマンドを使用して分割したデータファイルのプロットを生成する場合は、LIST PLOT コマンドで（TIME 分割データファイルの期間内で）FROM および TO オプションを指定して、元のデータファイルに対して行われた Measure のプロットと、TIME 分割データファイルに対して行ったプロットとの忠実性を確保する必要があります。
- H06.23/J06.12 以降の RVU では、パーティション分割されたディスクの場合、パーティション番号が LIST PLOT コマンドの凡例に表示されます。**LIST PLOT の例**(100 ページ)を参照してください。
- H03/J01 以降の PVU では、CLIM DISC データは STYLE LEGACY でプロットできません。

LIST PLOT の例

- 次の LIST PLOT コマンドの例では、パーティション分割された CLIM 接続ディスクの凡例でパーティション番号（**太字**で強調表示）を表示します。

```
+ ADD PLOT REQUESTS
+ LIST PLOT,SCALE-TO 330
         0:::::+:::66.0:::::+::::132:::::+::::198:::::+::::
264:::::+::::330
       A
*************************************************************|     -
       B
************|                                                      -
       C
**********|                                                       -
       D
******|                                                          -
       E
*****|                                                          -
         0:::::+:::66.0:::::+::::132:::::+::::198:::::+::::
264:::::+::::330

      Min value = 24.567              Max value = 307.541

   A: REQUESTS                        Disc $DISC1-P (3, X, C1002531, 309,
1)
   B: REQUESTS                        Disc $DISC2-P (2, X, C1002531, 311)
   C: REQUESTS                        Disc $DISC3-P (3, X, C1002531, 309,
2)
   D: REQUESTS                        Disc $SYSTEM-P (0, X, C1002531, 101)
   E: REQUESTS                        Disc $DISC4-P (1, X, C1002531, 107)
```

- 4 秒の収集間隔を使用して測定データが収集されたと想定して、デフォルトの形式でプロットを表示するには、次のように入力します。

```
+ LIST CPU *
      .
      .     (CPU レポートがここに表示されます)
      .
+ ADD PLOT CPU-BUSY-TIME
+ LIST PLOT
```

```
         0:::::+:::20.0:::::+:::40.0:::::+:::60.0 ...
14:28:19 -                  B          A
14:28:23 - B       A
14:28:27 - B      A
14:28:31 -B                       A
14:28:35 - B        A
14:28:39 -B   A
14:28:43 -B              A
14:28:47 - B               A
14:28:51 -B   A
14:28:55 -B   A
14:28:59 -B   A
14:29:03 -B   A
14:29:07 -          B                          A
         0:::::+:::20.0:::::+:::40.0:::::+:::60.0 ...

      Min. value =   2.000               Max value =  88.000

      A: CPU-BUSY-TIME                 Cpu 6
      B: CPU-BUSY-TIME                 Cpu 7
```

- 収集間隔を使用せずに測定データが収集されたと想定して、デフォルトの形式でプロットを表示するには、次のように入力します。

```
+ LIST CPU *
      .
      .    (CPU レポートがここに表示されます)
      .
+ ADD PLOT CPU-BUSY-TIME
+ LIST PLOT

        0:::::+:::20.0:::::+:::40.0:::::+:::60.0 ...
      A ************|
      B ***|
        0:::::+:::20.0:::::+:::40.0:::::+:::60.0 ...

      Min. value =   2.000               Max value =  88.000

      A: CPU-BUSY-TIME                 Cpu 6
      B: CPU-BUSY-TIME                 Cpu 7
```

- 次の例は、測定された高速化されたコードと TNS コードのサンプル数を示す PROCESSH プロットを表示します。アスタリスク（*）は、高速化されたコードを表し、プラス記号（+）は TNS コードを表します。プロットには、個別にプロットされた TNS コードの結果も示されます。

```
+ LIST PROCESSH $SYSTEM.SYS02.MEASCOM (UC.0)
      .
      .    (PROCESSH エンティティレポートがここに表示されます)
      .
+ ADD PLOT CODE-RANGE
+ ADD PLOT TNS-BUSY-SAMPLES
+ SET PLOT TIME-BASE OFF

+ LIST PLOT

      0:::::+:::20.0:::::+:::40.0:::::+:::60.0:::::+:::80.0:::+::::100
     A ************************++++++++++++++++++|                    -
     B ********************************++|                          -
```

```
C ************************************************************+++++++++-
  ++++++++++++++++++++++|                                                -
D +++++++++++++++|                                                       -
E +                                                                      -
F ++++++++++++++++++++++++++++++++++|                                    -
 0:::::+:::20.0:::::+:::40.0:::::+:::60.0:::::+:::80.0:::+::::100

  Min. value =   2.000                Max value =  88.000

A: FORMAT           ProcessH $SYSTEM.SYSO2.MEASCOM (7,37)
B: PRINT^T RESULT   ProcessH $SYSTEM.SYSO2.MEASCOM (7,37)
C: FIND^T           ProcessH $SYSTEM.SYSO2.MEASCOM (7,37)
D: FORMAT           ProcessH $SYSTEM.SYSO2.MEASCOM (7,37)
    (TNS)
E: PRINT^T RESULT   ProcessH $SYSTEM.SYSO2.MEASCOM (7,37)
    (TNS)
F: FIND^T           ProcessH $SYSTEM.SYSO2.MEASCOM (7,37)
    (TNS)
```

- 測定された TNS/R ネイティブコードのサンプル数を示す PROCESSH プロットを表示するには、次のように入力します（シャープ記号（#）は、TNS/R ネイティブコードを表します）。

```
+ LIST PROCESSH $SYSTEM.SYS02.TSYSDP2 (UCR)
      .
      .    (PROCESSH エンティティレポートがここに表示されます)
      .

+ ADD PLOT TNSR-BUSY-TIME
+ ADD PLOT TIME-BASE OFF
+ LIST PLOT

      0:::::+:::20.0:::::+:::40.0:::::+:::60.0:::::+:::80.0:::+::::100
    A ##########################################|                       -
    B ###################################|                              -
    C #|                                                                 -
     0:::::+:::20.0:::::+:::40.0:::::+:::60.0:::::+:::80.0:::+::::100

      Min. value =   2.000                Max value =  65.000

    A: AC1^CKPT^GET^BLOCK              ProcessH $SYSTEM.SYSO2.TSYSDP2
(7,20)
       (TNSR)
    B: AC1^RECMOVE^DATA                ProcessH $SYSTEM.SYSO2.TSYSDP2
(7,20)
       (TNSR)
    C: AC1^REDCHAIN^PURGE              ProcessH $SYSTEM.SYSO2.TSYSDP2
(7,20)
       (TNSR)
```

- L シリーズの RVU での次の LIST PLOT コマンドの表示例では、10 秒の収集間隔で測定データが収集されたという想定で、ファブリックエンティティの X-SENDS カウンターが示されます。

```
+ list /out $null/fabric *,by x-sends
+ add plot x-sends
+ list plot
$SYSTEM KMZMEAS 16..
         0:::::+:::20.0:::::+:::40.0:::::+:::60.0:::::+:::
80.0:::::+::::100
 6:54:21 B
```

```
A                                                                      -
 6:54:31 C    B
A                                          -
 6:54:41 CB
A                                                                   -
 6:54:51 C  A
B                                                                    -
 6:55:01 CB
A                                                                      -
 6:55:11 B
A                    -
 6:55:21 B
A                                                                       -
 6:55:31 B
A                                                                      -
         0:::::+:::20.0:::::+:::40.0:::::+:::60.0:::::+:::
80.0:::::+::::100

      Min value = 0.000               Max value = 72.352

   A: X-SENDS                         Fabric 0 (CPU, \BLIBO (0), STOVC01)
   B: X-SENDS                         Fabric 0 (IP, NCLIM000, COMVQP10)
   C: X-SENDS                         Fabric 0 (STORAGE, SCLIM001, MIMC)
   D: X-SENDS                         Fabric 0 (STORAGE, SCLIM001,
COMVQP00)
   E: X-SENDS                         Fabric 0 (STORAGE, SCLIM000,
COMVQP00)
   F: X-SENDS                         Fabric 0 (IP, NCLIM001, COMVQP00)
   G: X-SENDS                         Fabric 0 (IP, NCLIM000, COMVQP00)
   H: X-SENDS                         Fabric 0 (STORAGE, SCLIM000, MIME)
   I: X-SENDS                         Fabric 0 (IP, NCLIM001, COMVQP10)
   J: X-SENDS                         Fabric 0 (STORAGE, SCLIM001,
STOVC01)
   K: X-SENDS                         Fabric 0 (STORAGE, SCLIM001, MIMM)
   L: X-SENDS                         Fabric 0 (STORAGE, SCLIM001, MIME)
   M: X-SENDS                         Fabric 0 (STORAGE, SCLIM001,
STOVC02)
   N: X-SENDS                         Fabric 0 (STORAGE, SCLIM001, MIMN)
   O: X-SENDS                         Fabric 0 (STORAGE, SCLIM000, MIMM)
```

- 次の例は、収集間隔のない、L シリーズの RVU での FABRIC エンティティのプロット表示を示します。

```
+ list /out $null/ fabric *,by x-recvs,if x-recvs > 10
+ add plot x-recvs
+ list plot,scale-to 300

         0:::::+:::60.0:::::+::::120:::::+::::180:::::+::::
240:::::+::::300
       A
*************************************************************************|-
       B
*******************************************|                             -
       C
*********************************|                                       -
       D
*********************************|                                       -
       E
```

```
**********************************|                                        -
       F
**********************|                                                    -
         0:::::+:::60.0:::::+::::120:::::+::::180:::::+::::
240:::::+::::300

      Min value = 101.930              Max value = 299.689

   A: X-RECVS                          Fabric 2 (CPU, \MEASCOG (3),
IPC_HREP)
   B: X-RECVS                          Fabric 0 (CPU, \MEASCOG (3),
IPC_HREP)
   C: X-RECVS                          Fabric 2 (CPU, \MEASCOG (2),
IPC_HREQ)
   D: X-RECVS                          Fabric 0 (STORAGE, SCLIM000,
STOVC01)
   E: X-RECVS                          Fabric 0 (STORAGE, SCLIM001,
STOVC01)
   F: X-RECVS                          Fabric 0 (CPU, \MEASCOG (0),
IPC_HREQ
```

上記の凡例では、「A」はタイプ CPU の CPU 2 内の FABRIC エンティティを表し、ターゲットシステム\MEASCOG の CPU 3 をターゲットとし仮想チャネルは IPC_HREP です。「D」は、タイプ STORAGE の CPU 0 内の FABRIC エンティティを表し、エンドポイントが SCLIM000（CLIM 名）で仮想チャネルが STOVC01 です。

LISTACTIVE entity-type

LISTACTIVE *entity-type* コマンドは、アクティブな測定からのデータをアセンブルし、*entity-spec* パラメーターに含まれるエンティティごとにレポートを表示します。

カウンターレコード内の値は、初期化された時点からのカウンターのアクティビティを反映します。

- CPU と OSSCPU エンティティの場合、システムは、CPU に負荷がかかっているときにアクティブなカウンターレコードを初期化します。
- 他のタイプのエンティティの場合、カウンターレコードは、エンティティが作成されるときに初期化されます。
- 複数の測定に同じエンティティが含まれる場合、すべての測定で同じカウンターレコードを使用します。

画面最下部の二重プロンプト（++）は、現在画面上にあるレポートに別のレポートが続くことを示します。次のレポートを表示するには Return キー（または他の任意のキー）を押します。後続のレポートを無視するには、Ctrl+Y キーを押すか二重プロンプトで BREAK と入力します。

- 前のコマンドラインで複数のコマンドを入力していた場合は、BREAK により現在のコマンドが中断され、次のコマンドが実行されます。
- 複数のコマンドを入力していない場合は、BREAK により現在のコマンドが中断され、MEASCOM プロンプトに戻ります。

LISTACTIVE entity-type の構文

```
LISTACTIVE [ / OUT filename / ] [ entity-type ]
           entity-spec [ , listactive-option ] ...
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

LISTACTIVE コマンドを実行した後、MEASCOM は現在の OUT ファイル（通常、端末）への出力の書き込みを再開します。このオプションは、ログファイルの内容には影響しません。

entity-type

次のいずれかです。

```
CLUSTER      DISKFILE   NETLINE    SERVERNET    TMF
CONTROLLER   ENDPOINT   OPDISK     SQLPROC      USERDEF
CPU          FABRIC     OSSCPU     SQLSTMT
DEVICE       FILE       OSSNS      SYSTEM
DISC         LINE       PROCESS    TERMINAL
```

entity-type キーワードは、ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして指定された場合はオプションです。

LISTACTIVE には、DEVICE、DISC、FILE、LINE、NETLINE、PROCESS、SQLPROC、USERDEF、および TERMINAL エンティティに対する特殊なエンティティ構文があります。詳しくは、**LISTACTIVE エンティティ指定の特殊なケース**(108 ページ)を参照してください。

注記:

LISTACTIVE コマンドでは、DEVICE または DISC エンティティにアスタリスク（*）は使用できません。特定のデバイスまたはディスクに名前を付ける必要があります。

---

entity-spec

測定対象のエンティティを識別します。エンティティ指定はエンティティタイプごとに固有です。構文については、**エンティティおよびカウンター**(167 ページ)にある指定したエンティティタイプの説明を参照してください。

注記:

LISTACTIVE は *entity-spec* を使用してエンティティ制御ブロックに直接アクセスするため（エンティティ制御ブロックはその後システムカウンター領域のカウンターレコードにアクセスするために使用される）、*entity-spec* は正確に 1 つのエンティティを識別し、そのエンティティが存在するプロセッサーを示す必要があります。

このため、DISKFILE は ANSI SQL パーティション名を受け入れますが、ANSI SQL オブジェクト名は受け入れません。

---

listactive-option

次のいずれかです。

DOTS { ON | OFF }

レポートの表示で、ラベルと数値の間に接続のドットを含めるかどうかを設定します。STYLE が ZMS の場合にのみ有効です。デフォルト設定は、OFF です。

ON

書式設定された数値にラベルを接続するドットを使用することを表示します。

OFF

書式設定された数値にラベルを接続するドットを使用しないことを表示します。

FOR *duration*

レコード内のフィールドごとに、即値と *duration* 後の値の差異を表示します。FOR を省略すると、フィールド値には、測定の開始以降のアクティビティが反映されます。

duration は次のいずれかの形式の時間間隔です。

n SECOND[S]
n MINUTE[S]
n HOUR[S]

指定する値は次の通りです。*n* は 1～9999 の範囲の整数です。

FORMAT { BRIEF | CSV | JSON | NORMAL | STRUCTURED | XML }

レポートの形式を設定します。デフォルトは NORMAL です。

BRIEF

エンティティタイプごとに最もよく使用されるカウンターを含む簡略レポートを表示します。

簡略形式のレポートに含まれるカウンターについて詳しくは、**エンティティおよびカウンター** (167 ページ)で、指定されたエンティティタイプの DDL レコードを参照してください。簡略形式のレポートに含まれるカウンターは、各 DDL レコードで**太字**で記されています。

CSV

エンティティタイプに基づいた名前と CSV のサフィックスを持つ「C」ディスクファイル（タイプ 180）に、コマンドからの出力を CSV 形式で書き込みます。ファイルの最大サイズは 1TB です。ファイルがすでに存在している場合、MEASCOM はそのファイルを上書きします。ファイルが存在しない場合、MEASCOM が作成します。

JSON

エンティティタイプに基づいた名前と JSN のサフィックスを持つ「C」ディスクファイル（タイプ 180）に、コマンドからの出力を JSON 形式で書き込みます。ファイルの最大サイズは 1TB です。ファイルがすでに存在している場合、MEASCOM はそのファイルを上書きします。ファイルが存在しない場合、MEASCOM が作成します。

NORMAL

すべてのカウンターを表示します。

STRUCTURED

Enform を使用した後の調査のために構造化ファイルにレポートを書き込みます。1 つのレコードは、単一のエンティティのすべてのカウンターから構成されます（DDL レコード形式については、**エンティティおよびカウンター**(167 ページ)にある指定したエンティティタイプの説明を参照してください）。

同じエンティティタイプのエンティティに関するレポートは同じファイルに書き込まれ、そのファイルはエンティティタイプに応じた名前が付けられます。ファイルがすでに存在する場合、MEASCOM がデータをそのファイルに追加します。ファイルが存在しない場合、MEASCOM が作成します。（FORMAT 属性を変更すると構造化ファイルは閉じられます）。

XML

エンティティタイプに基づいた名前と XML のサフィックスを持つ「C」ディスクファイル（タイプ 180）に、コマンドからの出力を XML 形式で書き込みます。ファイルの最大サイズは 1TB です。ファイルがすでに存在している場合、MEASCOM はそのファイルを上書きします。ファイルが存在しない場合、MEASCOM が作成します。

LOADID *loadid*

このコマンドで生成されるレコードの *loadid* フィールドに配置される名前を指定します。LOADID オプションは、構造化レポートにのみ適用されます。

loadid

1～8 文字の長さの英数字文字列です。文字列には、英字、数字、キャレット（^）、ハイフン（-）、およびアンダースコア（_）を含められます。最初の文字は、英字でなければなりません。

RATE { ON | OFF }

カウンター値を表示する方法を指定します。RATE 属性は、構造化ファイルには影響がありません。デフォルトは ON です。

ON

解釈されたカウンター値（1 秒あたりの数とビジー率）を表示します。

OFF

解釈されていないカウンター値を表示します。

STYLE { LEGACY | ZMS }

ZMS 形式のインターフェイスを使用するか、Measure G10 以前の PVU と互換性のあるレガシーインターフェイスを使用して、表示または構造化データをフォーマットするかどうかを設定します。G シリーズの RVU ではデフォルトは LEGACY で、H シリーズ以降の RVU ではデフォルトは ZMS です。

LEGACY

表示および構造化レコードで Legacy 形式を使用します。

ZMS

表示および構造化レコードで ZMS 形式を使用します。

ZERO-REPORTS { INCLUDE | SUPPRESS }

すべてのゼロの値を含むレコードが表示されるかどうかを指定します。デフォルトは SUPPRESS です。

INCLUDE

すべてのカウンターの値がゼロの場合でも、エンティティレポートを表示します。

SUPPRESS

すべてのカウンターの値がゼロの場合、エンティティレポートを表示しません。

注記:

スナップショットカウンターは、特定の時点で取得された測定を返します。ゼロ以外のスナップショットカウンターの値は、必ずしも測定間隔中のアクティビティを示すわけではありません。スナップショットカウンターの値は通常、ZERO-REPORTS 句のレコードを評価するときに考慮されません。

ZERO-VALUES { INCLUDE | SUPPRESS }

0.005 未満の値をゼロまたは空白として表示するかどうかを指定します。デフォルトは SUPPRESS です。

INCLUDE

カウンターの値が 0.005 未満の場合にゼロを表示します。

SUPPRESS

カウンターの値が 0.005 未満の場合に空白を表示します。

ZERO-VALUES 属性は、構造化ファイルには影響はありません。

LISTACTIVE エンティティ指定の特殊なケース

LISTACTIVE コマンドには、いくつかのエンティティに対する特殊な構文があります。

- **DEVICE**(108 ページ)
- **DISC**(109 ページ)
- **FABRIC**(110 ページ)
- **FILE**(111 ページ)
- **LINE および NETLINE**(111 ページ)
- **PROCESS、SQLPROC、および USERDEF**(112 ページ)
- **TERMINAL**(113 ページ)

DEVICE

```
LISTACTIVE  DEVICE device-spec [ , listactive-spec ]
```

device-spec

次のように指定された、測定するデバイスを識別します。

```
{ $devicename ( cpu , svnet , group , module , slot [, scsi-id ] )}
```

または、CLIM 接続デバイスのみを指定するには、次のように指定します。

```
{ devicename ( cpu , svnet , sac , lun ) }
```

$*devicename*

測定対象のデバイスの名前です。

cpu

測定対象のデバイスが構成されている CPU の番号です。

svnet

（G シリーズ以降の RVU）X ファブリックを示す X、Y ファブリックを示す Y、または両方のファブリックを示すアスタリスク（*）。Measure 製品は、指定したファブリックに関連付けられているすべてのデバイスレコードを返します。

ServerNet デバイスアダプター（ServerNet D/A）によって接続されているデバイスには X ファブリックや Y ファブリックはありません。これらのデバイスを指定するには、アスタリスク（*）を使用します。このような場合は、エンティティ指定の残りの部分が 1 つのデバイスを記述している場合、LISTACTIVE コマンドでアスタリスク（*）を使用することができます。

FCSA または CLIM によって接続されているデバイスはデュアルファブリックのアダプターです。つまり、X と Y の両方のファブリックに接続されています。これらのデバイスの場合、LIST コマンドでは「*」または「X」のみを指定し、LISTACTIVE コマンドでは「X」のみを指定する必要があります。

group

（G シリーズ以降の RVU）測定対象のデバイスのグループ番号です。（グループは物理的なエンクロージャーに相当します）。

module

（G シリーズ以降の RVU）測定対象のデバイスのモジュール番号です。

slot

（G シリーズ以降の RVU）測定対象のデバイスのスロット番号です。

scsi-id

（G シリーズ以降の RVU）測定対象のデバイスの SCSI ポート識別子です。Measure では 0 から 65534 までの範囲がサポートされています。

sac

CLIM の名前です。DEVICE エンティティについては 1 つの CLIM 名のみがサポートされています。

lun

CLIM 内のディスクに割り当てられている論理ユニット番号です。

注記:

LISTACTIVE コマンドの場合、測定対象の特定のデバイスを指定する必要があります。ワイルドカード（*）を使用することはできません。

LISTACTIVE DEVICE の例

```
FCSA device:    + LISTACTIVE DEVICE $TAPE1 (0,X,110,3,1)
                + LISTACTIVE DEVICE $TAPE2 (0,X,110,3,1,101)
CLIM device:    + LISTACTIVE DEVICE $TAPE3 (0,X,C1002531,201)
```

DISC

```
LISTACTIVE DISC disc-spec [ , listactive-spec ]
```

disc-spec

次のように指定された、測定するディスクを識別します。

```
{ $discname ( cpu , svnet , group , module , slot [, scsi-id ] )}
```

または、複数のパスのみを持つディスクの場合は、次のように指定します。

```
{ discname-path ( cpu ) }
```

または、CLIM 接続デバイスのみを指定するには、次のように指定します。

```
{ discname ( cpu , svnet , sac , lun [, partition ] ) }
```

discname

測定対象のディスクの名前です。

cpu

ディスクが構成されている CPU の番号です。

svnet

（G シリーズ以降の RVU）デバイスへのこの特定のパスで使用される ServerNet ファブリック。Measure サブシステムは、指定されたファブリックに関連するすべてのディスクレコードを測定します。値は、X ファブリックでは 0、Y ファブリックでは 1 です。

ServerNet/DA で接続されたディスクには、X または Y ファブリックはありません。これらのディスクを指定するには、アスタリスク（*）を使用します。この場合、エンティティ指定の残りの部分が 1 つのディスクを記述している場合に限り、LISTACTIVE コマンドでアスタリスク（*）を使用することができます。

FCSA または CLIM によって接続されているディスクはデュアルファブリックのアダプターです。つまり、X と Y の両方のファブリックに接続されています。これらのディスクの場合、LIST コマ

ンドでは「*」または「X」のみを指定し、LISTACTIVE コマンドでは「X」のみを指定する必要があります。

group

（G シリーズ以降の RVU）測定対象のディスクのグループ番号です。（グループ番号は物理的なエンクロージャーに相当します）。

module

（G シリーズ以降の RVU）測定対象のディスクのモジュール番号です。

slot

（G シリーズ以降の RVU）測定対象のディスクのスロット番号です。

scsi-id

（G シリーズ以降の RVU）測定対象のディスクの SCSI ポート識別子です。ServerNet/DA および FCSA ディスクに使用され、CLIM ディスクには使用されません。

discname-path

ストレージ CLIM ディスクだけでなく FCSA ストレージディスクにも適用されます。*パス*の有効な値は「-P」、「-B」、「-M」または「-MB」です（それぞれ、プライマリ、バックアップ、ミラー、ミラーバックアップを指定します）。**discname-path** が指定された場合、svnet、sac、lun、および partition に指定された値は無視されます。

sac

CLIM の名前です。

lun

CLIM 内のディスクに割り当てられている論理ユニット番号です。Measure では 0 から 65534 までの範囲がサポートされています。

partition

パーティション分割されたディスク内のパーティションに割り当てられている番号です。H06.23/J06.12 以降の RVU の CLIM 接続ディスクに対してのみ有効です。

注記:

LISTACTIVE コマンドの場合、測定対象となる特定のディスクを指定する必要があります。ワイルドカード（*）は使用できません。

注記:

パーティションに分割した物理ディスクの総合パフォーマンスを、LISTACTIVE DISC コマンドとその同等の API によって取得することはできません。仕様では、LISTACTIVE コマンドは単一のエンティティにのみアクセスでき、パーティションに分割したディスクは複数のエンティティで表されます。

LISTACTIVE DISK の例

```
FCSA disk:                + LISTACTIVE DISC $SYSTEM (0,X,110,3,1)
                          + LISTACTIVE DISC $SYSTEM (0,X,110,3,1,101)
Path option:              + LISTACTIVE DISC $SYSTEM-P (0)
CLIM partitioned disk:    + LISTACTIVE DISC $DATA (0,X,C1002531,208,3)
CLIM unpartitioned disk:  + LISTACTIVE DISC $SYSTEM (0,X,C1002531,201)
```

FABRIC

```
LISTACTIVE FABRIC entity-spec [ , listactive-spec ]
```

entity-spec

次のように指定された、測定するファブリックを識別します。

```
{ cpu ( CPU     , system-spec , virt chan ) }
{ cpu ( STORAGE , clim-name   , virt chan ) }
{ cpu ( IP      , clim name   , virt chan ) }
{ cpu ( OPEN    , clim name   , virt chan ) }
```

system-spec

次のいずれかになります。

```
{ { \sys-name | \node-number } ( target cpu ) }
```

sys-name

ターゲットシステムのシステム名です。デフォルトはローカルシステムです。

node-number

ターゲットシステムのノード番号です（0～254 の範囲）。デフォルトはローカルシステムです。

cpu

この仮想チャネルの送信元 CPU の番号です（0～15 の範囲）。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

target-cpu

ターゲットシステムの CPU です（0～15 の範囲）。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

clim-name

CLIM の名前です。

virt-chan

仮想チャネルです。

FILE

```
LISTACTIVE FILE file-spec [ , listactive-spec ]
```

file-spec

次のように指定された、測定するファイルを識別します。

```
filename ( cpu, pin, filenum )
```

filename

ファイルの名前です。

cpu, pin

オープナーの PID です。

filenum

オープナーのファイル番号です（0～255 の範囲）。

LINE および NETLINE

```
LISTACTIVE { LINE | NETLINE } WAN-spec [ , listactive-spec ]
```

WAN-spec

次のいずれかとして指定された、測定する回線を識別します。

```
{ cpu ( line-name )              }
{ cpu ( trackid , clip , line ) }
```

cpu

0～15 の範囲のプロセッサーの数です。

line-name

$*name7* の形式での回線の名前です。ここで *name7* は最大 7 文字です。

trackid

%H*hexnum6* の形式でのトラックの番号です。ここで *hexnum6* は最大 6 文字です。

clip

1～6 の範囲の CLIP の数です。デフォルトは、すべての CLIP です。

line

0 または 1 の回線の数です。

PROCESS、SQLPROC、および USERDEF

```
LISTACTIVE { PROCESS | SQLPROC | USERDEF } process-spec [ , listactive-spec ]
```

process-spec

次のいずれかとして指定された、測定対象のプロセスを識別します。

```
{ cpu,pin                                                  }
{ $process-name ( cpu,pin )                                }
{ [[device.]subvolume.]filename[:crvsn] [ ( cpu,pin ) ] }
```

cpu,pin

cpu が 0～15 の範囲内にあり、*pin* が 0～65534 の範囲内にある PID です。

$*process-name*

プロセスの名前です。

$*device*

プロセスのオブジェクトファイルが置かれているボリューム（デバイス）です。

subvolume

プロセスのオブジェクトファイルが置かれているサブボリュームです。

filename

実行中のプロセスのオブジェクトファイルの名前です。

:*crvsn*

Measure G09 以降の PVU では、一意のファイル名を形成するために必要な、タイムスタンプ、作成バージョンのシリアル番号（CRVSN）、またはファイル名拡張子です。ファイルの CRVSN を取得するには、Measure のレポートを表示するか、LISTGNAME コマンドを使用します。

TERMINAL

```
LISTACTIVE TERMINAL terminal-spec [ , listactive-spec ]
```

terminal-spec

次のように指定して、測定する端末を識別します。

***cpu* (*line-name.subdevice*)**

cpu は、0～15 の範囲のプロセッサーの番号です。

line-name は回線の名前です。{ $*name7* | $*num* | *logical-Define-name* }のように指定します。

subdevice はサブデバイスの名前です。{ #*name7* }のように指定します。

関連コマンド

コマンド	機能
LIST	データファイルからカウンター値を読み取ります
SET REPORT	デフォルトのレポート形式を指定します

すべての PVU および RVU での LISTACTIVE entity-type の使用上の注意

- LISTACTIVE コマンドを使用して、DISKFILE エンティティのアクティブなレコードを読み取ることができます。LISTACTIVE DISKFILE コマンドを使用する場合、ディスクファイル指定が有効で一意のディスクファイル名である必要があります。MEASCOM は、現在の SYSTEM および VOLUME のデフォルトで、指定したファイル名を一部展開します。
- LISTACTIVE コマンドは、DISCOPEN カウンターや PROCESSH カウンターの読み取りには使用できません。
  - アクティブなファイルのオープン制御ブロック（OCB）番号を一意に特定する方法はないので、DISCOPEN レコードは LISTACTIVE コマンドには使用できません。MEASCTL では、レコードの検索に OCB が必要です。
  - 複雑な測定はシステムパフォーマンスを低下させる可能性があるので、PROCESSH レコードは使用できません。

  収集間隔を指定している場合、LIST コマンドを使用して、現在アクティブなデータファイルから DISCOPEN カウンターと PROCESSH カウンターを読み取ることができます。収集間隔を使用する場合、データは内部バッファーに収集され、そのバッファーがいっぱいになるとデータファイルに書き込まれます。データがファイルに書き込まれる前に、1 つまたは複数の収集間隔が経過することがあります。
- FILE および PROCESS のエンティティは、開いているファイルの名前を表示します。プロセスはファイル以外のリソース（プロセスや$0 など）を開くことができるので、これらのエンティティのレポートには、Measure の構文によれば無効であるファイル名が含まれることがあります。このようなレポートを表示するには、そのファイル名ではなく、リソースの CPU、PIN、およびファイル番号を指定します。**LIST entity-type**(75 ページ)にある例を参照してください。
- LISTACTIVE コマンドによって表示される情報は、LIST コマンドで表示される情報と異なる場合があります。
  - LISTACTIVE コマンドの場合、MEASCOM は、それぞれの CPU における Measure 制御プロセス（MEASCTL）からデータを要求します。MEASCTL は、システムデータ領域内のアクティブなカウンターレコードから要求されたデータを読み取ります。
  - LIST コマンドの場合、MEASCOM は、それぞれのデータファイルの Measure ファイル処理プロセス（MEASFH）からデータを要求します。MEASFH は、データファイルに格納された情報を使用して、測定ウィンドウ中（または LIST コマンドのレポートウィンドウ中）にカウンターのアクティビティを特定します。

L シリーズの PVU および RVU での LISTACTIVE entity-type の使用上の注意

L シリーズの PVU および RVU では、LISTACTIVE SERVERNET コマンドは、GMS、ポート、およびファイバー仕様をサポートしません。L シリーズの RVU の SERVERNET エントリーでは、CLIM に関連する IPC および RIPC ノードクラスのみが利用できます。

H シリーズ以降の PVU および RVU での LISTACTIVE entity-type の使用上の注意

- H06.23/J06.12 以降の RVU では、CLIM 接続ディスクに対する LISTACTIVE DISC コマンドは、オプションの DISC 指定パラメーターとしてパーティション番号をサポートします。パーティション番号は、パーティション分割されたディスク内のパーティションに割り当てられている番号です。さらに、CLIM 接続ディスクのパーティション番号が測定で指定されていれば、そのパーティション番号が LISTACTIVE DISC コマンドで表示されます。**DISC**(109 ページ)については「LISTACTIVE エンティティ指定の特殊なケース」のサブセクションも参照してください。
- Measure H04/J02 以降の PVU では、LISTACTIVE コマンドは、レポート出力ファイルが構造化ファイルである場合、測定データファイルのサイズに応じて形式 1 または形式 2 の構造化ファイルを作成します。
- Measure H03/J01 以降の PVU では、LISTACTIVE コマンドは、CLIM を使用する SERVERNET、DEVICE、および DISC エンティティ指定を可能にします。構文と例については、**SERVERNET**(399 ページ)、**DEVICE**(214 ページ)、および **DISC**(226 ページ)エンティティを参照してください。
- Measure H01 以降の PVU では、LISTACTIVE コマンドは、ANSI SQL 名を使用する DISKFILE、FILE、および SQLSTMT のエンティティ指定を可能にします。構文については、**DISKFILE**(259 ページ)、**FILE**(284 ページ)、および **SQLSTMT**(421 ページ)エンティティを参照してください。

G シリーズの PVU での LISTACTIVE entity-type の使用上の注意

Measure G12 以降の PVU では、TNS コードファイルでのみコード領域識別子が表示されます（高速化 TNS コードファイルを含む）。

LISTACTIVE entity-type の例

- HPE NonStop K シリーズサーバーでのディスク$NEW01 に関する情報を表示するには、次のように入力します。

```
+ SET REPORT FORMAT BRIEF
+ LISTACTIVE DISC $NEW01 (2,0,25,2)
Disc $NEW01 (4160 -- 895MB)   PID 2,8   Ldev 5   C,C,U (0,%25,%2)
Local System \BILLS From 18 Oct 1994, 14:35:33 For 113 Seconds

Disc-Busy-Time        39.53 %    Disc-Rate        16.04
Request Qtime          0.62 #    Request-Qlen-Max     3 #
```

- HPE NonStop S シリーズサーバーでのディスク$DATA01 に関する情報を表示するには、次のように入力します。

```
+ LISTACTIVE DISC $DATA01 (X, DISK-DEVICE-14)
Disc $DATA01      (4160 -- 4GB)   Type 3  Subtype 23 PID 0,287
Cfg:    DISK-DEVICE-14
SAC:   CONTROLLER-2
GMS  1,4,402   SvNet X   Logical Device 34
Local System \ACCT   From 28 Oct 1995, 16:43:24  For 179 seconds

Disc-Busy-Time          43.51 %  Disc-Rate           13.24
Request-Qtime            0.42 #  Request-Qlen-Max       2#
```

- 次の例では、CPU 1 のレポートが表示されます。CPU レコードはシステムロード時に初期化されるので、測定には、システムロード時からレポート生成時点までの CPU アクティビティが反映されます。

```
+ LISTACTIVE CPU 1
Cpu  1 VLX          Initial Lock Pgs 2048      Mem Pages  8192
Memory MB 16        PCBs    200                Page Size  2048
Local System \BILLS From 10 Oct 1994, 18:45:18 For 187 Hours

Cpu-Busy-Time       0.67%   Swaps                0.01
Cpu-Qtime           0.01#   Cpu-Qlen-Max          13#
Mem-Qtime                   Mem-Qlen-Max           4#
Dispatches           5.63    Intr-Busy-Time      0.35%
Process-Ovhd                Send-Busy-Time
Disc-IOs            0.02    Cache-Hits           0.01
Transactions                 Response-Time
Page-Requests               Page-Scans
Ending-Free-Mem      4092   Ending-UCME
Ending-UDS            500   Ending-SDS           1500
Ending-UCL            100   Ending-SCL           2000
```

- その他の例は、**LIST entity-type の例**(84 ページ)を参照してください。

LISTALL entity-type

LISTALL *entity-type* コマンドは、現在のデータファイル（ADD MEASUREMENT コマンドで最後に指定されたファイル）からデータを読み取り、*entity-spec* パラメーターに含められたエンティティごとにレポートを表示します。

LISTALL コマンドの機能と構文は、LIST *entity-type* コマンドに似ています。ただし、LISTALL は、指定した測定ウィンドウ内での各間隔レコードを表示する点で異なります。

二重プロンプト（++）は、画面上のレポートに別のレポートが続くことを示します。次のレポートを表示するには Return キー（または他の任意のキー）を押します。後続のレポートを無視するには、Ctrl+Y キーを押すか二重プロンプトで BREAK と入力します。

- 前のコマンドラインで複数のコマンドを入力していた場合は、BREAK により現在のコマンドが中断され、次のコマンドが実行されます。
- 複数のコマンドを入力していない場合は、BREAK により現在のコマンドが中断され、MEASCOM プロンプトに戻ります。

LISTALL entity-type の構文

```
LISTALL [ / OUT filename / ] [ entity-type ] entity-spec
        [ , list-option ] ...
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

LISTALL コマンドを実行した後、MEASCOM は現在の OUT ファイル（通常、端末）への出力の書き込みを再開します。このオプションは、ログファイルの内容には影響しません。

entity-type

次のいずれかです。

```
CLUSTER     DISCOPEN   LINE      PROCESS    SYSTEM
CONTROLLER  DISKFILE   NETLINE   PROCESSH   TERMINAL
CPU         ENDPOINT   OPDISK    SERVERNET  TMF
```

```
DEVICE      FABRIC     OSSCPU   SQLPROC    USERDEF
DISC        FILE       OSSNS    SQLSTMT
```

entity-type キーワードは、ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして指定された場合はオプションです。

entity-spec

測定対象のエンティティを識別します。エンティティ指定はエンティティタイプごとに固有です。構文については、**エンティティおよびカウンター**(167 ページ)にある指定したエンティティタイプの説明を参照してください。

list-option

次のいずれかです。

```
BY item-name [ ( ASCENDING ) | ( DESCENDING ) ]
CR-NAME-LEN { SHORT | LONG }
CR-NAME-FORM { STANDARD | DEMANGLED | BOTH }
CR-NAME-QUAL { UNQUALIFIED | QUALIFIED }

COUNT record-count


DOTS { ON | OFF }

FIRST record-num


FOR duration

FORMAT { BRIEF | CSV | JSON | NORMAL | STRUCTURED | XML }

FROM [ [ start-date,] start-time-of-day ]
IF { item-name operation value }
LOADID loadid


RATE { ON | OFF }
RECORD-NUM { ON | OFF }


STYLE { LEGACY | ZMS }
TO [ end-date,] end-time-of-day ]
TOLERANCE { ON | OFF }
TOTALS { INCLUDE | ONLY | SUPPRESS }
ZERO-REPORTS { INCLUDE | SUPPRESS }
ZERO-VALUES { INCLUDE | SUPPRESS }
```

これらのリストオプションの詳細な構文と説明については、**LIST entity-type**(75 ページ)コマンドを参照してください。

関連コマンド

コマンド	機能
LIST *entity-type*	選択したエンティティのレポートを表示します
LISTACTIVE *entity-type*	アクティブなエンティティに関するレポートを表示します
SET REPORT	レポート形式のオプションを設定します

すべての PVU および RVU での LISTALL entity-type の使用上の注意

- 測定に間隔がない場合、または *entity-type* が PROCESSH である場合、LISTALL コマンドは LIST *entity-type* コマンドと同じ結果を生成します。
- LISTALL コマンドは C20 以降のバージョンのデータファイルでのみ使用できます。
- STRUCTURED レポートオプションを使用する場合（レポートを構造化ファイルに送信）、以下のリストオプションは使用できません。

  ```
  BY item-name
  RATE ON
  ```

L シリーズの PVU および RVU での LISTALL entity-type の使用上の注意

- L16.05/J06.20 以降の RVU では、COUNT、FIRST、および RECORD-NUM を使用するときに、各外部レコードは、PROCESSH と USERDEF を除くすべてのエンティティの 1 つのレコードとして数えられます。PROCESSH と USERDEF では、各プロセスが 1 つのレコードとして数えられます。
- L シリーズの RVU 上に作成された測定の場合、LIST コマンドは、SERVERNET エンティティの GMS、ポート、またはファイバー指定をサポートしません。これらの指定は、H および J シリーズのシステム上で作成された測定では引き続きサポートされます。
- L シリーズの RVU 上に作成された測定の場合、Adpt（アダプター）フィールドには「CPUnn-MITE」ではなく「HCA-1」が表示され、SERVERNET エンティティでは GMS、ポート、およびファイバーのフィールドは表示されません。

H シリーズ以降の PVU および RVU での LISTALL entity-type の使用上の注意

- H06.25/J06.14 以降の RVU で、MEASFT SPLIT BY TIME コマンドを使用して分割したデータファイルのレポートを生成する場合は、LISTALL *entity-type* コマンドで（TIME 分割データファイルの期間内で）FROM および TO オプションを指定して、元のデータファイルに対して行われた Measure のレポートと、TIME 分割データファイルに対して行ったレポートとの忠実性を確保する必要があります。
- H06.23/J06.12 以降の RVU では、CLIM 接続ディスクに対する LISTALL DISC コマンドは、オプションの DISC 指定パラメーターとしてパーティション番号をサポートします。パーティション番号は、パーティション分割されたディスク内のパーティションに割り当てられている番号です。IF および BY 句でパーティション番号を指定する PARTITION の使用もサポートされます。

  さらに、CLIM 接続ディスクのパーティション番号が測定で指定されていれば、そのパーティション番号が LISTALL DISC コマンドで表示されます。
- Measure H04/J02 以降の PVU では、CPU および PROCESS エンティティでは、測定間隔の開始から終了まで PROCESSH サンプル数が変更されていない場合、PROCESSH 測定がアクティブになっていても、NATIVE-BUSY-TIME、ACCEL-BUSY-TIME、および TNS-BUSY-TIME のフィールドは MEASCOM によって表示されません。LISTALL コマンドでは、これらのフィールドは表示される間隔と表示されない間隔があります。
- Measure H04/J02 以降の PVU では、LISTALL コマンドは、レポート出力ファイルが構造化ファイルである場合、測定データファイルのサイズに応じて形式 1 または形式 2 の構造化ファイルを作成します。
- Measure H03/J01 以降の PVU では、LISTALL コマンドは、CLIM を使用する SERVERNET、DEVICE、および DISC エンティティ指定を可能にします。構文と例については、**SERVERNET**(399 ページ)、**DEVICE**(214 ページ)、および **DISC**(226 ページ)エンティティを参照してください。
- Measure H01 以降の PVU では、LISTALL コマンドは、ANSI SQL 名を使用する DISCOPEN、DISKFILE、FILE、および SQLSTMT のエンティティ指定を可能にします。構文については、**DISCOPEN**(249 ページ)、**DISKFILE**(259 ページ)、**FILE**(284 ページ)、および **SQLSTMT**(421 ページ)エンティティを参照してください。
- H シリーズの RVU では、IF 句は、Legacy 形式のレコードおよびレポートにのみ表示されるフィールドを、Legacy 形式のレポートモードの場合でも参照できません。

G シリーズの PVU および RVU での LISTALL entity-type の使用上の注意

- Measure G12 以降の PVU では、TNS コードファイルでのみコード領域識別子が表示されます（高速化 TNS コードファイルを含む）。
- G シリーズの RVU では、ZMS 形式のレポートモードの場合でも、IF 句は、ZMS 形式のレコードおよびレポートにのみ表示されるフィールドを参照できません。

LISTALL entity-type の例

- 以下のコマンドが CPU アクティビティの測定で時間単位の間隔を確立すると想定します。

  ```
  + ADD CPU *
  + START MEASDATA, INTERVAL 1 HOUR, FROM 8:00, TO 14:00
  ```

  LISTALL コマンドは、9:00～10:00、10:00～11:00、および 11:00～12:00 の各間隔時に CPU 0 のアクティビティを示す 3 つのレコードを生成します。

  ```
  + LISTALL CPU 0, FROM 9:00, TO 12:00
  ```

  一方、次の LIST コマンドは、9:00～12:00 の CPU 0 のアクティビティを示す単一のレコードを生成します。

  ```
  + LIST CPU 0, FROM 9:00, TO 12:00
  ```

- 次の例は、同じ時間間隔の LIST コマンドと LISTALL コマンドの出力の差異を示します。

```
+ add cpu *
+ start mdfcpuh, interval 1 hour, from 15:00, to 21:00
...
(Sometime later, after the measurement has stopped at 21:00)
+ add mdfcpuh
+ list cpu 0, from 16:00, to 19:00
Cpu  0 NSE-M          Init Lock Pgs     19          Mem Pages       1048576
Memory MB      16384  PCBs           12100          Pg Size    16384  Bytes
IPUs    2
Format Version:  H06  Data Version:  H06  Subsystem Version:  3
Local System \BLPROD  From  24 Jan 2014, 16:00:00   For     3 Hours
------------ Processor
-----------------------------------------------------
Cpu-Busy-Time                1.24 %    Dispatches                 1,759 /s
Cpu-Qtime                    0.03 AQL  Intr-Busy-Time               0.24 %
Comp-Traps                 392.48 /s   Process-Ovhd
------------ Memory
-----------------------------------------------------
Starting-Free-Mem         901,211 #    Ending-Free-Mem           901,487 #
Swaps                        1.28 /s   Page-Requests               10.93 /s
MM-Page-Scans               12.13 /s   Page-Scans                  12.13 /s
Unsp-Pages-Qtime             7.86 AQL  Ending-UCME
Unsp-Pages-Start                8 #    Unsp-Pages-End                14 #
Starting-SCL               84,779 #    Ending-SCL                 84,779 #
Starting-UCL              226,628 #    Ending-UCL                227,252 #
Starting-Free-CIDs         61,448 #    Ending-Free-CIDs           61,559 #
Starting-Timer-Cells                   Ending-Timer-Cells
Locked-Pages-Start         69,198 #    Locked-Pages-End           69,727 #
------------ Device I/O
-----------------------------------------------------
Disc-IOs                     4.65 /s   Cache-Hits                124.33 /s
Transactions                           Response-Time
------------ Messaging
-----------------------------------------------------
```

```
Link-Prepush-Msgs          135.47 /s   Readlinkcache-All          522.14 /s
Link-Readlink-Msgs          12.92 /s   Readlinkcache-Ctrl          74.25 /s
Link-Large-Msgs              0.14 /s   Readlinkcache-None
                                       Replyctrlcache-Msgs        492.79 /s


------------ IPUs
-------------------------------------------------------------
IPU 0
Ipu-Busy-Time                1.26 %    Ipu-Dispatches              1,392 /s
Ipu-Qtime                    0.01 AQL
IPU 1
Ipu-Busy-Time                1.22 %    Ipu-Dispatches              367.11 /s
Ipu-Qtime                    0.01 AQL


------------ ServerNet Transfers Initiated by this Local Processor
---------
to/fm     READS INTO CPU     READ BYTES    WRITES FROM CPU    WRITE BYTES
CPU-0          750.14 /s     1,002,000 /s        780.71 /s       568,986 /s
CPU-1
CPU-2
CPU-3
CPU-4
CPU-5
CPU-6
CPU-7
CPU-8
CPU-9
CPU-10
CPU-11
CPU-12
CPU-13
CPU-14
CPU-15

See TOTAL for sum over all initiating CPUs: total data into & from each
CPU.
+ listall cpu 0, from 16:00, to 19:00
Cpu  0 NSE-M          Init Lock Pgs     19          Mem Pages       1048576
Memory MB      16384  PCBs           12100          Pg Size    16384  Bytes
IPUs    2
Format Version:  H06  Data Version:  H06  Subsystem Version:  3
Local System \BLPROD  From  24 Jan 2014, 16:00:00   For    60 Minutes
------------ Processor
-------------------------------------------------------
Cpu-Busy-Time                1.74 %    Dispatches                  2,248 /s
Cpu-Qtime                    0.04 AQL  Intr-Busy-Time                0.30 %
Comp-Traps                 739.96 /s   Process-Ovhd                  0.01 %
------------ Memory
-------------------------------------------------------------
Starting-Free-Mem          901,211 #   Ending-Free-Mem            901,208 #
Swaps                        3.48 /s   Page-Requests                32.44 /s
MM-Page-Scans               12.13 /s   Page-Scans                   12.13 /s
Unsp-Pages-Qtime             7.27 AQL  Ending-UCME
Unsp-Pages-Start                8 #    Unsp-Pages-End                  8 #
Starting-SCL                84,779 #   Ending-SCL                  84,779 #
```

```
Starting-UCL              226,628 #    Ending-UCL                226,631 #
Starting-Free-CIDs         61,448 #    Ending-Free-CIDs           61,453 #
Starting-Timer-Cells                   Ending-Timer-Cells
Locked-Pages-Start         69,198 #    Locked-Pages-End           69,471 #
------------ Device I/O
-------------------------------------------------
Disc-IOs                     6.25 /s   Cache-Hits                190.75 /s
Transactions                           Response-Time
------------ Messaging
-------------------------------------------------------
Link-Prepush-Msgs          258.21 /s   Readlinkcache-All          582.25 /s
Link-Readlink-Msgs          38.42 /s   Readlinkcache-Ctrl          96.96 /s
Link-Large-Msgs              0.12 /s   Readlinkcache-None
                                       Replyctrlcache-Msgs        541.43 /s


------------ IPUs
-------------------------------------------------------------
IPU 0
Ipu-Busy-Time                1.91 %    Ipu-Dispatches              1,736 /s
Ipu-Qtime                    0.02 AQL
IPU 1
Ipu-Busy-Time                1.57 %    Ipu-Dispatches            512.29 /s
Ipu-Qtime                    0.02 AQL


------------ ServerNet Transfers Initiated by this Local Processor
---------
to/fm     READS INTO CPU     READ BYTES    WRITES FROM CPU    WRITE BYTES
CPU-0          981.04 /s     1,047,703 /s        987.82 /s       599,014 /s
CPU-1
CPU-2
CPU-3
CPU-4
CPU-5
CPU-6
CPU-7
CPU-8
CPU-9
CPU-10
CPU-11
CPU-12
CPU-13
CPU-14
CPU-15
See TOTAL for sum over all initiating CPUs: total data into & from each
CPU.


Cpu  0 NSE-M          Init Lock Pgs     19          Mem Pages       1048576
Memory MB      16384  PCBs           12100          Pg Size    16384  Bytes
IPUs    2
Format Version:  H06  Data Version:  H06  Subsystem Version:  3
Local System \BLPROD  From  24 Jan 2014, 17:00:00   For    60 Minutes
------------ Processor
-------------------------------------------------------
Cpu-Busy-Time                1.01 %    Dispatches                  1,449 /s
```

```
Cpu-Qtime                    0.02 AQL  Intr-Busy-Time              0.18 %
Comp-Traps                 219.45 /s   Process-Ovhd
------------ Memory
-------------------------------------------------------
Starting-Free-Mem         901,208 #    Ending-Free-Mem           901,237 #
Swaps                                  Page-Requests               0.09 /s
MM-Page-Scans               12.13 /s   Page-Scans                 12.13 /s
Unsp-Pages-Qtime             7.71 AQL  Ending-UCME
Unsp-Pages-Start                8 #    Unsp-Pages-End                7 #
Starting-SCL               84,779 #    Ending-SCL                84,779 #
Starting-UCL              226,631 #    Ending-UCL               226,603 #
Starting-Free-CIDs         61,453 #    Ending-Free-CIDs          61,507 #
Starting-Timer-Cells                   Ending-Timer-Cells
Locked-Pages-Start         69,471 #    Locked-Pages-End          69,117 #
------------ Device I/O
-----------------------------------------------------
Disc-IOs                     3.87 /s   Cache-Hits                 91.96 /s
Transactions                           Response-Time
------------ Messaging
-----------------------------------------------------
Link-Prepush-Msgs           71.36 /s   Readlinkcache-All         454.94 /s
Link-Readlink-Msgs           0.17 /s   Readlinkcache-Ctrl         68.62 /s
Link-Large-Msgs              0.06 /s   Readlinkcache-None
                                       Replyctrlcache-Msgs       435.61 /s


------------ IPUs
-------------------------------------------------------------
IPU 0
Ipu-Busy-Time                0.93 %    Ipu-Dispatches             1,229 /s
Ipu-Qtime                    0.01 AQL
IPU 1
Ipu-Busy-Time                1.08 %    Ipu-Dispatches             219.17 /s
Ipu-Qtime                    0.01 AQL


------------ ServerNet Transfers Initiated by this Local Processor
---------
to/fm     READS INTO CPU     READ BYTES    WRITES FROM CPU    WRITE BYTES
CPU-0          598.06 /s       578,631 /s        601.04 /s       203,309 /s
CPU-1
CPU-2
CPU-3
CPU-4
CPU-5
CPU-6
CPU-7
CPU-8
CPU-9
CPU-10
CPU-11
CPU-12
CPU-13
CPU-14
CPU-15
See TOTAL for sum over all initiating CPUs: total data into & from each
CPU.
```

```
Cpu  0 NSE-M          Init Lock Pgs     19          Mem Pages       1048576
Memory MB      16384  PCBs           12100          Pg Size    16384  Bytes
IPUs    2
Format Version:  H06  Data Version:  H06  Subsystem Version:  3
Local System \BLPROD  From  24 Jan 2014, 18:00:00   For    60 Minutes
------------ Processor
-----------------------------------------------------
Cpu-Busy-Time                0.97 %    Dispatches                 1,581 /s
Cpu-Qtime                    0.02 AQL  Intr-Busy-Time               0.23 %
Comp-Traps                 218.02 /s   Process-Ovhd
------------ Memory
-----------------------------------------------------------
Starting-Free-Mem         901,237 #    Ending-Free-Mem           901,487 #
Swaps                        0.35 /s   Page-Requests                0.25 /s
MM-Page-Scans               12.13 /s   Page-Scans                  12.13 /s
Unsp-Pages-Qtime             8.60 AQL  Ending-UCME
Unsp-Pages-Start                7 #    Unsp-Pages-End                14 #
Starting-SCL                84,779 #    Ending-SCL                 84,779 #
Starting-UCL               226,603 #    Ending-UCL                227,252 #
Starting-Free-CIDs          61,507 #    Ending-Free-CIDs           61,559 #
Starting-Timer-Cells                    Ending-Timer-Cells
Locked-Pages-Start          69,117 #    Locked-Pages-End           69,727 #
------------ Device I/O
-------------------------------------------------
Disc-IOs                     3.83 /s   Cache-Hits                  90.27 /s
Transactions                           Response-Time
------------ Messaging
-------------------------------------------------
Link-Prepush-Msgs           76.84 /s   Readlinkcache-All          529.25 /s
Link-Readlink-Msgs           0.16 /s   Readlinkcache-Ctrl          57.16 /s
Link-Large-Msgs              0.26 /s   Readlinkcache-None
                                       Replyctrlcache-Msgs        501.33 /s


------------ IPUs
-------------------------------------------------------------
IPU 0
Ipu-Busy-Time                0.94 %    Ipu-Dispatches               1,211 /s
Ipu-Qtime                    0.01 AQL
IPU 1
Ipu-Busy-Time                1.01 %    Ipu-Dispatches             369.88 /s
Ipu-Qtime                    0.01 AQL


------------ ServerNet Transfers Initiated by this Local Processor
---------
to/fm     READS INTO CPU     READ BYTES    WRITES FROM CPU    WRITE BYTES
CPU-0          671.31 /s     1,381,942 /s        753.28 /s       904,636 /s
CPU-1
CPU-2
CPU-3
CPU-4
CPU-5
CPU-6
CPU-7
```

```
CPU-8
CPU-9
CPU-10
CPU-11
CPU-12
CPU-13
CPU-14
CPU-15
See TOTAL for sum over all initiating CPUs: total data into & from each
CPU.
```

LISTENAME

Measure G11 以降の PVU では、LISTENAME コマンドは、Guardian ファイル名と作成バージョンシリアル番号（CRVSN）を、対応する外部名（ANSI SQL 名または OSS パス名）に変換します。ファイル名を再利用する場合、CRVSN は、Guardian ファイル名の特定のインスタンスを識別します。

LISTENAME の構文

```
LISTENAME [ / OUT filename / ] guardian-name[:crvsn]
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

guardian-name

変換対象の Guardian ファイル名です。

:*crvsn*

Guardian ファイル名の特定のインスタンスに関連付けられた CRVSN です。CRVSN 値を取得するには、MEASCOM エンティティ表示を確認します。CRVSN はコロンを使用して Guardian ファイル名に追加されます。

LISTENAME の例

この例では、ANSI SQL 名に変換した Guardian ファイル名を示します。

```
+ LISTENAME $DATA01.ZSD12345.Z1234567:340359
$DATA.ZSD12345.Z1234567:340259 'TABLE CATALOG_12.SCHEMA_34.TABLE_56
PARTITION PARTITION_78'
```

LISTGNAME

LISTGNAME コマンドは、OSS ファイルパス名（Measure G09 以降の PVU）または ANSI SQL 名（Measure H01 以降の PVU）を、対応する Guardian ファイル名と作成バージョンシリアル番号（CRVSN）に変換します。ファイル名を再利用する場合、CRVSN は Guardian ファイル名の特定のインスタンスを識別します。

指定した OSS ファイルパス名がディレクトリを識別する場合、ディレクトリの内容が一覧表示されます。ディレクトリのサブディレクトリは OSS ファイルパス名で表示され、ファイルは Guardian ファイル名と完全修飾の OSS ファイルパス名で表示されます。

OSS ファイルパス名の変換は、現在の MEASCOM セッションと、追加または削除されるデータファイルのタイプによって影響されます。以下の例では、このパス名変換を決定する方法について説明します。

OSS ファイルパス名の変換の基本	MEASCOM セッションの履歴
OSS ファイルシステムインターフェイス	MEASCOM セッションが ADD MEASUREMENT コマンドを実行していないか、MEASCOM セッションがアクティブなデータファイル、または OSS ジャーナルセグメントを含んでいない現在のシステムのデータファイルに対し、ADD MEASUREMENT コマンドを実行しています。
OSS ジャーナルの内容	MEASCOM セッションは、OSS ジャーナルセグメントを含むデータファイルに対し、ADD MEASUREMENT コマンドを実行しています。
特定できない	MEASCOM セッションは、OSS ジャーナルセグメントを含まないデータファイルに対して ADD MEASUREMENT コマンドを実行しており、データファイルは現在のノードからのものではありません。エラーメッセージが表示されます。

LISTGNAME の構文

```
LISTGNAME [ / OUT filename / ]
    { "pathname"                                                    }
    { '{TABLE|INDEX} [[catalog.]schema.]object[ PARTITION partition]'    }
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

"*pathname*"

完全修飾で有効な OSS ファイルパス名または部分的な OSS ファイルパス名です。OSS ファイルパス名は大文字と小文字が区別され、二重引用符（""）で囲んで指定する必要があります。

完全修飾の有効な OSS ファイルパス名を取得するには、*name* を現在の OSSPATH 設定と組み合わせます。OSS ファイルパス名がバックスラッシュ（/）で始まらない場合は、バックスラッシュの前に OSSPATH の現在の設定を置くことで拡張します。

'{TABLE|INDEX} [[*catalog.*] *schema.*] *object* [PARTITION *partition*]'

（Measure H01 以降の PVU）完全修飾の ANSI SQL 名で使用される ANSI SQL 識別子です。完全修飾の ANSI SQL 名には、TABLE または INDEX のどちらかが必要で、オプションで PARTITION を続けます。

LISTGNAME の使用上の注意

- OSS ファイルパス名は長くでき、画面の表示幅と Guardian EDIT ファイルの最大行長の両方を超える可能性があります。長い OSS ファイルパス名を含む出力は、表示または編集ファイルの後続行に 80 文字のセグメントで折り返されます。
- Measure は、以下の警告またはエラーメッセージを表示することがあります。

  ```
  3116 Unable to translate OSS file pathname.
  ```

LISTGNAME の例

- 「*pathname*」で OSS ファイルを指定すると、次の情報が表示されます。

```
+ LISTGNAME "/Path/to/the/file"
$DATA01.ZYQ00001.Y00003D2:340359       "/Path/to/the/file"
```

- 「*pathname*」で OSS ディレクトリを指定すると、次の情報が表示されます。

```
+ LISTGNAME "/Path/to/the"
Directory:                          "/Path/to/the/Sourcefiles"
Directory:                          "/Path/to/the/Objectfiles"
$DATA01.ZYQ00001.Y00003D2:340359    "/Path/to/the/file"
$DATA01.ZYQ00001.Y00003D2:340223    "/Path/to/the/file1"
$DATA01.ZYQ00001.Y00003D2:340864    "/Path/to/the/file2"
```

- ANSI SQL を指定すると、次の情報が表示されます。

```
+ LISTGNAME 'TABLE CATALOG_12.SCHEMA_34.TABLE_56'
$DATA01.ZSD12345.Z1234567:340259
'TABLE CATALOG_12.SCHEMA_34.TABLE_56 PARTITION PARTITION_01'
$DATA01.ZSD12345.Z1234567:340282
'TABLE CATALOG_12.SCHEMA_34.TABLE_56 PARTITION PARTITION_02'
$DATA01.ZSD12345.Z1234567:340276
'TABLE CATALOG_12.SCHEMA_34.TABLE_56 PARTITION PARTITION_03'
```

LISTPNAME

Measure G09 以降の PVU で、LISTPNAME コマンドは、Guardian ファイル名（*gname*）と作成バージョンシリアル番号（CRVSN）を、対応する OSS ファイルパス名に変換します。ファイル名を再利用する場合、CRVSN は、Guardian ファイル名の特定のインスタンスを識別します。指定された Guardian ファイル名が 1 つまたは複数のパスを通じてアクセス可能なファイルを識別する場合、すべてのパスが表示されます。

OSS ファイルパス名の変換は、現在の MEASCOM セッションと、追加または削除されるデータファイルのタイプによって影響されます。このパス名変換を決定する方法を説明した例については、**LISTGNAME**(123 ページ)の表を参照してください。

Measure G11 以降の PVU では、LISTPNAME は引き続き使用できますが、OSS ファイルパス名と ANSI SQL 名の両方をサポートする **LISTENAME**(123 ページ)に置き換えられます。

LISTPNAME の構文

```
LISTPNAME [ / OUT filename / ] gname [:crvsn]
```

OUT *filename*

filename へのコマンド出力を指示します。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

gname

変換対象の Guardian ファイル名です。

:*crvsn*

gname の特定のインスタンスに関連付けられた CRVSN です。CRVSN は、MEASCOM エンティティ表示から取得でき、コロンを使用して Guardian ファイル名に付加されます。

LISTPNAME の使用上の注意

- /G 形式の Guardian ファイル名は、常に、有効な OSS ファイルパス名として存在するとみなされます。/G 名は、変換が OSS ジャーナルセグメントを通じて行われた場合にのみ LISTPNAME 出力に表

示され、測定データファイルに記録されたアクセス情報は、特定のレコードのファイルへのアクセスに/G 形式が使用されたことを示します。

- Measure は、このコマンドに対して以下の警告またはエラーメッセージを表示することがあります。

```
3023 WARNING. CRVSN was not specified, translation may be incorrect.

3114 OSS journal segment is required in this context and is not available
for access.
```

LISTPNAME の例

次の例は、*CRVSN* が指定されていない LISTPNAME コマンドの出力を示します。この場合、潜在的に正しくない OSS ファイルパス名変換が警告に示されます。

```
+ LISTPNAME $DATA01.ZYQ00001.Y00003D2
  MEAS 3023 WARNING. Crvsn was not specified, OSS file pathname translation
     may not be correct.
  $DATA01.ZYQ00001.Y00003D2:340259    "/Path/to/the/file"
  $DATA01.ZYQ00001.Y00003D2:340259    "/Another/Path/to/the/file"
  $DATA01.ZYQ00001.Y00003D2:340259    "/Still/another/path"
```

次の例は、CRVSN が指定された LISTPNAME コマンドの出力を示します。

```
+ LISTPNAME $DATA01.ZYQ00001.Y00003D2:340359
  $DATA01.ZYQ00001.Y00003D2:340259    "/Path/to/the/file"
  $DATA01.ZYQ00001.Y00003D2:340259    "/Another/Path/to/the/file"
  $DATA01.ZYQ00001.Y00003D2:340259    "/Still/another/path"
```

LOG

LOG コマンドはセッションのログ記録を開始および停止します。ログには、入力されたすべてのコマンドとそれらの出力が含まれます。

LOG の構文

```
LOG { TO filename | STOP }
```

TO *filename*

指定したファイルへのセッションのログ記録を開始します。ログ記録がすでに進行している場合は、MEASCOM はそのログファイルを閉じ、*filename* を開き、新しく開かれたファイルにセッションのログを記録します。

filename はログファイルの名前です。*filename* がすでに存在している場合、データは既存のファイルに追加されます。*filename* が存在しない場合、MEASCOM が作成します。

STOP

セッションのログ記録を停止し、ログファイルを閉じます。

LOG の使用上の注意

ログ記録は MEASCOM コマンド出力に影響せず、OUT コマンドはログファイルの内容には影響しません。

LOG の例

この例では、セッションのログ記録を開始します。ENV コマンドは、現在のログファイルを含む環境情報を表示します。

```
+ LOG TO $DATA.MEAS.LOG04
+ ENV
  System \BUYER
  Volume $DATA.MEAS
  Log to $DATA.MEAS.LOG04
  Assume MEASUREMENT
```

OBEY

OBEY コマンドは、指定したファイルから MEASCOM コマンドを読み取ります。

OBEY の構文

```
OBEY filename [ , NOECHO ]
```

filename

MEASCOM コマンドを含むコマンドファイルの名前です。

NOECHO

コマンドファイルがコマンドを出力ファイルにエコーするのを禁止します。

OBEY の使用上の注意

- 「コマンドファイル」と「OBEY ファイル」という用語は、多くの場合同じ意味で使われます。
- 最大 4 つのコマンドファイルをネストすることができます。MEASCOM は、プライマリコマンドファイルの終わりに達すると、最初の OBEY コマンドが入力されたデバイスからのコマンドの読み取りを再開します。ネストされているファイルには、異なるエコーオプションを指定することができます。
- MEASCOM のほとんどのコマンド出力の形式は、コマンドファイルの正しい形式です。OUT コマンドを使用して、または MEASCOM の一部のコマンドで OUT オプションを使用して、ファイルに出力を送ることができます。その後、少し編集するだけで、そのファイルをコマンドファイルとして使用することができます。

OBEY の例

この例は、コマンドファイルの実行を示しています。OBEY コマンドの後に続くコマンドとコメントは、DAILY という名前のコマンドファイルから読み取られたものです。

```
+ OBEY DAILY
+ -- begin obey file
+ Add cpu 7
+ Add process *
+ Add file $*.*.ACCOUNT
+ -- end obey file
```

OSSPATH

Measure G09 以降の PVU では、OSSPATH コマンドは、OSS ファイルパス名を展開するときに使用するデフォルトのディレクトリを指定します。スラッシュ（/）で始まらない OSS ファイルパス名が MEASCOM で見つかると、OSSPATH の現在の設定を使用して展開されます。

OSSPATH の構文

```
OSSPATH "directory"
```

"*directory*"

デフォルトのディレクトリとして使用するディレクトリを識別する完全修飾の OSS ファイルパス名です。後続の OSS ファイルパス名のうちスラッシュ（/）で始まらないものは、完全修飾の OSS ファイルパス名を形成するためにデフォルトのディレクトリ名に追加されます。デフォルト値はスラッシュ（/）です。

OSSPATH の例

次の例は、OSSPATH コマンドを使用してデフォルトのディレクトリを宣言したときの出力を示しています。

```
+ OSSPATH "/Path/to/the/SourceFiles"
+ LISTGNAME "Server"
  $DATA01.ZYQ00001.Y000019F.340568 "/Path/to/the/SourceFiles/Server"
```

OUT

OUT コマンドは、指定されたファイルにコマンド出力を送ります。OUT コマンドを使用すると、MEASCOM コマンド出力に基づいてコマンド（OBEY）ファイルを作成することができます。たとえば、INFO MEASUREMENT コマンドの出力を使用して、少し編集するだけで、測定を再実行することができます。

OUT の構文

```
OUT filename
```

filename

リスニングファイルの名前です。*filename* がすでに存在している場合、コマンド出力はそのファイルに追加されます。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに送信します。

OUT の使用上の注意

コマンド出力がすでに別のファイルに送信されている場合は、そのファイルが閉じてから新しい出力ファイルが開きます。

OUT の例

次の例は、ファイル$DATA.MEAS.OUT04 にコマンド出力を送ります。後続のコマンド（この例の ENV コマンドなど）からの出力は、端末には表示されません。

```
+ OUT $DATA.MEAS.OUT04
+ ENV
```

PAGESIZE

Measure G09 以降の PVU では、PAGESIZE コマンドは、二重プロンプト（++）の前に表示される出力行（プロンプトを含む）の数を指定します。エンティティレポート形式の多くは、標準の 24 行×80 カラムの画面の表示領域を越えます。OSS ファイルパス名の表示ではテキストを複数の行に折り返すことができるため、多くのエンティティレポート形式の長さはさまざまです。読む前にデータが表示領域の外にスクロールされないようにするには、PAGESIZE コマンドを使用して二重プロンプト（++）を設定します。

MEASCOM も、レポートのグループが表示されるときに、個々のレポートの間に二重プロンプト（++）を表示します。レポートの間の二重プロンプト（++）は、PAGESIZE の設定の影響を受けることはなく、引き続きレポートの間に表示されます。

PAGESIZE の構文

```
PAGESIZE [ integer ]
```

integer

6～128 の値です。指定すると、「integer - 1」行のレポートが二重プロンプト（++）の後に表示されます。値を設定しない場合は、デフォルト値の 128 が使用され、二重プロンプト（++）は個々のレポートの間にのみ表示されます。

PAGESIZE の使用上の注意

- レポートデータの表示を続行するには、ENTER キーを押します。
- 表示を終了してメインの MEASCOM プロンプトに戻るには、BREAK キーまたは Ctrl-Y を押します。
- スプーラーまたはディスク出力ファイルにリストされるレポートには、PAGESIZE コマンドは効力がありません。
- Measure は、このコマンドに対して以下の警告またはエラーメッセージを表示することがあります。

```
3053 Value out of range; value must be 6:128.
```

PAGESIZE の例

標準の 24 行×80 文字の表示領域の外にデータがスクロールされないようにするには、次のコマンドを使用します。

```
+ PAGESIZE 24
```

RESET PLOT

RESET PLOT コマンドは、1 つまたは複数のプロットオプションをデフォルト値にリセットします。

注記:

RESET PLOT コマンドは属性を受け入れますが、値は受け入れません。プロットをデフォルト以外の値に設定するには、**SET PLOT**(133 ページ)を参照してください。

RESET PLOT の構文

```
RESET [ PLOT ] { * | plot-attribute [ , plot-attribute ] ... }
```

PLOT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして PLOT を指定した場合はオプションです。

plot-attribute

次のいずれかです。

FOR

プロットウィンドウの持続時間をリセットして、測定終了の日付と時刻を延長します。

FROM

プロットウィンドウの開始を測定開始の日付と時刻にリセットします。

SCALE-FROM

データ値軸の下限を 0 にリセットします。

SCALE-TO

データ値軸の上限を 100 にリセットします。

TIME-BASE

TIME-BASE 属性をデフォルト値にリセットします。データを収集したときに時間間隔を指定した場合、デフォルトは TIME-BASE ON です。指定しなかった場合、デフォルトは TIME-BASE OFF です。

TO

プロットウィンドウの終了を測定終了の日付と時刻にリセットします。

VERT-BASE

時間軸の表示を ON にリセットします。

WIDE-ITEM

プロットの密度を OFF にリセットします。

関連コマンド

コマンド	機能
ADD PLOT	プロットを指定します
DELETE PLOT	プロット記述からカウンターを削除します
LIST PLOT	プロットを表示します
SET PLOT	プロット属性を設定します

RESET REPORT

RESET REPORT コマンドは、1 つまたは複数のレポートオプションをデフォルト値に設定します。

注記:

RESET REPORT コマンドは属性を受け入れますが、値は受け入れません。レポートをデフォルト以外の値に設定するには、**SET REPORT**(137 ページ)を参照してください。

RESET REPORT の構文

```
RESET [ REPORT ] { * | report-attribute [ , report-attribute ] ... }
```

REPORT

ASSUME コマンドを使用して REPORT をコマンドオブジェクトとして指定した場合はオプションです。

report-attribute

次が含まれます。

COUNT

表示される選択したレコードの数をすべてのレコードにリセットします。

CR-NAME-LEN

プロシージャー（code-range）名の表示される長さを SHORT にリセットします。

CR-NAME-FORM

デマングルされたプロシージャー（code-range）名を STANDARD にリセットします。

CR-NAME-QUAL

プロシージャー（code-range）名を UNQUALIFIED にリセットします。

DOTS

ラベルと数値の間に接続のドットを含めないようにレポートの表示をリセットします。

FIRST

表示される開始レコード番号を 1 にリセットします。

FOR

レポートウィンドウの持続時間をリセットして、測定終了の日付と時刻を延長します。

FORMAT

レポートの形式を NORMAL にリセットします。

FROM

レポートウィンドウの開始を測定開始の日付と時刻にリセットします。

LOADID

構造化レコードの *loadid* フィールドに配置する名前を未指定としてリセットします。

RATE

カウンター値の表示を ON にリセットします。

RECORD-NUM

レコード番号の表示を OFF にリセットします。

STYLE

G シリーズの RVU ではデフォルトを LEGACY にリセットし、H シリーズの RVU ではデフォルトを ZMS にリセットします。

TO

レポートウィンドウの終了を測定終了の日付と時刻にリセットします。

TOTALS

TOTALS の表示を SUPPRESS にリセットします。

ZERO-REPORTS

カウンター値がすべてゼロのレポートの表示を SUPPRESS にリセットします。

ZERO-VALUES

値が 0.005 より小さい場合は空白を表示する（SUPPRESS）ように表示をリセットします。

ZERO-VALUES 属性は、構造化ファイルには影響がありません。

関連コマンド

コマンド	機能
SET REPORT	レポート形式のオプションを設定します

RUN

RUN コマンドでは、Measure を終了せずに別のプロセスを実行することができます。MEASCOM の RUN コマンドは TACL の RUN コマンドに似ています。

RUN の構文

```
RUN [ / OUT out-file / ] run-file [ / options / ]
    [ "parameters" ]
```

out-file

RUN コマンドによって生成されるエラーメッセージの出力ファイルです。

run-file

実行するプログラムファイルです。

options

実行オプションです。一般的に使用される実行オプションは、次のとおりです。

IN *filename*	実行するプロセスの入力ファイルを指定します。
OUT *filename*	実行するプロセスの出力ファイルを指定します。
NOWAIT	RUN コマンドを実行した直後に MEASCOM プロンプトに戻るようにします。NOWAIT を使用しない場合、Measure は *run-file* プロセスが完了するまで一時停止します。

Measure は TACL のすべての実行オプションをサポートするわけではありません。**RUN の使用上の注意**(132 ページ)を参照してください。

"*parameters*"

新しく作成されたプロセスに渡されるパラメーターまたはパラメーターの文字列です。パラメーターは、二重引用符（" "）で囲む必要があります。

RUN の使用上の注意

- RUN コマンドの基本情報とそのオプションの詳細な説明については、*TACL Reference Manual* を参照してください。
- ボリュームやサブボリュームのないプログラム名（たとえば、RUN FUP）を指定すると、Measure は現在のサブボリュームをチェックします。ファイルが見つからない場合やオブジェクトファイルではない場合、Measure は$SYSTEM.SYSTEM および$SYSTEM.SYS *nn* をチェックします。リモートシステムを指定した場合（たとえば、RUN \REMOTE.SURVCOM）、Measure はリモートシステムの $SYSTEM.SYSTEM and $SYSTEM.SYS *nn* をチェックします。
- Measure は特定の TACL オプションを完全にサポートしているわけではありません。

- INSPECT および DEFMODE では、OFF および ON キーワードのみがサポートされます。
- RUN オプションの HIGHPIN および EXTSWAP はサポートされません。
- RUND コマンドは、直接はサポートされません。DEBUG を使用するには、次のコマンドを入力します。

  RUN *program-file* / DEBUG /

RUN の例

MEASCOM プロンプトから FUP INFO を実行するには、次のコマンドを使用します。

+ **RUN FUP INFO MYSUBVOL.***

SET PLOT

SET PLOT コマンドは、新しい SET PLOT または RESET PLOT コマンドによって変更されるまでは、後続のプロットを制御する 1 つまたは複数のプロット属性を設定します。

SET PLOT の構文

SET [PLOT] *plot-attribute value* [, *plot-attribute value*] ...

PLOT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして指定した場合はオプションです。

plot-attribute value

次のいずれかです。

FOR *duration*

プロットウィンドウの持続時間です。FOR と TO を同時に使用することはできません。この属性は、新しい SET PLOT または RESET PLOT コマンドによって変更されるまでは、以降のプロットのデフォルトになります。

duration

次のいずれかの形式の時間間隔です。

n SECOND[S]
n MINUTE[S]
n HOUR[S]

指定する値は次の通りです。*n* は 1～9999 の範囲の整数です。

FROM { [*date* ,] *time-of-day* }

プロットウィンドウの開始を指定します。この属性は、新しい SET PLOT または RESET PLOT コマンドによって変更されるまでは、以降のすべてのプロットのデフォルトになります。FROM を省略すると、測定の開始時刻か、エンティティ測定の開始時刻のいずれか遅い方が使用されます。

date

プロットで使用されるデータの開始日で、次のいずれかの形式です。

{ [*d*]*d mmm*[*yyyy*] }
{ *mmm* [*d*]*d*[*yyyy*] }

指定する値は次の通りです。

dd は日付で、1～31 の範囲の数字です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT です。

yyyy は西暦です。有効な年は 1984～2047 です。

date を省略した場合、測定の開始日が使用されます。

time-of-day

プロットで使用されるデータの開始日で、次の形式です。

```
hh:mm[:ss]
```

指定する値は次の通りです。

hh は時（0～23）です。

mm は分（0~59）です。

ss は秒（0~59）です。

time-of-day を省略した場合、測定の開始時刻が使用されます。

SCALE-FROM *number*

データ値軸の下限を設定します。

number

0～999999999999.999 の範囲の数字です。デフォルトは 0 です。

SCALE-TO *number*

データ値軸の上限を設定します。

number

0～999999999999.999 の範囲の数字です。デフォルトは 100 です。

TIME-BASE { ON | OFF }

カウンター値の棒グラフまたは 2 軸プロットを表示します。データを収集したときに時間間隔を指定した場合、デフォルトは TIME-BASE ON です。指定しなかった場合、デフォルトは TIME-BASE OFF です。

ON

カウンター値のプロットを時系列で表示します。

OFF

カウンター値の平均を棒グラフで表示します。

TO [*date* ,] *time-of-day*]

プロットウィンドウの終了を指定します。TO と FOR を同時に使用することはできません。この属性は、新しい SET PLOT または RESET PLOT コマンドによって変更されるまでは、以降のプロットのデフォルトになります。

date

プロットの終了日で、次のいずれかの形式です。

```
{ [d]d mmm[ yyyy] }
{ mmm [d]d[ yyyy] }
```

指定する値は次の通りです。

dd は日付で、1～31 の範囲の数字です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT です。

yyyy は西暦です。有効な年は 1984～2047 です。

date を省略した場合、測定の終了日が使用されます。

time-of-day

プロットで使用されるデータの終了日で、次の形式です。

hh:*mm*[:*ss*]

指定する値は次の通りです。

hh は時（0～23）です。

mm は分（0~59）です。

ss は秒（0~59）です。

time-of-day を省略した場合、測定の終了時刻が使用されます。

VERT-BASE { ON | OFF }

時間軸の表示を制御します。デフォルトは ON です。

ON

2 軸プロットでは、時間を縦軸に表示します。棒グラフでは、（縦軸を基準にして）棒を横に表示します。

OFF

2 軸プロットでは、時間を横軸に表示します。棒グラフでは、（横軸を基準にして）棒を縦に表示します。

WIDE-ITEM { ON | OFF }

プロットの密度を設定します。デフォルト設定は、OFF です。

ON

2 軸プロットでは、時間軸と最も小さいカウンター値の間の領域を埋めます。棒グラフでは、グラフ上のカウンター値の数に応じて、2～6 文字の幅の棒を表示します。

OFF

2 軸プロットでは、各時間間隔でカウンター値ごとに 1 つのプロット文字を表示します。棒グラフでは、1 文字の幅の棒を表示します。

関連コマンド

コマンド	機能
ADD PLOT	プロットを指定します
LIST PLOT	プロットを表示します

SET PLOT の例

次の例は、2 軸プロットと棒グラフを表示します。両方とも同じデータを使用して行われました。

```
+ SET PLOT SCALE-TO 8
+ SET PLOT VERT-BASE OFF
+ SET PLOT WIDE-ITEM ON
+ LIST PLOT, INTERVAL 1 MINUTE
```

```
    7.99 -----------------
    7.62 -                 -
    7.26 -                 -
    6.90 -                 -
    6.53 -                 -
    6.17 -                 -
    5.81 -    B            -
    5.45 -                 -
    5.08 -  B              -
    4.72 -                 -
    4.36 -                 -
    3.99 -                 -
    3.63 -B A A            -
    3.27 -  * *            -
    2.90 -  * *   B        -
    2.54 -A * *            -
    2.18 -* * *   A     B  -
    1.82 -* * *   *   B    -
    1.45 -* * * B *        -
    1.09 -* * *   * B A    -
    .726 -* * * A *   *    -
    .363 -* * * * * A * A  -
       0 -*-*-*-*-*-*-*-A-
   11:43:14  45  47  49  51
           44  46  48  50

       Min Value = 0          Max Value = 5.86
     A: READ-BUSY-TIME        Disk $BOOKS1 (6, 1, 17, 1)
     B: READ-BUSY-TIME        Disk $BOOKS1 (6, 1, 17, 2)
```

```
+ SET PLOT TIME-BASE OFF
+ LIST PLOT, INTERVAL 1 MINUTE
```

```
    7.99 ---------------------------------------
    7.62 -                                      -
    7.26 -                                      -
    6.90 -                                      -
    6.53 -                                      -
    6.17 -                                      -
    5.81 -                                      -
    5.45 -                                      -
    5.08 -                                      -
    4.72 -                                      -
    4.36 -                                      -
    3.99 -                                      -
    3.63 -                                      -
    3.27 -                                      -
    2.90 -       -----                        -
    2.54 -      |     |                         -
    2.18 -      |     |                         -
    1.82 - -----|     |                        -
    1.45 -|     |     |                         -
    1.09 -|     |     |                         -
    .726 -|     |     |                         -
    .363 -|     |     |                         -
       0 -|     |     |-------------------------
          A     B
```

```
   Min Value = 1.80        Max Value = 2.92
A: READ-BUSY-TIME          Disk $BOOKS1 (6, 1, 17, 1)
B: READ-BUSY-TIME          Disk $BOOKS1 (6, 1, 17, 2)
```

SET REPORT

SET REPORT コマンドは、新しい SET REPORT コマンド、RESET REPORT コマンドによって変更されるか、新しいデータファイルを追加するまで、後続のレポートの形式を制御します。

Enform を使用して ENDPOINT の構造化ファイルまたは ZMS 形式の SERVERNET エンティティを調べるには、HPE NonStop X システムからの測定の場合は ZMSENDPT 構造体を使用し、以前のシステムからの測定の場合は ZMSSVNET 構造体を使用します。ZMSSVNET 構造体は L シリーズの測定ファイルに使用されることもありますが、NonStop X システムでは使用されないフィールドが表示されます。

SET REPORT の構文

```
SET [ REPORT ] report-attribute value [ , report-attribute value ] ...
```

REPORT

ASSUME コマンドを使用して REPORT をデフォルトのコマンドオブジェクトとして指定した場合はオプションです。

report-attribute value

次が含まれます。

CR-NAME-LEN { SHORT | LONG }

表示されるプロシージャー（code-range）名前の長さを指定します。デフォルトは SHORT です。

SHORT

プロシージャー（code-range）名を 32 文字で切り捨てます（必要な場合）。プロシージャーのマングル名は、表示前にデマングルされません。

LONG

プロシージャー（code-range）名を SHORT 形式で表示し、32 文字より長い場合は次の行に全体を表示します。プロシージャーのマングル名は、表示用にデマングルされません。

CR-NAME-FORM { STANDARD | DEMANGLED | BOTH }

デマングルされたプロシージャー（code-range）名を表示するかどうかを指定します（該当する場合）。デフォルトは STANDARD です。

STANDARD

プロシージャー（code-range）名を、コードファイル（マングル）または EDIT ファイル（デマングル）で指定された形式で表示します。

DEMANGLED

必要に応じて、表示する前に、プロシージャー（code-range）名をデマングルします。

BOTH

プロシージャー（code-range）名を、STANDARD と DEMANGLED の両方の形式で表示します。

CR-NAME-QUAL { UNQUALIFIED | QUALIFIED }

プロシージャー（code-range）名を、可能な場合はコード範囲出力の従来の行の直後の行の「Code-Range Name」列に、オブジェクトファイル名修飾子を付けて表示するかどうかを指定します。デフォルトは UNQUALIFIED です。

UNQUALIFIED

修飾子のない Legacy 形式でプロシージャー名を表示します。

QUALIFIED

関連付けられたコードの Guardian または OSS オブジェクトファイル名とともにプロシージャー名を表示します。この機能は、さまざまなオブジェクトファイル内にある似た名前のプロシージャーを区別する場合に役立ちます。Guardian ファイル名では、関連付けられた CRVSN が名前の末尾に付加されています。

COUNT *record-count*

表示する選択したレコードの数（10 進数）を指定します。たとえば、FIRST が 24 で COUNT が 6 の場合、レコード 24 から 29 が表示されます。指定しない場合、デフォルトはすべてのレコードです。指定した場合、値は正の整数である必要があります。

DOTS { ON | OFF }

レポートの表示で、ラベルと数値の間に接続のドットを含める必要があるかどうかを指定します。STYLE が ZMS に設定されている場合にのみ有効です。デフォルト設定は、OFF です。

ON

フォーマットされた数値にラベルを接続するドットを表示します。

OFF

フォーマットされた数値にラベルを接続するドットを表示しません。

FIRST *record-num*

開始レコード番号（10 進数）を指定します。指定しない場合、デフォルトは 1 です。指定した場合、値は正の整数である必要があります。

FOR *duration*

レポートウィンドウの持続時間です。FOR と TO を同時に使用することはできません。

duration

次のいずれかの形式の時間間隔です。

```
n SECOND[S]
n MINUTE[S]
n HOUR[S]
```

指定する値は次の通りです。*n* は 1～9999 の範囲の整数です。

FORMAT { BRIEF | CSV | JSON | NORMAL | STRUCTURED | XML }

レポートの形式を設定します。デフォルトは NORMAL です。

BRIEF

カウンターのサブセットを表示します。このセクションにリストされる DDL レコードは、エンティティタイプごとのカウンターをリストします。BRIEF サブセットに属するカウンターは太字で表示されます。PROCESSH エンティティでは、TOTALS INCLUDE または TOTALS ONLY も指定した場合は、コード範囲レベルの合計のみ（プロセスごとの合計ではなく）が表示されます。

簡略形式のレポートに含まれるカウンターについて詳しくは、**エンティティおよびカウンター** (167 ページ)で、指定されたエンティティタイプの DDL レコードを参照してください。

CSV

エンティティタイプに基づいた名前と CSV のサフィックスを持つ「C」ディスクファイル（タイプ 180）に、コマンドからの出力を CSV 形式で書き込みます。ファイルの最大サイズは 1TB です。ファイルがすでに存在している場合、MEASCOM はそのファイルを上書きします。ファイルが存在しない場合、MEASCOM が作成します。

JSON

エンティティタイプに基づいた名前と JSN のサフィックスを持つ「C」ディスクファイル（タイプ 180）に、コマンドからの出力を JSON 形式で書き込みます。ファイルの最大サイズは 1TB です。ファイルがすでに存在している場合、MEASCOM はそのファイルを上書きします。ファイルが存在しない場合、MEASCOM が作成します。

NORMAL

すべてのカウンターを表示します。

STRUCTURED

Enform を使用した後の調査のために構造化ファイルにレポートを書き込みます。1 つのレコードは、単一のエンティティのすべてのカウンターから構成されます。同じエンティティタイプのエンティティに関するレポートは同じファイルに書き込まれ、そのファイルはエンティティタイプに応じた名前が付けられます。ファイルがすでに存在する場合、MEASCOM がデータをそのファイルに追加します。ファイルが存在しない場合、MEASCOM が作成します。（FORMAT 属性を変更すると構造化ファイルは閉じられます）。

Measure は、要求側のアプリケーションが使用するテンプレートバージョンと一致するようにレコードの形式を調整することで、将来のリリースも含むさまざまなリリースからのレコードを処理します。

構造化レコードでは、以前の構造化ファイルが存在する場合、ファイルの最初のレコードはテンプレートバージョンを確認するために使用され、構造化ファイルに追加される新しいレコードはそのテンプレートバージョンを使用します。実際には、データファイルはインストールされているシステムよりも新しいバージョンである可能性があり、Measure が引き続きレコード形式を判断し、「将来」のテンプレートバージョンに合わせて正しい場所に関連するカウンターを書き込む可能性があることに注意してください。「将来」のテンプレートバージョンを使用するには、目的のテンプレートバージョンの「ダミー」レコードを空のファイルに挿入することで、以前のファイルを作成します。その後、レコードに"SKIPTHIS"や"IGNOREIT"などの一意な LOADID を割り当てることによって、実際のファイルの処理中はこの「ダミー」レコードを無視するようにアプリケーションに指示します。この例では、より新しいテンプレートバージョンの「将来」の値はゼロで埋められます。同じ機能が、Measure の新しいカウンターをサポートするために更新されていてもまだ古いシステムで実行されているアプリケーションにも当てはまります。古いシステム上で実行する場合、Measure インターフェイスは、レコードをアプリケーションが要求する新しいテンプレートバージョンの形式に調整します。

構造化ファイルが存在していても空の場合、構成されたファイルのレコード長は、現在または過去のテンプレートバージョンとのみ照合されます。MEASCOM では、テンプレートバージョンはインストールされているシステムのバージョンと照合されます。

このテンプレートバージョン機能により、Measure は、さまざまなリリースや Measure バージョンの複数のシステムから取得したデータファイルからのレコードを、共通のレコード形式を持つ 1 セットの ZMS 構造化ファイル内に配置できます。さらに、アプリケーションは、データを収集した Measure のバージョンの形式に関係なく、アプリケーションをコンパイルした形式で Measure レコードを要求および処理できます。アプリケーションで更新された形式の新しい情報を利用したい場合を除き、Measure レコード形式を最新の状態に保つためにアプリケーションを再コンパイルする必要はありません。

XML

エンティティタイプに基づいた名前と XML のサフィックスを持つ「C」ディスクファイル（タイプ 180）に、コマンドからの出力を XML 形式で書き込みます。ファイルの最大サイズは 1TB

です。ファイルがすでに存在している場合、MEASCOM はそのファイルを上書きします。ファイルが存在しない場合、MEASCOM が作成します。

FROM [*start-date* ,] *start-time-of-day*]

レポートウィンドウの開始を指定します。FROM を省略すると、測定の開始時刻か、エンティティ測定の開始時刻のいずれか遅い方が使用されます。

start-date

レポートで使用されるデータの開始日で、次のいずれかの形式です。

```
{ [d]d mmm[ yyyy] }
{ mmm [d]d[ yyyy] }
```

指定する値は次の通りです。

dd は日付で、1～31 の範囲の数字です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT です。

yyyy は西暦です。有効な年は 1984～2047 です。

start-date を省略した場合、測定の開始日が使用されます。

start-time-of-day

レポートで使用されるデータの開始時刻で、次の形式です。

```
hh:mm[:ss]
```

指定する値は次の通りです。

hh は時（0～23）です。

mm は分（0~59）です。

ss は秒（0~59）です。

start-time-of-day を省略した場合、測定の開始時刻が使用されます。

LOADID *loadid*

構造化レコードの *loadid* フィールドに配置する名前を指定します。

loadid

1～8 文字の長さの英数字文字列です。文字列には、英字、数字、キャレット（^）、ハイフン（-）、およびアンダースコア（_）を含められます。最初の文字は、英字でなければなりません。

RATE { ON | OFF }

カウンター値を表示する方法を指定します。RATE 属性は、構造化ファイルには影響がありません。デフォルトは ON です。

ON

解釈されたカウンター値（1 秒あたりの数とビジー率）を表示します。

OFF

解釈されていないカウンター値を表示します。

RECORD-NUM { ON | OFF }

各レコードのレコード番号を表示するかどうかを指定します。デフォルト設定は、OFF です。

ON

各レコードのレコード番号を表示します。

OFF

各レコードのレコード番号を表示しません。

STYLE { LEGACY | ZMS }

表示および構造化データのフォーマットを、ZMS 形式の外部インターフェイス（Measure G11 以降の PVU）を使用して行うか、Measure G11 より前の PVU と互換性のある外部インターフェイスを使用して行うかを指定します。G シリーズ以前の RVU ではデフォルトは LEGACY で、H シリーズ以降の RVU ではデフォルトは ZMS です。

LEGACY

表示および構造化レコードで Measure G11 より前の PVU 形式を使用します。

ZMS

表示および構造化レコードで、Measure G11 で導入された形式を使用します。

TO [*end-date* ,] *end-time-of-day*]

レポートウィンドウの終了を指定します。TO と FOR を同時に使用することはできません。

end-date

レポートの終了日で、次のいずれかの形式です。

```
{ [d]d mmm[ yyyy] }
{ mmm [d]d[ yyyy] }
```

指定する値は次の通りです。

dd は日付で、1～31 の範囲の数字です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT です。

yyyy は西暦です。有効な年は 1984～2047 です。

end-date を省略した場合、測定の終了日が使用されます。

end-time-of-day

レポートで使用されるデータの終了時刻で、次の形式です。

```
hh:mm[:ss]
```

指定する値は次の通りです。

hh は時（0～23）です。

mm は分（0~59）です。

ss は秒（0~59）です。

end-time-of-day を省略した場合、測定の終了時刻が使用されます。

TOTALS { INCLUDE | ONLY | SUPPRESS }

TOTALS を表示するかどうかを指定します。デフォルトは SUPPRESS です。

INCLUDE

プロセスごとの合計と総合計の両方が表示されることを示します。総合計が表示されるときに、一連の合計は大カッコ（[…]）で囲まれ、総計されているコードを実行していたプロセス数

が個々のコード範囲の結果の前に置かれ、これも同様に大カッコで囲まれます（つまり[Totals across 3 processes]）。

ONLY

PROCESSH および USERDEF を除くすべてのエンティティについて、最終の TOTALS レポートのみを表示します。1 つのエンティティレポートのみを生成した場合は、TOTALS レポートは表示されません。

PROCESSH エンティティの場合は、集計した PROCESSH データのみが表示されます。

注記:

PROCESSH 測定が（ADD PROCESSH コマンドで）構成されたときにコードファイル指定が与えられたかどうかに応じて、コードファイルごとか、プロシージャーごとのどちらかに基づいて PROCESSH データを収集できます。したがって、表示された合計は、コードファイル全体を含む場合も、コードファイル内の一連のコード範囲それぞれを含む場合もあります。

SUPPRESS

エンティティレポートのみを表示します （レポートに 1 つのエンティティの測定のみが含まれている場合、TOTALS 属性が無視されます）。

PROCESSH の場合、SUPPRESS はプロセスごとのデータだけが表示されることを指定します。（集計されたデータは表示されません）。

SUPPRESS は USERDEF レポートには影響しません。

ZERO-REPORTS { INCLUDE | SUPPRESS }

ゼロの値のみが含まれるレコードが表示されるかどうかを指定します。デフォルトは SUPPRESS です。

INCLUDE

すべてのカウンターの値がゼロの場合でも、エンティティレコードを表示します。

SUPPRESS

すべてのカウンターの値がゼロの場合、エンティティレコードを表示しません。

注記:

スナップショットカウンターは、特定の時点で取得された測定を返します。ゼロ以外のスナップショットカウンターの値は、必ずしも測定間隔中のアクティビティを示すわけではありません。スナップショットカウンターの値は通常、ZERO-REPORTS 句のレコードを評価するときに考慮されません。

ZERO-VALUES { INCLUDE | SUPPRESS }

0.005 未満の値をゼロまたは空白として表示するかどうかを指定します。

INCLUDE

カウンターの値が 0.005 未満の場合にゼロを表示します。

SUPPRESS

カウンターの値が 0.005 未満の場合に空白を表示します。

ZERO-VALUES 属性は、構造化ファイルには影響がありません。デフォルトは SUPPRESS です。

*****

レポートのすべての属性をデフォルト値にリセットします。

関連コマンド

コマンド	機能
ADD MEASUREMENT	レポートを指定します
LIST *entity-type*	選択したエンティティに関するレポートを表示します
LISTACTIVE *entity-type*	アクティブなエンティティに関するレポートを表示します
LISTALL *entity-type*	間隔レコードによって選択したエンティティに関するレポートを表示します

SET REPORT の例

- すべての CPU レポートを簡単な形式で表示し、レポート全体のカウンター値を合計して、合計したカウンター値が含まれている最終レポートを生成するには、次のコマンドを使用します。

```
+ SET REPORT FORMAT BRIEF, TOTALS INCLUDE
+ LIST CPU *

Cpu  6 VLX          Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16        PCBs    100               Page Size  2048
Local System \SAIID From 7 Aug 1994, 14:53:01 For 75 Seconds

Cpu-Busy-Time       17.86%    Swaps                0.03
Ending-Free-Mem               Ending-UCME
++
Cpu  7 VLX          Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16        PCBs    100               Page Size  2048
Local System \SAIID From 7 Aug 1994, 14:53:01 For 75 Secs

Cpu-Busy-Time        2.22%    Swaps
Ending-Free-Mem               Ending-UCME
++
Totals for     2 Records
From   7 Aug 1994, 14:53:01    For    75 Seconds

Cpu-Busy-Time       20.07%    Swaps                0.03
Ending-Free-Mem               Ending-UCME
```

- レポート全体を表示し、ビジー率と割合ではなくビジーのカウントと秒としてカウンター値を表示するには、次のコマンドを使用します。

```
+ SET REPORT FORMAT NORMAL, RATE OFF
+ LIST CPU 6

Cpu  6 VLX          Initial Lock Pgs 2048     Mem Pages  8192
Memory MB 16        PCBs    100               Page Size  2048
Local System \SAIID From 7 Aug 1994, 14:53:01 For 75 Seconds

Cpu-Busy-Time       13.52s    Swaps                 2#
Cpu-Qtime           18.56s    Cpu-Qlen-Max         14#
Mem-Qtime           82.42m    Mem-Qlen-Max          5#
Dispatches           3,297#    Intr-Busy-Time     2.49s
Process-Ovhd                  Send-Busy-Time    46.65m
Disc-IOs               143#   Cache-Hits            67#
```

```
Transactions                    Response-Time
Page-Requests                   Page-Scans
Ending-Free-Mem      4092       Ending-UCME
Ending-UDS            500       Ending-SDS           1500
Ending-UCL            100       Ending-SCL           2000
```

- Measure G11 以降の PVU で利用可能な ZMS 形式を使用してレポートを表示するには、次のコマンドを使用します。

```
+ SET REPORT STYLE ZMS
+ LIST CPU 3, RATE OFF

Cpu  3 NSR-T          Init Lock Pgs    269          Mem Pages   131072
Memory MB  2048       PCBs            2200          Pg Size    16384  Bytes
Format Version:  G11  Data Version:  G09  Subsystem Version:   1
Local System \MEASURE From  27 Mar 2002,  5:55:53   For    47 Minutes
------------ Processor
-----------------------------------------------------
Cpu-Busy-Time                1.94 sec  Dispatches                  3,647 #
Cpu-Qtime                    4.41 sec  Intr-Busy-Time             123.95 ms
Process-Ovhd               155.45 ms   Send-Busy-Time
Comp-Traps                   1,298 #
Native-Busy-Time             1.63 sec  Accel-Busy-Time            165.03 ms
TNS-Busy-Time               18.22 ms   PROCESSH-Samples
------------ Memory
-----------------------------------------------------------
Starting-Free-Mem         111,999 #    Ending-Free-Mem           128,999 #
Swaps                       1,999 #    Page-Requests               3,999 #
MM-Page-Scans               3,999 #    Page-Scans                 14,995 #
Unsp-Pages-Qtime          499,999 sec  Mem-Qtime
Unsp-Pages-Start               14 #    Unsp-Pages-End                 14 #
Starting-SCL                29,999 #    Ending-SCL                 7,999 #
Starting-UCL                25,999 #    Ending-UCL                 1,999 #
Starting-Free-CIDs          31,999 #    Ending-Free-CIDs           31,999 #
Ending-UCME
------------ Device I/O
-----------------------------------------------------
Disc-IOs                        4 #    Cache-Hits                   141 #
Transactions                           Response-Time
------------ Messaging
-----------------------------------------------------
Link-Prepush-Msgs       7,989,999 #    Readlinkcache-All        7,289,999 #
Link-Readlink-Msgs        102,999 #    Readlinkcache-Ctrl      22,959,999 #
Link-Large-Msgs           102,999 #    Readlinkcache-None
                                       Replyctrlcache-Msgs      7,289,999 #
------------ ServerNet Transfers Initiated by this Local Processor
----------
to/fm     READS INTO CPU     READ BYTES    WRITES FROM CPU    WRITE BYTES
CPU-0              458 #        337,750 #            452 #        195,478 #
CPU-1
                  .
                  .
                  .
CPU-15
See TOTAL for sum over all initiating CPUs: total data into & from each
CPU.
```

- 書式設定された数字にラベルを接続するドットを使用してレポートを表示するには、次のコマンドを使用します。

```
+ SET REPORT STYLE ZMS
+ SET REPORT DOTS ON
+ LIST DISKFILE *

Diskfile &D0TAE.SAFE.GUARD                          File Format 1
Device Name $D10D                                   Pool
Local CPU 2          File Type  Key Sequenced       File Code 889
Format Version:  G11  Data Version:  G09  Subsystem Version:   1
Local System \SYSTEM  From  27 Mar 2002,  5:55:53   For  69.9 Seconds
------------ File
-------------------------------------------------------------
Open-Qtime ............... 999799 sec  Transient-Opens ............. 238 #
Ending-EOF ............... 45,989 B    File-Growth-Ratio .......... 1.00
Extent-Allocations
------------ Logical I/O
-------------------------------------------------
Lockwait-Time ............... 3.14 sec  Requests .................. 2,999 #
Max-Lockwait-Time .......... 3.14 sec  Requests-Blocked
Lock-Timeouts                          Lock-Bounces
Cache-Read-Hits ........... 2,999 #    Cache-Write-Hits .......... 2,479 #
Block-Splits                           Cache-Write-Cleans
Driver-Input-Calls ........ 3,999 #    Driver-Output-Calls ........2,999 #
DBIO-Input-Calls ............ 178 #    DBIO-Output-Calls ............ 29 #
SQL-Inserts ................... 1 #    SQL-Deletes
SQL-Updates                            SQL-Ending-Rows ............... 1 #
OSS-Cache-Callbacks                    OSS-Callback-Writes
OSS-Block-Writes                       OSS-Block-Write-Bytes
```

- この例は、G シリーズでは、完全、デマングル、および修飾のコード範囲名（LONG、DEMANGLED、QUALIFIED）を持つ収集された PROCESSH サンプルを表示します。

```
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.TCPIP)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.ZCRESRL)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.ZINETSRL
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.ZCRTLSRL)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.ZLANCSRL)
+ ADD PROCESSH $SYSTEM.SYS00.TCPIP ($SYSTEM.SYS00.TSYSCLR)
+ SET REPORT CR-NAME-LEN  LONG
+ SET REPORT CR-NAME-FORM DEMANGLED
+ SET REPORT CR-NAME-QUAL QUALIFIED
+ LIST PROCESSH *, RATE OFF, IF CODE-RANGE > 0
```

SETPROMPT

SETPROMPT コマンドは、TACL の SETPROMPT コマンドに似ています。SETPROMPT では、現在の仮定オブジェクトまたは現在のログファイルなどの環境情報を含めるように、MEASCOM 入力プロンプトを再定義できます。TACL の SETPROMPT コマンドについて詳しくは、*TACL Reference Manual* を参照してください。

SETPROMPT の構文

```
SETPROMPT { ASSUME   }
          { COMMENTS }
          { LOG      }
          { NONE     }
```

```
{ OUT       }
{ SYSTEM    }
{ SWAPVOL   }
{ VOLUME    }
{ WARNINGS }
```

ASSUME

現在の仮定オブジェクトを表示します。

COMMENTS

DISPLAY ALL または SUPPRESS ALL のうち、COMMENTS コマンドに対して現在アクティブな方を表示します。個別に表示または非表示にするコメントを確認するには、ENV コマンドを使用します。

LOG

現在のログファイルを表示します。

NONE

標準の MEASCOM プロンプト（行番号とプラス記号）を表示します。デフォルトは NONE です。

OUT

現在の出力ファイルを表示します。

SYSTEM

現在のシステム名を表示します。

SWAPVOL

現在のスワップボリュームを表示します。

VOLUME

現在のボリュームとサブボリューム名を表示します。

WARNINGS

DISPLAY ALL または SUPPRESS ALL のうち、WARNINGS コマンドに対して現在アクティブな方を表示します。個別に表示または非表示にする警告を確認するには、ENV コマンドを使用します。

SETPROMPT の使用上の注意

- MEASCOM プロンプトを現在アクティブでないパラメーターに設定すると、プロンプトに情報は表示されません。たとえば、ログ記録が有効でないときにプロンプトを LOG に設定すると、プロンプトにログファイル名は表示されません。ただし、ログを開始すると、プロンプトが現在のログファイル名に変更されます。
- Measure を終了すると、プロンプトはデフォルト設定（NONE）に戻ります。

SETPROMPT の例

- MEASCOM プロンプトに現在のボリューム名を表示するように設定するには、次のコマンドを使用します。

```
+ SETPROMPT VOLUME
$DATA MEAS
```

- MEASCOM プロンプトに現在の仮定オブジェクトを表示するように設定してから、仮定オブジェクトを変更するには、次のコマンドを使用します。

```
+ SETPROMPT ASSUME
MEASUREMENT
+ ASSUME PLOT
```

SHOW PLOT

SHOW PLOT コマンドは、1 つまたはすべてのプロット属性の設定を表示します。

SHOW PLOT の構文

```
SHOW [ / OUT filename / ] [ PLOT ] [ attribute ]
```

OUT *filename*

コマンド出力を *filename* に送ります。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

SHOW PLOT コマンドを実行した後、MEASCOM はその出力を現在の OUT ファイル（通常は端末）にリダイレクトします。この属性は、ログファイルの内容には影響しません。

PLOT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして PLOT を指定した場合はオプションです。

attribute

次のいずれかです。

```
FOR    SCALE-FROM    TIME-BASE    VERT-BASE
FROM   SCALE-TO      TO           WIDE-ITEM
```

これらの属性の説明については、**SET PLOT**(133 ページ)を参照してください。

attribute を省略した場合、MEASCOM はすべての属性を表示します。

関連コマンド

コマンド	機能
RESET PLOT	プロット形式のオプションをデフォルト値にリセットします
SET PLOT	プロット形式のオプションを設定します

SHOW PLOT の使用上の注意

- SHOW PLOT コマンドの出力は、プロット属性を設定するために使用したコマンドのリストです。後で現在の属性セットを復元するために使用できるコマンド（OBEY）ファイルを作成するには、OUT *filename*/を使用します。
- MEASCOM は、時間範囲に一致するレコードがなくなったことを検出すると、TIME-BASE ON で PLOT を終了します。

SHOW PLOT の例

すべてのプロット属性を表示するには、次のコマンドを使用します。

```
+ SHOW PLOT
  Set Plot Vert-Base  on
  Set Plot Time-Base
  Set Plot Scale-From       0.000
```

```
Set Plot Scale-To       100.000
Set Plot Wide-Item  off
Set Plot From
Set Plot To
Set Plot For
```

SHOW REPORT

SHOW REPORT コマンドは、1 つまたはすべてのレポート属性の設定を表示します。

SHOW REPORT の構文

```
SHOW [ / OUT filename / ] [ REPORT ] [ attribute ]
```

OUT *filename*

コマンド出力を *filename* に送ります。*filename* が存在しない場合、MEASCOM は EDIT ファイルをその名前で作成し、コマンド出力をそのファイルに書き込みます。*filename* が存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

SHOW REPORT コマンドを実行した後、MEASCOM はその出力を現在の OUT ファイル（通常は端末）にリダイレクトします。この属性は、ログファイルの内容には影響しません。

REPORT

ASSUME コマンドを使用してコマンドオブジェクトとして指定した場合はオプションです。

attribute

次のいずれかです。

COUNT	DOTS	FROM	STYLE	ZERO-VALUES
CR-NAME-FORM	FIRST	LOADID	TO	
CR-NAME-LEN	FOR	RATE	TOTALS	
CR-NAME-QUAL	FORMAT	RECORD-NUM	ZERO-REPORTS	

これらの属性の説明については、**SET REPORT**(137 ページ)を参照してください。

attribute を省略した場合、MEASCOM はすべての属性を表示します。

関連コマンド

コマンド	機能
RESET REPORT	レポート形式のオプションをデフォルト値にリセットします
SET REPORT	レポート形式のオプションを設定します

すべての PVU および RVU での SHOW REPORT の使用上の注意

SHOW REPORT コマンドの出力は、レポート属性を設定するために使用したコマンドのリストです。後で現在の属性セットを復元するために使用できるコマンド（OBEY）ファイルを作成するには、OUT *filename*/を使用します。

G シリーズの PVU での SHOW REPORT の使用上の注意

Measure G12 以降の PVU では、Measure は、TNS コードファイル（高速化 TNS コードファイルを含む）に対してのみコード領域識別子を表示します。

SHOW REPORT の例

- すべてのレポート属性を表示するには、次のコマンドを使用します。

```
+ SHOW REPORT
Set Report Format normal
Set Report Rate on
Set Report Totals suppress
Set Report Zero-Values suppress
Set Report Zero-Reports suppress
Set Report From
Set Report For
Set Report LoadId
Set Report CR-NAME-LEN  short
Set Report CR-NAME-FORM standard
Set Report CR-NAME-QUAL unqualified
```

- CR-NAME-LEN のみの現在の値を表示するには、次のコマンドを使用します。

```
+ SHOW REPORT CR-NAME-LEN
Set Report CR-NAME-LEN short
```

SQLCATALOG

Measure H01 以降の PVU では、SQLCATALOG コマンドは、部分修飾の ANSI SQL 名の展開で使用するデフォルトのカタログを指定します。

SQLCATALOG の構文

```
SQLCATALOG [ 'catalog-name' ]
```

catalog-name

ANSI SQL カタログ識別子です。*catalog-name* を指定しない場合は、デフォルトの空の値が使用されます。

SQLCATALOG の例

部分修飾の ANSI SQL 名を展開するためのデフォルトのカタログを指定するには、次のコマンドを使用します。

```
+ SQLCATALOG 'CATALOG_12'
```

SQLSCHEMA

Measure H01 以降の PVU では、SQLSCHEMA コマンドは、部分修飾の ANSI SQL 名の展開で使用するデフォルトのスキーマを指定し、オプションでデフォルトのカタログを指定します。

SQLSCHEMA の構文

```
SQLSCHEMA [ '[catalog-name.]schema-name' ]
```

catalog-name

ANSI SQL カタログ識別子です。デフォルトのカタログが指定されておらず、以前のデフォルトのカタログがない場合は、警告メッセージが表示されます。

schema-name

ANSI SQL スキーマ識別子です。

catalog-name も *schema-name* も指定しない場合は、デフォルトの空の値が使用されます。

SQLSCHEMA の例

部分修飾の ANSI SQL 名を展開するためのデフォルトのスキーマを指定するには、次のコマンドを使用します。

```
+ SQLSCHEMA 'SCHEMA_34'
```

START MEASSUBSYS

START MEASSUBSYS コマンドは、Measure サブシステムを起動します。

START MEASSUBSYS の構文

```
START [ MEASSUBSYS ] [ , cpu ]
```

MEASSUBSYS

ASSUME コマンドを使用してコマンドオブジェクトとして MEASSUBSYS を指定した場合はオプションです。

cpu

システム上の CPU の番号です。*cpu* は、Measure サブシステムがすでに実行中であり、特定の CPU で MEASCTL を再起動する必要があるときに指定します。

通常の状況では、*cpu* パラメーターを使用する必要はないはずです。CPU がダウンした後に再起動すると、MEASMON は、その CPU で新しい MEASCTL プロセスを自動的に起動します。ただし、MEASMON を使用して MEASCTL を開始しようとして失敗した場合は、*cpu* パラメーターを指定した START MEASSUBSYS を発行することで、MEASCTL を手動で開始できます。

cpu パラメーターは、Measure サブシステムが実行中の場合にのみ有効です。

関連コマンド

コマンド	機能
STOP MEASSUBSYS	Measure サブシステムを停止します

△ **注意:**

Measure の実行中は MEASCTL スワップボリュームを停止しないでください。これを行うと、システム障害が発生します。詳しくは、**すべての PVU および RVU での START MEASSUBSYS の使用上の注意**(150 ページ)を参照してください。

すべての PVU および RVU での START MEASSUBSYS の使用上の注意

- START MEASSUBSYS コマンドは、スーパーグループユーザー、つまりユーザー ID（255, *n*）を持つユーザーのみが実行できます。
- MEASCTL スワップボリュームが誤ってダウンするのを避けるため、スワップボリュームとして $SYSTEM またはミラーボリュームのみを使用してください。$SYSTEM 以外のスワップボリュームを使用して MEASSUBSYS を起動すると、MEASCOM は警告メッセージを発行します。可能であれば、MEASSUBSYS を停止してから、再起動してください。

  ```
  >MEASCOM/SWAP $SYSTEM/ START MEASSUBSYS
  ```

- 大規模な測定を開始するのに過度に時間がかかる場合は、測定アプリケーションの START MEASSUBSYS コマンドの前に、以下の DEFINE 文を追加してください。

```
ADD DEFINE  =_MEASCTL_SKEW_COPYTIME, FILE A
ADD DEFINE  =_LOCK_THE_CID_TABLE, FILE A
```

注記:

2 番目の ADD DEFINE 文は、Measure G08 以前の PVU に適用されます。

構文では「FILE A」の補足が必要ですが、TACL では使用されません。

これらの DEFINE 文は、小規模な測定の開始時間には影響しません。

- CID テーブルは、測定できるエンティティの数を決定する内部構造です。このテーブルは、測定対象の各エンティティに 1 つのカウンター識別子（CID）を割り当てます。たとえば、5 つの異なるファイルの測定を要求すると、CID テーブルは、ファイルごとに測定に含まれるカウンターの数に関係なく、1 ファイルあたり 1 つの CID を割り当てます。

  Measure G05 以前の PVU では、CID テーブルにはプロセッサーあたり 32,000 CID という上限があります。製品バージョン D45、G05 以降では、以下に示すように、DEFINE 文に従って CID テーブルの上限を大きくすることができます。H シリーズ以降の RVU では、CID テーブルのデフォルトのサイズはプロセッサーあたり 64,000 CID です。

  CID テーブルの上限を大きくするには、以下のいずれかの DEFINE 文を発行してから、MEASCOM を起動して START MEASSUBSYS コマンドを発行します。注記：

  - 192,000 および 256,000 CID のテーブルの上限は、H シリーズ以降の RVU でのみ有効です。
  - 512,000 CID のテーブルの上限は、Measure H04/J02 以降の PVU でのみ有効です。
  - 1,024,000 CID のテーブルの上限は、H06.25/J06.14 以降の RVU でのみ有効です。

```
> ADD DEFINE  =_SET_MAX_CIDS_TO_64000,  file a
> ADD DEFINE  =_SET_MAX_CIDS_TO_96000,  file a
> ADD DEFINE  =_SET_MAX_CIDS_TO_128000, file a
> ADD DEFINE  =_SET_MAX_CIDS_TO_192000, file a
> ADD DEFINE  =_SET_MAX_CIDS_TO_256000, file a
> ADD DEFINE  =_SET_MAX_CIDS_TO_512000, file a
> ADD DEFINE  =_SET_MAX_CIDS_TO_1024K, file a
```

  `file a` パラメーターは、構文を正確にするためには必要ですが、Measure では使用されません。MEASCOM は、MEASMONCONTROL を呼び出して、DEFINE が有効な状態で Measure サブシステムを起動します。

  512,000 CID より下の 32,000 CID を追加するたびに、Measure パフォーマンスモニターには、CPU ごとに、CID 用に 3.5 メガバイトとカウンター領域用に 6.5 メガバイト（合計 10 メガバイト）のメモリが必要になります。512,000 CID の場合、カウンター領域は約 105 メガバイトです。1,024,000 CID の場合、カウンター領域は約 123 メガバイトです。CID テーブルの上限を大きくすると、システムのパフォーマンスに大きな影響を与える可能性があります。

H シリーズ以降の PVU および RVU についての START MEASSUBSYS の使用上の注意

- H06.20/J06.09 以降の RVU では、START MEASSUBSYS コマンドに対して次のようなエラーメッセージが表示されることがあります。

  `Process_Create_ error` *`MM`* `(`*`NN`*`) from Measmoncontrol`

  指定する値は次の通りです。

MM

Process_Create_から返されたエラーです。

NN

Process_Create_から返されたエラーの詳細です。

- （特定の SPR の）H06.14/J06.04 以降の RVU では、CPU あたりの CID の使用量が設定されているしきい値を超えたことを示す EMS 警告イベント 4063 を表示するように、Measure サブシステムを構成できます。この動作は、CID の数を監視する際に役立ち、特定の時点で測定されるエンティティの数を直接制御します。

  CPU レコードの ENDING-FREE-CIDS カウンターで、使用可能な CID の正確な数を確認することもできます。CID テーブルの使用量の警告しきい値を構成するには、=_MEAS_EMIT_CID_USAGE DEFINE を使用します。このしきい値を超えると、Measure は 4063 警告イベントを生成します。

  しきい値を構成するには、ファイル名の最後の 2 文字を使用して、警告を発行する前に消費できる CID テーブルサイズの割合を指定します。たとえば、CID テーブルの使用量が CID テーブルサイズの 80% を超えると警告が生成されることを要求するには、次のように入力します。

```
TACL> ADD DEFINE =_MEAS_EMIT_CID_USAGE, FILE WARN80
```

  Measure サブシステムを開始する前に、DEFINE を追加する必要があります。Measure は、しきい値が 01 より小さいか 99 より大きい DEFINE は無視します。

  警告イベント 4063 は次のようなものになります。

```
10-04-07 10:04:44 \MEASYOS.$XM00  TANDEM.MEASURE.H04  004063 Exceeded CID
usage threshold.        80%
```

  Measure は警告イベント 4063 を再度生成することはありません。ただし、CID テーブルの使用量を事前に決められたレベルまで減らした後に、指定された警告しきい値をもう一度超えた場合を除きます。これにより、生成される 4063 イベントの数が必要最小限に保たれます。

  この機能は、最初は、RVU H06.21/J06.10 の H05-AGJ/J03-AGK で導入され、H06.20-21/J06.09-10 とともに使用できるようになりました。その後、H04-AGM/J02 AGN SPR で使用可能になり、H06.14-19/J06.04-08 とともに使用できるようになりました。

G シリーズの PVU での START MEASSUBSYS の使用上の注意

- Measure G11 以降の PVU では、Measure のデフォルトの操作は、ANSI SQL 名情報をジャーナリングすることではありません。この機能は、測定のシャットダウン時にパフォーマンスを多用する可能性のあるジャーナルファイル解決のタスクを回避します。デフォルトをオーバーライドするには、START MEASUREMENT 要求で SQL 修飾子を使用します。

  START MEASSUBSYS コマンドを発行する前に、TACL セッションで次の DEFINE 文を使用することもできます。この DEFINE は、START MEASUREMENT コマンドでこれを行うように明示的に要求されている場合を除き、常に ANSI SQL 名情報を収集して解決するように Measure のデフォルトの操作モードを変更します。

```
> ADD DEFINE  =_SQL_JOURNAL_DEFAULT_ON, FILE A
```

  デフォルトのジャーナリングモードの現在の設定を照会するには、STATUS MEASSUBSYS コマンドを使用します。
- Measure G09 以降の PVU では、Measure のデフォルトの操作は、OSS ファイルパス名情報をジャーナリングすることではありません。この機能は、測定のシャットダウン時にパフォーマンスを多用する可能性のあるジャーナルファイル解決のタスクを回避します。デフォルトをオーバーライドするには、START MEASUREMENT 要求で OSS 修飾子を使用します。

  START MEASSUBSYS コマンドを発行する前に、TACL セッションで次の DEFINE 文を使用することもできます。この DEFINE は、START MEASUREMENT コマンドでこれを行うように明示的に要求さ

れている場合を除き、常に OSS ファイルパス名情報を収集して解決するように Measure のデフォルトの操作モードを変更します。

```
> ADD DEFINE  =_OSS_JOURNAL_DEFAULT_ON, FILE A
```

デフォルトのジャーナリングモードの現在の設定を照会するには、STATUS MEASSUBSYS コマンドを使用します。

START MEASUREMENT

START MEASUREMENT コマンドは、測定データが書き込まれるファイルを指定して、測定を開始します。

Measure G09 以降の PVU では、データレコードの内部 OSS パス情報を解決するための OSS ジャーナルセグメントの構築を有効または無効にすることができます。

Measure H02 以降の PVU では、FILESIZE および NOCOUNTERS オプションを使用して、測定データファイルのサイズを選択し、カウンターデータレコードを抑制することができます。START MEASUREMENT コマンドは、オプションを任意の順序で指定できるように拡張されています。

START MEASUREMENT の構文

```
START [ MEASUREMENT ] data-file

      [ , FROM [ start-date , ] start-time-of-day    ]
      [ , FOR duration                               ]
      [ , TO [ end-date , ] end-time-of-day          ]
      [ , INTERVAL interval                          ]
      [ , { OSS | NOOSS }                            ]
      [ , { SQL | NOSQL }                            ]
      [ , FILESIZE file-size [ MB | GB ]             ]
      [ , NOCOUNTERS                                 ]
```

MEASUREMENT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして MEASUREMENT を指定した場合はオプションです。

data-file

測定データが書き込まれるファイルです。独自のデータファイルを割り当てる場合、最大セグメントサイズは 127.5 MB です。

data-file が存在する場合、Measure ファイル処理プロセス（MEASFH）は、測定を開始する前にファイルを消去します。*data-file* が現在アクティブな場合、MEASFH はエラーメッセージを返します。

data-file が存在しない場合は、MEASFH がファイルコード 175 の非構造化ファイルとして作成します。プライマリおよびセカンダリのエクステントサイズはそれぞれ 2048 ページで、エクステントの最大数は 256 です。その結果、デフォルトのファイルサイズ（容量）は 1024 MB です。

data-file がディスクファイルの場合、MEASCOM は暗黙的な ADD MEASUREMENT *data-file* を実行するため、ADD MEASUREMENT コマンドを明示的に入力しなくても、（LIST コマンドのいずれかを使用して）収集したデータを調べることができます。

data-file がテープファイルの場合は、MEASCOM を使用して収集したデータを調べる前に、そのファイルをディスクファイルにコピーする必要があります。

FROM [*start-date* ,] *start-time-of-day*]

測定の開始を指定します。FROM を省略した場合、測定はすぐに開始されます。

start-date

測定の開始日で、次のいずれかの形式です。

```
{ [d]d mmm[ yyyy] }
{ mmm [d]d[ yyyy] }
```

指定する値は次の通りです。

dd は日付で、1～31 の範囲の数字です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT です。

yyyy は西暦です。有効な年は 1984～2047 です。

start-date を省略した場合、現在の日付が使用されます。

start-time-of-day

測定の開始時刻で、次の形式です。

```
hh:mm[:ss]
```

指定する値は次の通りです。

hh は時（0～23）です。

mm は分（0~59）です。

ss は秒（0~59）です。

start-time-of-day を省略した場合、現在の時刻が使用されます。

FOR *duration*

測定の持続時間を指定します。FOR と TO を同時に使用することはできません。両方とも省略する場合は、STOP MEASUREMENT を使用して測定を停止する必要があります。

duration

次のいずれかの形式の時間間隔です。

```
n SECOND[S]
n MINUTE[S]
n HOUR[S]
```

指定する値は次の通りです。*n* は 1～9999 の範囲の整数です。

TO [*end-date* ,] *end-time-of-day*]

測定の停止時間を指定します。TO と FOR を同時に使用することはできません。両方とも省略する場合は、STOP MEASUREMENT を使用して測定を停止する必要があります。

end-date

測定を停止する日付です。次のいずれかの形式を使用します。

```
{ [d]d mmm[ yyyy] }
{ mmm [d]d[ yyyy] }
```

指定する値は次の通りです。

dd は日付で、1～31 の範囲の数字です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT です。

yyyy は西暦です。有効な年は 1984～2047 です。

end-date を省略した場合、現在の日付が使用されます。

end-time-of-day

測定を停止する時刻です。次の形式を使用します。

```
hh:mm[:ss]
```

指定する値は次の通りです。

hh は時（0～23）です。

mm は分（0~59）です。

ss は秒（0~59）です。

end-time-of-day を省略する場合は、STOP MEASUREMENT を使用して測定を停止する必要があります。

INTERVAL *interval*

収集間隔、つまり、Measure が測定データファイルに書き込む前にデータを収集する時間の長さを指定します。INTERVAL を省略した場合、Measure 制御プロセスはファイルにカウンター値を 2 回書き込みます。1 回目は測定（または一時的なエンティティの測定）の開始時で、2 回目は測定（または一時的なエンティティの測定）の終了時です。

INTERVAL を使用すると、測定の最中に（または一時的なエンティティが存在している間に）、ファイルへの定期的な追加のカウンターレコードの書き込みを指定できます。

interval

次のいずれかの形式の収集間隔です。

```
n SECOND[S]
n MINUTE[S]
n HOUR[S]
```

指定する値は次の通りです。*n* は 1～9999 の範囲の整数です。

> △ **注意:**
>
> 収集間隔はシステムリソースに対する負荷を増やすため、測定中のパフォーマンスに影響を及ぼす可能性があります。収集間隔とパフォーマンスについては、**すべての PVU および RVU での START MEASUREMENT の使用上の注意**(156 ページ)を参照してください。

OSS

OSS ファイルパス名の解決のための OSS ジャーナルセグメントを、測定データの一部として構築することを指定します。このオプションは、OSS ジャーナルセグメントの構築のデフォルト設定をオーバーライドします。

NOOSS

OSS ファイルパス名の解決のための OSS ジャーナルセグメントを、測定データの一部として構築しないことを指定します。このオプションは、OSS ジャーナルセグメントの構築のデフォルト設定をオーバーライドします。

SQL

SQL 名の解決のための SQL ジャーナルセグメントを、測定データの一部として構築することを指定します。このオプションは、SQL ジャーナルセグメントの構築のデフォルト設定をオーバーライドします。

NOSQL

SQL 名の解決のための SQL ジャーナルセグメントを、測定データの一部として構築しないことを指定します。このオプションは、SQL ジャーナルセグメントの構築のデフォルト設定をオーバーライドします。

FILESIZE *file-size* [MB | GB]

目的のファイルサイズ（容量）を指定します。GB で指定する場合、ファイルサイズはギガバイト数（カウント * $1{,}024^3$）と解釈されます。MB で指定する場合、または修飾を省略した場合、ファイルサイズはメガバイト数（カウント * $1{,}024^2$）と解釈されます。このオプションは、TMF ADD/ALTER AUDITTRAIL コマンドの対応するオプションに似ています。FILESIZE の最小値は 127MB です。最大値は、1048572 MB、1023 GB、または最大ディスクサイズ（1,048,572 MB、つまり 1 TB – 4 MB より小さい場合）です。デフォルト値は 1024 です（結果のファイルは 1024 MB）。ファイルの容量は最小として指定された容量になり、実際の容量はより大きくなる場合があります。ファイルがすでに存在する場合、このオプションは無視され、既存のファイルの容量が使用されます。

NOCOUNTERS

測定データファイルからのカウンターデータレコードを抑制します。構成情報のみが含まれる測定データファイルが作成されます。アクティブなカウンターレコードは、システムカウンター領域に維持されます。このオプションは、アクティブなカウンターレコードのみを使用する予定の場合にディスク領域を節約するために使用することができます。PROCESSH カウンターを測定の一部として定義すると、ユーザーに対して、これらのカウンタータイプではアクティブなカウンターは使用できないという警告 3026 が表示されます。OSS または SQL オプションを指定すると、ユーザーに対して、NOCOUNTERS オプションが有効な場合は OSS/SQL ジャーナルセグメントは生成されないという警告 3027 が表示されます。

関連コマンド

コマンド	機能
ADD *entity-type*	測定を構成します
STOP MEASUREMENT	測定を停止します

すべての PVU および RVU での START MEASUREMENT の使用上の注意

- 測定中にシステム時間が変更されると、後処理で間違った結果が報告されることがあります。測定間隔の精度を維持するためには、システム時間を変更する前に（たとえば、TACL の SETTIME コマンドまたは SETSYSTEMCLOCK プロシージャーを使用する前に）、すべての測定を停止してください。時間の変更は、通常は夏時間（DST）への移行時に行われます。
- INTERVAL は、測定内の特定の時間間隔のデータを調べる必要がある場合にのみ使用します。INTERVAL は、指定された間隔ごとに測定対象のすべての CPU のすべてのカウンターをデータファイルに書き込むため、実行される I/O 数が増加します。INTERVAL の値が小さいほど、システムリソース（CPU、ディスク、メモリ）の消耗が激しくなり、データファイルが急速に拡大します。システムリソースを節約するには、INTERVAL 値を、最も長い間隔で有用な情報が提供されるように設定してください。

  非常に大規模な測定で間隔の時間が小さいと、パフォーマンスに深刻な問題が発生する可能性があります。問題が発生した場合は、測定対象のエンティティの数とタイプを減らすか、収集間隔を指定せずに測定を実行してください。

H シリーズ以降の PVU についての START MEASUREMENT の使用上の注意

Measure H02 以降の PVU では、MEASCOM START MEASUREMENT コマンドを使用して、ユーザーが測定データファイルのサイズを選択したり、測定データファイルのカウンターデータレコードを抑制したり、あるいはその両方を実行したりできます。

G シリーズの PVU での START MEASUREMENT の使用上の注意

- Measure G11 以降の PVU では、Measure サブシステムのデフォルトでは、START MEASUREMENT コマンドで特に要求されている場合を除き、SQL 名のジャーナルセグメントは構築されません。SQL 名のジャーナルセグメントが常に構築されるようにデフォルトを切り替えるには、START MEASSUBSYS コマンドの前に DEFINE を使用します。詳しくは、**START MEASSUBSYS**(150 ページ)の使用上の注意を参照してください。
- Measure G09 以降の PVU では、Measure サブシステムのデフォルトでは、START MEASUREMENT コマンドで特に要求されている場合を除き、OSS ファイルパス名のジャーナルセグメントは構築されません。OSS ファイルパス名のジャーナルセグメントが常に構築されるようにデフォルトを切り替えるには、START MEASSUBSYS コマンドの前に DEFINE を使用します。詳しくは、**START MEASSUBSYS**(150 ページ)の使用上の注意を参照してください。

START MEASUREMENT の例

- Measure G11 以降の PVU では、SQL 名のジャーナルセグメントを構築して、それが測定のシャットダウン時に測定データファイルに追加されるようにするには、次のコマンドを使用します。

```
+ START MEASUREMENT MDATA, SQL
```

- Measure G09 以降の PVU では、OSS ファイルパス名のジャーナルセグメントを構築して、それが測定のシャットダウン時に測定データファイルに追加されるようにするには、次のコマンドを使用します。

```
+ START MEASUREMENT MDATA, OSS
```

- すぐに測定を開始し、MAY07 というデータファイルに 1 時間ごとに 1 回書き込みを行い、8 時間で停止するには、次のコマンドを使用します。

```
+ START MEASUREMENT MAY07, FOR 8 HOURS, INTERVAL 1 HOUR
```

- 2004 年 10 月 19 日の 12:00 に測定を開始し、2004 年 10 月 26 日の 12:00 まで測定を継続して、この測定で 6 時間ごとに OCTDATA というデータファイルに書き込みが行われるようにするには、次のコマンドを使用します。

```
+ START MEASUREMENT OCTDATA, FROM 19 OCT 2004, 12:00 &
& TO 26 0CT 2004, 12:00, INTERVAL 6 HOURS
```

STATUS MEASSUBSYS

STATUS MEASSUBSYS コマンドは、以下のステータス情報を表示します。

- アクティブな測定または設定済みの測定の数と、測定データファイル名のリスト
- アクティブな MEASCTL プロセスの数と、それらのプロセスがアクティブである CPU のリスト
- （Measure G09 以降の PVU）サポートされる CID の数、CID テーブルのロック、OSS ジャーナルセグメント構築の現在のデフォルト設定など、いくつかのサブシステム環境設定の現在の設定
- （Measure G11 以降の PVU）SQL ジャーナルセグメント構築の現在の設定（ON または OFF）

STATUS MEASSUBSYS の構文

```
STATUS [ / OUT filename / ] [ MEASSUBSYS ]
```

OUT *filename*

コマンド出力を *filename* に送ります。*filename* で指定したファイルが存在しない場合、MEASCOM はその名前で EDIT ファイルを作成し、コマンド出力をそのファイルに書き込みます。*filename* で指定したファイルが存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

STATUS MEASSUBSYS コマンドを実行した後、MEASCOM はその出力を現在の OUT ファイル（通常は端末）にリダイレクトします。

OUT オプションは、ログファイルの内容には影響しません。

MEASSUBSYS

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして指定した場合はオプションです。

STATUS MEASSUBSYS の使用上の注意

CID は、カウンター識別子を表します。プロセッサー内のカウンターレコードごとに、Measure は CID を使用してカウンターレコードを割り当てて追跡します（カウンターレコードは、ファイルやプロセスなどの測定対象エンティティに対応します）。プロセッサーでサポートされる CID の数には制限があります。この制限は、START MEASSUBSYS が発行された時点で、CPU ごとに 32,000、64,000、96,000、128,000、192,000、256,000、512,000、または 1,024,000 に指定されます。（値 192,000 と 256,000 は H シリーズ以降の RVU のみ、値 512,000 は Measure H04/J02 以降の PVU のみ、値 1,024,000 は H06.25/J06.14 以降の RVU のみで有効です。）報告される CID 値は、CID テーブルのサイズです。現在使用中の CID の数は、各プロセッサーの CPU エンティティレポートで報告されます。

STATUS MEASSUBSYS の例

- Measure H06.25/J06.14 以降の RVU では、1,024,000 の新しい CID テーブルサイズの表示がサポートされています。

```
+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements = 0
Number of Active MEASCTL Processes = 4
  in CPU(s): 0, 1, 2, 3
Number of Active MEASIP Processes = 8
  in CPU(s): 0, 1, 2, 3
OSS OFF, SQL OFF, Skew copy time OFF, Lock CID table ON,
CIDs = 1024K/CPU
```

- Measure J01 以降の PVU では、MEASIP プロセスは各 IPU でアクティブです。この変更は、実行中の MEASIP プロセスの合計数に反映されます。

```
+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements =  0

Number of Active MEASCTL Processes =  4
  in CPU(s): 0, 1, 2, 3
Number of Active MEASIP Processes =  8
  in CPU(s): 0, 1, 2, 3
OSS OFF, SQL OFF, Skew copy time OFF, Lock CID table ON,
CIDs = 64K/CPU
```

- Measure H01 以降の PVU で MEASIP プロセスに関する情報を表示するには、次のコマンドを使用します。

```
+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements =  3
  $SYSTEM.SYSTEM.ZASPAA
  $SYSTEM.SYSTEM.ZASPZOO
  $SYSTEM.MEASURE.DOFF3

Number of Active MEASCTL Processes =  8
  in CPU(s): 0, 1, 2, 3, 4, 5, 6, 7
Number of Active MEASIP Processes =  8
  in CPU(s): 0, 1, 2, 3, 4, 5, 6, 7

OSS OFF, SQL OFF, Skew copy time OFF, Lock CID table ON,
CIDs = 64K/CPU
```

- Measure G11 以降の PVU で、アクティブな測定と Measure サブシステムのステータス（アクティブな MEASCTL プロセスを含む）を表示して、SQL ジャーナルセグメントの構築がオンかどうかを示すには、次のコマンドを使用します。

```
+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements =  1
  $DATA1.SUBVOL.MDATA

Number of Active MEASCTL Processes = 8
  in CPU(s): 0, 1, 2, 3, 4, 5, 6, 7
SQL OFF, OSS OFF, Skew copy time OFF, Lock CID table OFF, CIDs = 32K/CPU
```

- Measure G09 以降の PVU で、アクティブな測定と Measure サブシステムのステータス（アクティブな MEASCTL プロセスを含む）を表示するには、次のコマンドを使用します。

```
+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements =  1
  $DATA1.SUBVOL.MDATA

Number of Active MEASCTL Processes = 6
  in CPU(s): 0, 1, 2, 3, 4, 5
OSS ON, Skew copy time OFF, Lock CID table OFF, CIDs = 128K/CPU
```

- Measure D シリーズ以降の PVU で、アクティブな測定と MEASCTL プロセスを表示するには、次のコマンドを使用します。

```
+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements =  1
  $DATA1.PERF.MAY04

Number of Active MEASCTL Processes = 9
  in CPU(s): 0, 1, 2, 3, 4, 5, 6, 7, 8
```

STATUS MEASUREMENT

STATUS MEASUREMENT コマンドは、現在アクティブな測定に関する以下の情報を表示します。

- 測定対象のエンティティの数
- 各エンティティタイプに現在割り当てられているシステムデータ領域内のワードの数
- 測定の開始時刻、停止時刻、および収集間隔
- データファイルの現在のサイズ
- データファイルの最大サイズ
- ファイルの使用パーセンテージ
- （Measure G09 以降の PVU）OSS ジャーナルセグメントの構築が ON であるか、OFF であるか
- （Measure G11 以降の PVU）SQL ジャーナルセグメントの構築が ON であるか、OFF であるか

STATUS MEASUREMENT の構文

```
STATUS [ / OUT filename / ] [ MEASUREMENT ] data-file
```

OUT *filename*

コマンド出力を *filename* に送ります。*filename* で指定したファイルが存在しない場合、MEASCOM はその名前で EDIT ファイルを作成し、コマンド出力をそのファイルに書き込みます。*filename* で指定したファイルが存在する場合、MEASCOM はそのファイルを開いて、コマンド出力を追加します。

STATUS MEASUREMENT コマンドを実行した後、MEASCOM はその出力を現在の OUT ファイル（通常は端末）にリダイレクトします。このオプションは、ログファイルの内容には影響しません。

MEASUREMENT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして MEASUREMENT を指定した場合はオプションです。

data-file

アクティブな測定データファイルを指定します。MEASCOM はこの情報をデータファイルから読み取るのではなく Measure 調整プロセス（MEASMON）に情報を提供するよう要求するため、MEASCOM からこのデータファイルに（ADD MEASUREMENT コマンドを使用して）アクセス可能である必要はありません。

STATUS MEASUREMENT の使用上の注意

- 停止時刻は、START MEASUREMENT コマンドで明示的に時間を指定した場合にのみ表示されます。
- Measure H02 以降の PVU では、非常に大規模なデータファイルの統計情報を表示したり、測定でカウンターデータレコードが抑制されているかどうかを表示したりできるように、MEASCOM の STATUS MEASUREMENT コマンドによって生成される出力が拡張されています。

STATUS MEASUREMENT の例

- Measure H02 以降の PVU では、STATUS MEASUREMENT で生成される出力で、カウンターデータレコードの抑制が測定に指定されたかどうかを表示できます。カウンターデータレコードが抑制されている場合は、「Counter Suppression ON」と表示されます。抑制されていない場合は、「Counter Suppression OFF」と表示されます。カウンターの抑制をオフにする例を次に示します。

```
+ STATUS MEASUREMENT MEASXX
From 16 Aug 2006, 13:52:32
Cpu                       1 Entities             380 Words
Current EOF            8192 B        Maximum EOF   1099510579200 B
OSS Journal OFF   SQL Journal OFF    Percentage Used        0.00 %
Counter Suppression OFF
```

- Measure G11 以降の PVU で、アクティブな測定と、OSS および SQL ジャーナルセグメント構築のステータスを表示するには、次のコマンドを使用します。

```
+ STATUS MEASUREMENT MDATA
From 26 Feb 2003, 15:59:54
File                     13 Entities            1144 Words
Current EOF            8192 B        Maximum EOF       133693440 B
OSS Journal OFF    SQL Journal OFF   Percentage Used        0.00 %
```

- Measure D シリーズ以降の PVU で、OCT18 に書き込みを行う現在のアクティブな測定に関する情報を表示するには、次のコマンドを使用します。

```
+ STATUS MEASUREMENT OCT18
From 18 Oct 1994, 14:35:33
Cpu                       1 Entities              46 Words
Disc                     64 Entities            7552 Words
Current EOF             178 B         Maximum EOF  1556480 B
                                      Percentage Used  0.3 %
```

STOP MEASSUBSYS

STOP MEASSUBSYS コマンドは、すべての測定とサブシステムプロセスを停止します。

STOP MEASSUBSYS の構文

```
STOP [ MEASSUBSYS ]
```

MEASSUBSYS

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして MEASSUBSYS を指定した場合、MEASSUBSYS はオプションです。

関連コマンド

コマンド	機能
START MEASSUBSYS	Measure サブシステムを起動します
STATUS MEASSUBSYS	測定がアクティブかどうかを判別します。

STOP MEASSUBSYS の使用上の注意

- STOP MEASSUBSYS コマンドは、スーパーグループユーザー、つまりユーザー ID（255,*）を持つユーザーのみが実行できます。
- アクティブな測定が存在する間に対話形式でこのコマンドを入力すると、MEASCOM は測定に関する警告を表示し、コマンドを確認するよう求めます。このコマンドを非対話形式で入力するかコマンド（OBEY）ファイルから入力すると、MEASCOM は確認を求めずにサブシステムを停止します。

STOP MEASSUBSYS の例

MEASCOM を起動し、Measure サブシステムのステータスをチェックしてからサブシステムを停止するには、次のコマンドを使用します。

```
MEASCOM
MEASURE Performance Monitor - T9086H06 - (01AUG11) - \SYSTEM
(C) Copyright 1986-2010 Hewlett-Packard Development Company, L.P.
+ STATUS MEASSUBSYS
Number of Active (or Configured) Measurements =  0

Number of Active MEASCTL Processes =  4
  in CPU(s): 0, 1, 2, 3
Number of Active MEASIP Processes =   4
  in CPU(s): 0, 1, 2, 3

OSS OFF, SQL OFF, Skew copy time OFF, Lock CID table ON, CIDs = 64K/CPU
+ STOP MEASSUBSYS
```

STOP MEASUREMENT

STOP MEASUREMENT コマンドは、指定した測定を停止します。

STOP MEASUREMENT の構文

```
STOP [ MEASUREMENT ] data-file [ , NO ADD ]
     [ , { TO | AT } [ end-date , ] end-time-of-day ]
```

MEASUREMENT

ASSUME コマンドを使用してデフォルトのコマンドオブジェクトとして MEASUREMENT を指定した場合はオプションです。

data-file

アクティブな測定データファイルを指定します。

data-file がディスクファイルであり、NO ADD オプションが指定されていない場合、MEASCOM は暗黙的に ADD MEASUREMENT *data-file* を実行します。そのため、最初に ADD MEASUREMENT コマンドを明示的に入力しなくても、（LIST コマンドのいずれかを使用して）収集したデータを調べることができます。

data-file がテープファイルの場合は、MEASCOM を使用して収集したデータを調べる前に、そのファイルをディスクファイルにコピーする必要があります。

NO ADD

測定を停止したときに暗黙的に ADD MEASUREMENT *data-file* コマンドが実行されないようにします。測定を停止してもデータファイルを調べない場合は、NO ADD オプションを使用します。

{ TO | AT } [*end-date* ,] *end-time-of-day*]

測定を停止する時刻を指定します。TO と AT は同義です。TO 句または AT 句を省略すると、MEASCOM はすぐに測定を停止します。

end-date

測定を停止する日付です。次のいずれかの形式を使用します。

```
{ [d]d mmm[ yyyy] }
{ mmm [d]d[ yyyy] }
```

指定する値は次の通りです。

dd は日であり、1～31 の範囲の数値です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT です。

yyyy は西暦です。有効な年は 1984～2047 です。

date を省略した場合、現在の日付が使用されます。

end-time-of-day

測定を停止する時刻です。次の形式を使用します。

```
hh:mm[:ss]
```

指定する値は次の通りです。

hh は時間です（0～23）。

mm は分です（0~59）。

ss は秒です（0~59）。

end-time-of-day を省略した場合、MEASCOM はすぐに測定を停止します。

関連コマンド

コマンド	機能
ADD MEASUREMENT	調査する測定データファイルを選択します。

STOP MEASUREMENT の例

- 測定をすぐに停止してから MEASFH プロセスを起動し、LIST コマンドを使用してデータを表示できるようにデータファイルを準備するには、次のコマンドを使用します。

  + **STOP MEASUREMENT $DATA.MEAS.MAY04**
- 測定をすぐに停止しても MEASFH プロセスを起動しない場合は、次のコマンドを使用します。その場合、LIST コマンドでデータを使用できません。

  + **STOP MEASUREMENT $DATA.MEAS.MAY04, NO ADD**

SWAPVOL

SWAPVOL コマンドは、作業ファイルが作成されるスワップボリュームを指定します。このコマンドは、MEASCOM の残りのセッションに対して、または別の SWAPVOL コマンドが実行されるまで適用されます。

SWAPVOL の構文

```
SWAPVOL [ $volume ]
```

$*volume*

ボリュームの名前です。ボリューム名を省略すると、スワップボリュームはデフォルトである MEASCOM スワップボリュームにリセットされます。

SWAPVOL の使用上の注意

デフォルトでは、スワップファイルは MEASCOM スワップボリューム上に作成されますが、これには一時ファイル用の十分な領域がない場合があります。その場合は、MEASCOM セッションの開始時に SWAPVOL コマンドを使用して、MEASCOM が十分なスペースのあるボリュームに一時ファイルを入れるようにしてください。

SYSTEM

SYSTEM コマンドは、ファイル名の拡張で使用されるデフォルトのシステム名を設定します。システムは、最初はオペレーティングシステムコマンドインタープリター（TACL）によって指定されます。

SYSTEM の構文

```
SYSTEM [ \system ]
```

system

システム名です。省略すると、コマンドインタープリターによって定義されたデフォルトのシステムが、MEASCOM のデフォルトになります。

SYSTEM の例

デフォルトのシステム名を変更するには、次のコマンドを使用します。

+ **SYSTEM \SELL**

TIME

TIME コマンドは、現在のシステム日付と時刻を表示します。

TIME の構文

```
TIME [ \system ]
```

system

システム名です。TIME は、このシステムの地方常用時を返します。システムを指定しない場合、TIME はローカルシステム時刻を返します。

TIME の例

- ローカルシステムの時刻を表示するには、次のコマンドを使用します。

  ```
  + TIME
  20 November 1994 18:02:25
  ```

- システム\NY のの時刻を表示するには、次のコマンドを使用します。

  ```
  + TIME \NY
  20 November 1994 21:02:30
  ```

VOLUME

VOLUME コマンドは、ファイル名拡張のためにデフォルトのデバイス名、サブボリューム名、またはその両方を設定します。最初は、コマンドインタープリターによってデフォルトのデバイスとサブボリュームが指定されます。

VOLUME の構文

```
VOLUME [   $device             ]
       [ [ $device.]subvolume ]
```

$*device*

デバイス名です。$*device* を省略した場合、デフォルトのデバイスは変更されません。$*device* と *subvolume* を両方とも省略した場合、コマンドインタープリターで指定したデフォルトが MEASCOM のデフォルトになります。

subvolume

サブボリューム名です。*subvolume* を省略した場合、デフォルトのサブボリュームは変更されません。$*device* と *subvolume* を両方とも省略した場合、コマンドインタープリターで指定したデフォルトが MEASCOM のデフォルトになります。

VOLUME の例

- デフォルトのボリュームとサブボリュームを変更するには、次のコマンドを使用します。

  ```
  + VOLUME $DATA2.MAINT
  ```

- デフォルトのボリュームとサブボリュームを、コマンドインタープリターで定義されたものに変更するには、次のコマンドを使用します。

  ```
  + VOLUME
  ```

WARNINGS

WARNINGS コマンドは、表示される警告メッセージを指定します。警告メッセージは、WARNING という単語と、3000 から 3049 までの範囲の番号によって識別されます。MEASCOM で、指定したものを除くすべての警告を表示することも、指定したものを除くすべての警告を表示しないこともできます。

WARNINGS の構文

```
WARNINGS { DISPLAY  { ALL | warn-num [ , warn-num ... ] } }
         { SUPPRESS { ALL | warn-num [ , warn-num ... ] } }
```

DISPLAY

表示される警告メッセージを指定します。

ALL

後続の 1 つまたは複数の WARNINGS SUPPRESS コマンドによって指定されたものを除き、すべての警告を表示します。デフォルトは、WARNINGS DISPLAY ALL です。

warn-num

警告の番号です（3000～3049、3116、3233、3276、3277、または 3299）。

SUPPRESS

表示しない警告メッセージを指定します。

ALL

後続の 1 つまたは複数の WARNINGS DISPLAY コマンドによって指定されたものを除き、すべての警告を表示しません。

関連コマンド

コマンド	機能
COMMENTS	コメントメッセージの表示または非表示を選択します。

WARNINGS の例

警告メッセージ 3013 を出力しないようにするには、次のコマンドを使用します。

```
+ WARNINGS SUPPRESS 3013
```

!（感嘆記号）

!コマンドは、以前に実行されたコマンドを履歴バッファーから取得し、コマンドをもう一度実行します。

!（感嘆記号）の構文

```
!  [  number ]
   [ -number ]
   [  text   ]
```

number

再実行するコマンドが含まれる履歴バッファー内の行数です。（**!（感嘆記号）の例**(166 ページ)の最初の例を参照してください。）

-number

再実行するコマンドに到達するために現在の行から減算する履歴バッファーの行数です。（**!（感嘆記号）の例**(166 ページ)の 2 番目の例を参照してください。）

text

テキスト文字列です。このテキスト文字列で始まる最後に入力されたコマンドが取得されます。（**!（感嘆記号）の例**(166 ページ)の 3 番目の例を参照してください。）

オプションを指定しない場合は、入力された最後のコマンドが再実行されます。（**!（感嘆記号）の例**(166 ページ)の 4 番目の例を参照してください。）

関連コマンド

コマンド	機能
FC	前の MEASCOM コマンドを取得、変更、および再実行します
HISTORY	履歴バッファーを表示します

!（感嘆記号）の例

- 履歴バッファーの 15 行目のコマンドを再実行するには、以下のようにします。

  ```
  21+ ! 15
  ```

- 履歴バッファー内で現在の行より 3 行前にあるコマンドを再実行するには（この場合、現在の行は 24 行目であるため、履歴バッファーの 21 行目のコマンド）、以下のようにします。

  ```
  24+ ! -3
  ```

- テキスト文字列 SE で始まる最後に入力されたコマンド（たとえば、SET REPORT FORMAT BRIEF）を再実行するには、以下のようにします。

  ```
  25+ ! SE
  ```

- 入力された最後のコマンドを再実行するには、以下のようにします。

  ```
  28+ !
  ```

エンティティおよびカウンター

この章では、Measure のエンティティタイプとその関連付けられたカウンターについて説明します。

トピック

カウンターの概要(168 ページ)

一般的なエンティティヘッダーフィールド(177 ページ)

オープンシステムサービス（OSS）に対する Measure のサポート(181 ページ)

ANSI SQL 名に対する Measure のサポート(183 ページ)

ZMS 形式のレコードへのアクセス（MEASDDLZ）(186 ページ)

表 2: Measure のエンティティタイプ

エンティティタイプ	測定されるシステムリソース
CLUSTER	光ファイバー拡張（FOX）および ServerNet/FX クラストラフィック。H シリーズ以降の RVU には適用されません。
CONTROLLER	NonStop K シリーズサーバーのみについてのデバイスコントローラーの I/O アクティビティ。（Measure D30（T9086ACV）、D44（T9086ACR）、および D45（T9086ACQ）では無効です。有効にするには、**CONTROLLER**(191 ページ)に説明されている定義を設定します。）
CPU	プロセッサー。
DEVICE	テープドライブなどの I/O デバイス。ディスク、通信回線、サブデバイス、および非同期端末は含まれません。これらの各デバイスは、それら独自のエンティティタイプを持ちます。G シリーズ以降の RVU では、このエンティティはテープおよび Open SCSI IOPs に適用されます。
DISC または DISK	ディスク。
DISCOPEN または DISKOPEN	指定されたディスクファイルへの単一のオープナープロセスによって実行される I/O 操作（物理的なファイルアクセス）。
DISKFILE	指定されたディスクファイルへのすべてのオープナープロセスによって実行される I/O 操作（物理的なファイルアクセス）。
ENDPOINT	システム相互接続のクライアント。つまりメッセージシステム、ストレージサブシステム、および通信サブシステム。
EXTNAMES	OSS パス名と ANSI SQL 名のための外部レコード形式を含むファイル。EXTNAMES（コマンドインターフェイスの正式な Measure エンティティではない）は、LIST EXTNAMES コマンドから生成されたレコード形式で構造化出力ファイルです。

表は続く

エンティティタイプ	測定されるシステムリソース
FABRIC	特定の仮想チャネル上のエンドポイントを含む物理 I/O 操作の詳細情報。
FILE	明示的に開かれたファイルにローカルユーザープロセスによって実行される I/O 操作（論理的なファイルアクセス）。
LINE	SNAX/XF や X25AM 回線などの通信回線。
NETLINE	Expand 回線などのネットワーク通信回線。
OPDISK	オプティカルディスク（D シリーズおよび Himalaya L シリーズの RVU）。
OSSCPU	各システムプロセッサーの OSS 要素。
OSSNAMES	OSS ファイルパス名マッピング。OSSNAMES ファイル（コマンドインターフェイスの正式な Measure エンティティではない）は、LIST OSSNAMES コマンドから生成されたレコード形式で構造化出力ファイルです。このファイルは、内部形式の OSS PATHID および CRVSN データ（別のレコードタイプ）を外部 OSS ファイルパス名変換にリンクするために使用します。
OSSNS	OSS 名のサーバープロセス。
PROCESS	プロセス。
PROCESSH	プロセスコード範囲ヒストグラム。
SERVERNET	ENDPOINT を参照してください。
SQLPROC	SQL/MP または SQL/MX プロセス。
SQLSTMT	SQL プロセス内の SQL/MP または SQL/MX 文。
SYSTEM	Expand 回線ハンドラー経由のネットワークトラフィック（ローカル側のみ）。
TERMINAL	端子およびサブデバイス I/O。
TMF	HPE NonStop トランザクション管理ファシリティ（TMF）のトランザクション。
USERDEF	ユーザー定義アプリケーションインストゥルメンテーション。

カウンターの概要

カウンター値の解釈

Measure パフォーマンスモニターは、さまざまなタイプのカウンターを使用して、データを累積します。カウンターが MEASCOM レポートに表示される場合、表示される値は現在の REPORT RATE および REPORT FORMAT 属性に影響されます。

REPORT RATE の設定	カウンター値は次のように表示されます。
ON	平均値または割合。具体的な解釈は、カウンタータイプによって異なります。
OFF	raw データ。これらのデータには数学的な操作が実行されていません。

Enform などの他の製品がアクセスできる Measure データの構造化ファイルを作成するには、REPORT FORMAT STRUCTURED を使用します。REPORT FORMAT が STRUCTURED の場合、カウンター値は raw データとして出力ファイルに書き込まれます。REPORT FORMAT STRUCTURED は、REPORT RATE 設定では影響されません。

注記:

このセクションのカウンターの説明では、REPORT FORMAT NORMAL または BRIEF で REPORT RATE OFF を想定しています。

Measure は、カンマとピリオドを含めて 10 文字以内に収まるように表示される値を拡張します。フォーマットされた値の単位は、以下の記号で示されます。

#	1 単位の 10 進数を示します
K	千単位の 10 進数を示します（キロ）
M	百万単位の 10 進数を示します（メガ）
G	十億単位の 10 進数を示します（ギガ）
T	兆単位の 10 進数を示します（テラ）

たとえば、1 メガバイトは 1048576 #と表示されます。

累積カウンター

累積カウンターは、ファイル、ディスク、またはその他のエンティティとやり取りされるバイト数をカウントします。エンティティとの間でデータが転送されるとき、カウンターに値が追加されます。

属性の設定	出力値	表示単位
REPORT RATE は ON	その期間に転送された 1 秒あたりのバイト数。	1 秒あたりの値：/s、K/s、M/s、G/s、または T/s[1]
REPORT RATE は OFF	その期間に転送されたバイト数。	#、K、M、G、または T
REPORT FORMAT は STRUCTURED	その期間に転送されたバイト数。	

[1] ZMS 形式のみ。Legacy 形式では単位は表示されません。位取りを行わない/s であるとみなされ、オーバーフローする場合があります。

Measure では、32 ビットおよび 64 ビットの両方の累積カウンターをサポートしています。G シリーズの RVU では、一部の累積カウンターは 32 ビットバージョンおよび 64 ビットバージョンの 2 つのバージョンで存在します。64 ビットバージョンは、カウンター名が文字-F で終わることによって区別されます。たとえば、SENT-BYTES は 32 ビットカウンターで、SENT-BYTES-F はこれに対応する 64 ビットカウンターです。H シリーズ以降の RVU では、このようなすべての累積カウンターは 64 ビットで、-F で終わ

る名前を持つ ZMS 形式のレコード内のカウンターはありません。たとえば H シリーズ以降の RVU では、SENT-BYTES は 64 ビットカウンターです。SENT-BYTES-F は ZMS 形式のカウンター名ではありませんが、Legacy 形式でデータを要求するアプリケーションで引き続き使用されます。

累積カウンターの例については、FILE エンティティカウンター **MESSAGE-BYTES**(300 ページ)を参照してください。

キューの平均時間カウンター（ZMS 形式のみ）

キューの平均時間カウンターは、キューにアクセスする各項目がキューで費やした時間の平均を測定します。この派生カウンターは、キューカウンターを、キューにアクセスする項目の対応する増分カウンターで割ることで算出されます。

キューカウンターの実装では、キューの平均時間カウンターが、イベントを処理するための平均サービス時間を含むかどうか、またはキューイング時間だけであるかどうかがカウンターのサフィックスによって判断されます。

サフィックス	サービス時間を含む
-AQT	いいえ
-ART	はい

キューの平均時間カウンターは、RATE を ON または OFF に設定することによる影響を受けません。単位は ms、sec、hr、day、または wk として表示されます。

キューの平均時間カウンターの例については、**DISC**(226 ページ)にある ZMS 形式のレポートの REQUEST-ART を参照してください。

サービスの平均時間カウンター（ZMS 形式のみ）

サービスの平均時間カウンターは、リソースの各要求を満たすまでの平均時間を測定します。この派生カウンターは、リソースの要求に対応する増分カウンターでリソースのビジーカウンターまたはキュービジーカウンターを割ることで算出されます。サービスの平均時間カウンターは、接尾辞-AST によって識別して表示されます。

サービスの平均時間カウンターは、RATE を ON または OFF に設定することによる影響を受けません。単位は ms、sec、hr、day、または wk として表示されます。

サービスの平均時間カウンターの例については、**DISC**(226 ページ)にある ZMS 形式のレポートの DEVICE-AST を参照してください。

ビジーカウンター

ビジーカウンターはリソースがビジーである時間を測定します。カウンターは、ビジーまたはアイドル状態の語句および 64 ビットのビジー時間アキュムレータを使用して、累積的なビジー時間をマイクロ秒単位で記録します。

属性の設定	出力値	表示単位
REPORT RATE は ON。	期間中でのビジー時間の割合。	間隔の長さに対する割合
REPORT RATE は OFF。	期間中でのビジー時間。	ms（ミリ秒）、sec、hr、day、または wk
REPORT FORMAT は STRUCTURED。	ビジー時間はマイクロ秒で表現されます。	

ビジーカウンターの例は、CPU エンティティカウンター（**CPU-BUSY-TIME**(201 ページ)）を参照してください。

H シリーズ以降の RVU では、ビジーカウンターには以下の 2 種類があります。

- BUSY カウンター、従来のタイマーで維持
- TCELLBUSY カウンター、タイマーセルで維持

タイマーセルおよび関連するカウンターの概要については、**タイマーセルカウンター**(175 ページ)を参照してください。

経過時間カウンター

経過時間カウンターは、イベントの開始から終了までの時間を測定します。

属性の設定	出力値	表示単位
REPORT RATE は ON。	各イベントごとの平均経過時間。	ミリ秒、秒、時間、日、または週
REPORT RATE は OFF。	経過時間。	ミリ秒、秒、時間、日、または週
REPORT FORMAT は STRUCTURED。	マイクロ秒で表現される経過時間。	

経過時間カウンターの例は、SQLSTMT エンティティカウンター（**ELAPSED-SORT-TIME**(431 ページ)）を参照してください。

増分カウンター

増分カウンターはイベントをカウントします。このカウンターは、イベントが起きるたびに加算されます。

属性の設定	出力値	表示単位
REPORT RATE は ON。	その期間中の 1 秒あたりの増分数。	1 秒あたりの値：/s、K/s、M/s、G/s、または T/s
REPORT RATE は OFF。	その期間中の増分数。	#、K、M、G、または T
REPORT FORMAT は STRUCTURED。	その期間中の増分数。	

Measure では、32 ビットおよび 64 ビットの両方の増分カウンターをサポートしています。G シリーズの RVU では、一部の増分カウンターは 32 ビットバージョンおよび 64 ビットバージョンの 2 つのバージョンで存在します。カウンター名の末尾に-F が付いていれば 64 ビットバージョンであることがわかります。H シリーズ以降の RVU では、2 つのバージョンに存在していたすべてのカウンターは現在、64 ビットバージョンにのみ存在します。末尾が-F のカウンター名は、ZMS 形式には存在しませんが、Legacy 形式でデータを要求するアプリケーションで引き続き使用されます。

増分カウンターの例については、CLUSTER エンティティカウンター **MESSAGES-RECEIVED**(189 ページ)を参照してください。

最大キューカウンター（Legacy 形式のみ）

最大キューカウンターは、キューの最大長を追跡します。レポート形式のオプションは、最大キューカウンターには影響しません。

最大 64 の測定要求の間でカウンターレコードを共有するため、最大キュー値をリセットすることはできません。レコードがシステムに長く存在すると、最大キューカウンターの値は減少します。報告された値が、測定期間とは無関係になります。

Measure G11 以降の PVU では、最大キューカウンターはインストゥルメンテーションで追跡されません。これらは新しい形式のレコードから削除され、G11 以降のデータの Legacy 形式のレコードおよび表示では 1 と報告されます。

最大キューカウンターの例については、PROCESS エンティティカウンター（**MAX-MQCS-INUSE**(382 ページ)）を参照してください。

最大値カウンター

最大値カウンターは、カウンターレコードの有効期間中の特定の測定の最大値を追跡します。レポート形式のオプションは、最大値カウンターには影響しません。

DISCOPEN レコードと DISKFILE レコードの MAX-LOCKWAIT-TIME でのみ最大値カウンターが使用されます。単位は ms、sec、hr、day、または wk としてフォーマットされます。

最大値カウンターの例については、DISCOPEN エンティティカウンター（**MAX-LOCKWAIT-TIME**(257 ページ)）を参照してください。

キューカウンター

キューカウンターは、キューに入れられたすべての要素がキューで費やす時間の合計を測定します。

属性の設定	出力値	表示単位
REPORT RATE は ON。	この時間間隔の間にキューに入れられた要素の平均数（AQL）。キューカウンターでは、RATE ON は RATE OFF の単なる平均や割合ではありません。RATE ON は、キューで費やした時間の平均ではなく、キューの長さの平均を提供します。	10 進数の単位の AQL（キューの長さの平均）
REPORT RATE は OFF。	すべての要素がキューで費やした時間の合計で、秒またはミリ秒で表示されます。たとえば、あるプロセスが 2 秒間キューに入れられていて、別のプロセスが 3 秒間キューに入れられている場合、合計キュー時間は 5 秒です。	ミリ秒、秒、時間、日、または週
REPORT FORMAT は STRUCTURED。	すべての要素がキューで費やした時間の合計で、マイクロ秒で表示されます。	

キューカウンターの例は、CPU エンティティカウンター **CPU-QTIME**(202 ページ)を参照してください。

H シリーズ以降の RVU では、ビジーカウンターには以下の 2 種類があります。

- QUEUE カウンター、従来のタイマーで維持
- TCELLQUEUE カウンター、タイマーセルで維持

タイマーセルおよび関連するカウンターの概要については、**タイマーセルカウンター**(175 ページ)を参照してください。

キュービジーカウンター

キュービジーカウンターは、キューからアクセスされたリソースがビジーである時間を測定します。同時に複数の要求（I/O）をデバイスのキューに入れることができる ServerNet 環境での、デバイスのビジー時間を測定するために使用されます。

属性の設定	出力値	表示単位
REPORT RATE は ON。	期間中でのビジー時間の割合。	間隔の長さに対する割合
REPORT RATE は OFF。	同時にキューに入れられていた要求の数に関係なく、期間中にキューがビジーだった時間。たとえば、キューに 2 つのプロセスが同時に入れられている場合、両方のプロセスが 1 秒間キューに入れられていて、一方のプロセスがさらに 2 秒間キューに残るとすると、キュービジーの合計時間は 3 秒です。	ミリ秒、秒、時間、日、または週
REPORT FORMAT は STRUCTURED。	キューに要素が存在していた時間の合計で、マイクロ秒で表示されます。	

キュービジーカウンターの例は、DEVICE エンティティカウンター **READ-QBUSY-TIME**(222 ページ)を参照してください。

H シリーズ以降の RVU では、キュービジーカウンターには以下の 2 種類があります。

- QBUSY カウンター、従来のタイマーで維持
- TCELLQBUSY カウンター、タイマーセルで維持

タイマーセルおよび関連するカウンターの概要については、**タイマーセルカウンター**(175 ページ)を参照してください。

応答時間カウンター

応答時間カウンターは、端末からの読み取りからその端末への次の書き込みまでの間隔を、マイクロ秒単位で測定します。応答時間カウンターに値が追加されたときは常に、対応するトランザクションカウンターが加算されます。

属性の設定	出力値	表示単位
REPORT RATE は ON。	各トランザクションの平均応答時間。	ミリ秒、秒、時間、日、または週
REPORT RATE は OFF。	累積応答時間。	ミリ秒、秒、時間、日、または週
REPORT FORMAT は STRUCTURED。	累積応答時間で、マイクロ秒で表示されます。	

RATE ON でのレポートでは、応答時間カウンターはキュー時間カウンターの特殊なケースで、応答時間の累積値がトランザクションの数で除算されます。

応答時間カウンターの例は、TERMINAL エンティティカウンター **RESPONSE-TIME**(443 ページ)を参照してください。

ロック待機カウンター

ロック待機カウンターは、ファイルロックの待機に費やした時間を測定します。ロック待機カウンターは、トランザクションがロックである応答時間カウンターの特殊なケースです。

属性の設定	出力値	表示単位
REPORT RATE は ON。	各トランザクションの平均ロック時間。	ミリ秒、秒、時間、日、または週
REPORT RATE は OFF。	ロック時間。	ミリ秒、秒、時間、日、または週
REPORT FORMAT は STRUCTURED。	ロック時間はマイクロ秒で表されます。	

ロック待機カウンターの例は、DISCOPEN エンティティカウンター（**LOCKWAIT-TIME**(257 ページ)）を参照してください。

サンプリングカウンター

サンプリングカウンターは、PROCESSH コード範囲の概算のビジー時間を測定します。ランダムな時間間隔では、Measure サブシステムは、割り込みプロセスを調べるサンプリング割り込みをアクティブにします。プロセスが PROCESSH 測定中の場合、サンプリングプロシージャーは process-busy-samples カウンターを加算し、測定対象のコード領域がビジーの場合は code-space-busy-samples カウンターを加算して、適切な code-range-busy-samples カウンターを加算します。

属性の設定	出力値	表示単位
REPORT RATE は ON。	測定対象の実行中のコード領域が見つかったサンプリング割り込みの割合。	コード領域の合計サンプルの割合
REPORT RATE は OFF。	測定対象の実行中のコードが見つかったサンプリング割り込みの数。	#、K、M、G、または T
REPORT RATE は ON。	測定対象の実行中のコード領域が見つかったサンプリング割り込みの割合。	コード領域の合計サンプルの割合
REPORT FORMAT は STRUCTURED。	測定対象の実行中のコードが見つかったサンプリング割り込みの数。	

サンプリングカウンターの例は、PROCESSH エンティティカウンター **PROCESS-BUSY-SAMPLES**(397 ページ)を参照してください。

スナップショットカウンター

スナップショットカウンターは、測定の開始時や終了時などの特定の時点に行われた測定の値を返します。返された値に対して計算は行われません。単に報告するだけです。レポート形式のオプションは、スナップショットカウンターには影響しません。

スナップショットカウンターは、RATE を ON または OFF に設定することによる影響を受けません。単位は#、K、M、G、または T として表示されます。

スナップショットカウンターの例は、DISC エンティティカウンター **STARTING-FREE-SPACE**(241 ページ)および **ENDING-FREE-SPACE**(238 ページ)を参照してください。

Syslink カウンター

Syslink カウンターは、リモートシステムへのメッセージリンクを確立するのに要した時間を測定します。Syslink カウンターに値が追加されたときは常に、対応するリンクカウンターが加算されます。Syslink カウンターは、トランザクションがメッセージリンクである応答時間カウンターの特殊なケースです。

属性の設定	出力値	表示単位
REPORT RATE は ON。	各トランザクションの平均リンク時間。	ミリ秒、秒、時間、日、または週
REPORT RATE は OFF。	リンク時間。	ミリ秒、秒、時間、日、または週
REPORT FORMAT は STRUCTURED。	リンク時間、マイクロ秒で表されます。	

Syslink カウンターの例は、SYSTEM エンティティカウンター **LINKS**(438 ページ)および **LINK-TIME**(438 ページ)を参照してください。

タイマーセルカウンター

HPE Integrity NonStop NS シリーズシステム（H シリーズ以降の RVU を実行）ではプロセッサー間の多数決同期が行われる影響で、マイクロ秒単位で表示した場合に標準の時刻機能の精度が G シリーズの RVU で実行されるシステムよりも低くなります。

非常に精度の高い測定で、同じプロセッサーでの 2 つのプロセス間の送信や単純なメッセージ交換のコンテキストで完了できるイベントの場合、測定時間の値にゆがみが生じる可能性があります。このゆがみは、プロセッサー間のメッセージまたは I/O の完了を含むタイマーの実行が長くなるという問題ではありません。このような測定のエラーの度合いは、イベントの長さと比べれば取るに足らないことです。

Measure およびその他のアプリケーションに対してより正確なタイミングでサービスを提供するため、H シリーズ以降のアーキテクチャーではタイマーセルと呼ばれるメカニズムを備えています。このメカニズムを使用すれば、すべてのタイマー更新時にプロセッサーの厳密でない同期による損失を発生させることなく正確で精度の高いタイマーを維持できます。タイマーセル値が同期されるのは、周期的にコピーされるときや、要求側のアプリケーションに報告されるときだけです。

タイマーセルカウンターのタイプには、TCELLBUSY、TCELLQUEUE、および TCELLQBUSY があります。それぞれ、BUSY、QUEUE、および QBUSY カウンタータイプと同じインストルメンテーションロジックの下で動作します。タイマーセルカウンターのタイプをユーザー定義（USERDEF）エンティティとして指定することもできます。

L シリーズの RVU ではタイマーセル機能は無効で、**ビジーカウンター**(170 ページ)、**キューカウンター**(172 ページ)、および**キュービジーカウンター**(172 ページ)に置き換えられます。

データファイルのエラーの識別

測定されたエンティティタイプそれぞれの出力レコードの最初のフィールドは、問題により測定やレコード割り当てが妨げられいないかどうかを示します。

表 3: ZMS 形式のフォーマットエラーフィールド値

値	意味
ERROR = 0	レコード内のデータに関するエラーや警告はありません。
ERROR > 0	エラーが発生しました。データが存在していません。測定対象のオブジェクトは識別されていますが、次の理由のためにカウンターレコードが開始しませんでした。 1 = ERR^NOCIDENTRY — データレコードエラー：CID テーブルオーバーフロー 2 = ERR^NOCOUNTERSPACE — データレコードエラー：カウンター領域のオーバーフロー 3 = ALREADY^INUSE — データレコードエラー：サンプリングの競合 4 = ERR^XPTR^INSTALL — データレコードエラー：Measure 内部エラー 8 = ERR^NOTIMERCELLS — データレコードエラー：使用可能なタイマーセルなし
ERROR < 0	警告。データは存在していますが、例外が示されています。具体的な原因については、残りの値を参照してください。
.<1>	数値オーバーフロー。合計のレコード累積で、1 つまたは複数のカウンターフィールドがオーバーフローしました。
.<2>	サブシステムのバージョンが一致しません。レコードをフォーマットした Measure PVU は、このデータレコードを生成した装備済み製品サブシステムよりも以前のものです。内部レコードに、MEASFH でフォーマットできる分量以上のデータがあります。測定対象のシステムで Measure PVU を更新してください。
.<3>	テンプレートのバージョンが一致しません。データレコードを要求しているアプリケーションが古いバージョンの MEASDDLS でコンパイルされたか、構造化データファイルがレコードを生成した MEASFH バージョンよりも古いレコードテンプレートで作成されました。レポートに利用できるデータがさらにありますが、インターフェイスで渡せませんでした。アプリケーションを更新するか、構造化データファイルを新しいサイズに変換してください。
.<4>	タイマーセルがランデブーに失敗しました。
.<5:11>	今後使用するために予約されています。
.<12>	タイマーセルが割り当てに失敗しました。
.<13>	ZMSPROC レコードのみ。このプロセスの USERDEF インストゥルメンテーションが遅延しています。
.<14>	ZMSPROC レコードのみ。このプロセスの SQL インストゥルメンテーションが遅延しています。
<15>.	ZMSPROC レコードのみ。このプロセスの OSSNS インストゥルメンテーションが遅延しています。

0 または-1 の値は、レコードにデータがあることを示します。0 かどうかをテストすることによってデータの存在を判断する古いアプリケーションがある場合は、これらのアプリケーションを変更する必要があります。たとえば、エラー状態を示す正の値をテストします。

表 4: Legacy 形式のエラーフィールドの値

値	意味
0	問題ありません。
1	同時測定で許可されるエンティティの最大数を超えたために、データレコードは割り当てられませんでした。
2	許可される最大カウンター領域を超えたために、データレコードは割り当てられませんでした。
3	コードサンプリングがすでに進行中だったために、エンティティは測定されませんでした。PROCESSH エンティティタイプにのみ適用されます。
4	MEASCTL 内部エラーのために、エンティティは測定されませんでした。内部アドレッシング構造の XPTR が見つかりませんでした。
8	タイマーセルがこれ以上使用できなくなったために、データレコードは割り当てられませんでした。
65535	累積または増分のカウンターでフィールドがオーバーフローしました。データレコードは書き込まれますが、1 つまたは複数の 32 ビットフィールドの容量を超過しました。正確な値は、同等の 64 ビットのフィールドに含まれています。これは警告であり、エラーではありません。
65534	(G シリーズの RVU) バイト数フィールドのオーバーフロー。1 つまたは複数の 16 ビットのバイト数フィールドの容量を超過しています。バイト数データについては、同等の 32 ビットフィールドを参照してください。これはエラーではなく警告です。DISC エンティティタイプにのみ適用されます。

一般的なエンティティヘッダーフィールド

これらのフィールドはすべてのエンティティレコードのヘッダーに現れ、各エンティティの説明で繰り返される代わりにここで 1 回定義されます。

DDL ヘッダー（ZMS 形式）

注記:

処理可能な形式に対応する外部レコードを取得するには、**TEMPLATE-VERSION**(180 ページ)を常に使用してください。

これを行うには、3 つの方法があります。

- 現在のバージョンのテンプレートバージョンを使用する MEASCOM を使用します。
- 要求されたテンプレートバージョンを使用して、ユーザープログラムから `Measread_diff_()` を呼び出します。
- 構造化ファイルの場合、問題になっているエンティティに対して要求されたテンプレートバージョンに対応するレコード長を持つ事前に作成された構造化ファイルを使用します。異なる製品バージョンのシステムで収集されたデータが同じ構造化ファイルに保存された場合、すべてのレコード用のテンプレートバージョンが、構造化ファイルを作成するために使用されるものになります。最初のレコードのテンプレートバージョンが、ファイル内のすべてのレコードのフォーマットを設定するために使用されます。

要求されたテンプレートバージョンで指定したバージョンに存在しないフィールドがデータファイル内にある場合、それらは要求されたレコード内に存在することはありません。

要求されたテンプレートバージョンによって示されるバージョンのフィールドが、データファイル内に存在しない場合、フィールドはゼロが入力されて要求されたレコード内に存在し、**DATA-VERSION**(179 ページ)フィールドを使用して、ゼロが埋められたこれらのフィールドが有効かどうかを判断することができます。

```
RECORD zms entity. FILE is "zms entity" ENTRY-SEQUENCED.
 02 loadid                      type character 8.
 02 load-id                     redefines loadid.
    03 prefix-id                type character 4.
    03 interval-id              type character 4.
 02 system-name                 type character 8.
 02 cpu-num                     type binary 16 unsigned.
 02 os-version.
    03 letter                   type character 1.
    03 number                   type binary 8.
 02 reserved                    type character 4.
 02 object-uid                  type character 16.
 02 from-timestamp              type binary 64.
 02 to-timestamp                type binary 64.
 02 delta-time                  type binary 64.
 02 template-version            type binary 64.
 02 format-version.
    03 letter                   type character 1.
    03 number                   type binary 8.
 02 data-version.
    03 letter                   type character 1.
    03 number                   type binary 8.
 02 subsystem-version           type binary 16 unsigned.
 02 error                       type binary 16.
end
```

DDL ヘッダー（Legacy 形式）

```
RECORD entity. FILE is "entity" ENTRY-SEQUENCED.
 02 error                       type binary 16 unsigned.
```

```
* time items:
 02 from-timestamp             type binary 64.
 02 to-timestamp               type binary 64.
 02 delta-time                 type binary 64.

* measurement identification items:
 02 system-name                type character 8.
 02 os-version.
    03 letter                   type character 1.
    03 number                   type binary 8.
 02 loadid                     type character 8.
 02 load-id                    redefines loadid.
    03 prefix-id                type character 5.
    03 interval-id              type character 3.
 02 cpu-num                    type binary 16 unsigned.
```

DDL ヘッダーフィールドの定義

フィールドはアルファベット順に一覧表示されます。

CPU-NUM

測定が実行された CPU の数。プロセッサー間通信（IPC）レコードの場合、CPU-NUM は MYCPU です。

DATA-VERSION

（ZMS 形式のみ）このレコードを生成した Measure データファイルを収集したシステムの Measure 製品バージョンと、データファイルと互換性のある MEASFH 製品バージョン。このフィールドは、LETTER と NUMBER の 2 つのサブフィールドに分かれています。

DELTA-TIME

マイクロ秒単位の測定の期間。

ERROR

エラーによって、Measure がデータを収集したり、このエンティティの出力レコードを書き込んだりすることを妨げていないかどうかを示します。値については、**データファイルのエラーの識別**(175 ページ)を参照してください。

FORMAT-VERSION

（ZMS 形式のみ）エンティティレポートまたは構造化レコードをフォーマットしたプログラムの Measure 製品バージョン。このフィールドは、LETTER と NUMBER の 2 つのサブフィールドに分かれています。

古いデータファイルがフォーマットされており、Measure 製品バージョンが変更していない場合、フォーマットバージョンはプログラムのバージョンではなく、データファイルのバージョンになる可能性があります。

FROM-TIMESTAMP

測定または一時的なエンティティの開始時刻。ユリウス暦ベースの地方平均時（LCT）として指定します。

CSV、JSON、または XML レポート形式を使用する場合、タイムスタンプは、ISO 8601 の日付と時刻の表現である、秒の小数部を持つ拡張形式の現地時間（YYYY-MM-DDThh:mm:ss.s）として発行されます。

LOAD-ID

構造化ファイルにのみ使用されます。このレコードが構造化ファイルに書き込まれるときに MEASCOM LIST コマンドの LOADID 句を使用した場合、このフィールドには、レコードの一意の識別文字列が含まれます。LOADID 句を使用しない場合、このフィールドには空白が含まれます。デフォルトは空白です（LOADID なし）。

構造化レコード出力を要求するときに、ユーザーが入力した値です。LOADID は 8 文字長の文字列で、英字、数字、キャレット（^）、ハイフン（-）、およびアンダースコア（_）を含められます。最初の文字は、英字でなければなりません。指定した文字列が 8 文字に満たない場合、文字列をスペースで埋めることも、生成された間隔シーケンス番号を含めることもできます。

このフィールドは、PREFIX-ID と INTERVAL-ID の 2 つのサブフィールドに分かれています。

Legacy 形式のレコードで LISTALL コマンドとともに使用されるときに、LOADID が 5 文字以下の場合、LOADID は、データレコードが表す間隔を示すシーケンス番号が付加されて生成されます。日付間隔は、1 から順番に番号が付けられます。LOADID が 3 文字以下の場合、生成される LOADID にはスペースが含められます。

ZMS 形式の構造化レコードの LISTALL 要求で、間隔番号付けが必要な場合は、4 文字の LOADID を指定してください。5 文字の LOADID を使用する場合、5 番目の文字は、フィールドの INTERVAL-ID 部分に置かれます。生成された LOADID は LISTALL データにはあまり使用されません。この機能を使用するアプリケーションを評価して、今後、4 文字の LOADID を指定する必要があるか、5 文字の LOADID を指定する必要があるかを判断してください。

LOADID

再定義されています。**LOAD-ID**(180 ページ)を参照してください。

OBJECT-UID

（ZMS 形式のみ）今後使用するために予約されています。

OS-VERSION

測定が行われたときのオペレーティングシステムのバージョン ID。このフィールドは、LETTER と NUMBER の 2 つのサブフィールドに分かれています。

SUBSYSTEM-VERSION

（ZMS 形式のみ）この値は、すべてのサブシステムの初期の H シリーズ以降のバージョンでは 1 に設定されます。値が 1 より大きい場合は、サブシステムのドキュメントで Measure カウンターに影響する変更に関する情報を確認してください。G シリーズの RVU では、この値は 0 です。

SYSTEM-NAME

測定が行われたシステムの名前。

TEMPLATE-VERSION

（ZMS 形式のみ）外部レコードの形式の、MEASDDLS からのレコードテンプレートバージョン。異なる製品バージョンのシステムで収集されたデータを同じ構造化ファイルに保存する場合、構造化ファイルを作成するために使用されるのは、すべてのレコード用のテンプレートバージョンです。最初のレコードのテンプレートバージョンが、ファイル内のすべてのレコードの形式を設定するために使用されます。

TO-TIMESTAMP

測定または一時的なエンティティの終了時刻。ユリウス暦ベースの地方平均時（LCT）として指定します。

CSV、JSON、または XML レポート形式を使用する場合、タイムスタンプは、ISO 8601 の日付と時刻の表現である、秒の小数部を持つ拡張形式の現地時間（YYYY-MM-DDThh:mm:ss.s）として発行されます。

オープンシステムサービス（OSS）に対する Measure のサポート

OSS 環境と Guardian 環境の間の相互運用性について詳しくは、以下を参照してください。

- *Open System Services Library Calls Reference Manual*
- *Open System Services Management and Operations Guide*
- *Open System Services Shell and Utilities Reference Manual*
- *Open System Services System Calls Reference Manual*
- *オープンシステムサービスユーザーガイド*

OSS ファイルパス名の処理

- OSS ファイルパス名は常に二重引用符（" "）で囲んで入力および表示され、大文字と小文字が区別されます。OSS ファイルパス名は、特定のファイルを指すことも、特定のディレクトリ内の一連のファイルを指すこともあります。ディレクトリを指定した場合、そのディレクトリ直下のファイルだけが含まれます。指定したディレクトリの下位のディレクトリにあるファイルは含まれません。
- Guardian 環境と OSS 環境の両方をサポートするために、pname の使用と意図された意味が標準化されています。pname は、従来からの Guardian プロセス名の短縮形であり、OSS シェルコマンドでもあります。
  - 二重引用符で囲まれた pname（"pname"）は OSS ファイルパス名です。
  - 前にドル記号が付いた pname（$pname）や単独の pname（pname）は Guardian プロセス名です。
- OSS ファイルパス名は長くすることができます（最大長 1023 文字）。このような長い OSS ファイルパス名を表示する場合、Measure は最初の 80 文字を表示して、残りの文字を次の行に折り返します。Measure は、OSS ファイルパス名全体が表示されるまで 80 文字ずつ名前を表示します。さらに、このような長い OSS ファイルパス名にカットアンドペースト機能を使用した場合、80 文字行のそれぞれの末尾にある不要なキャリッジリターン文字を選択しないように、名前を部分ごとにカットアンドペーストする必要があるかもしれません。
- POSIX 標準では、OSS ファイルパス名で印刷できない（表示できない）文字を使用できます。ただし、これらの文字を使用する名前は、ユーザーインターフェイス、ワークステーション、および端末環境での差異に従って別々に処理されることがあります。OSS ユーティリティでも Measure でもこの問題に対処しません。
- Measure G09 以降の PVU は、OSS ファイルパス名のエンティティサポートを提供しますが、パス名はバッファーの上限を超える可能性があるので、測定の指定で OSS ファイルパス名を使用するときには注意してください。詳細な測定の場合は、測定をいくつかの別々の測定構成に分割します。Measure は最大 64 の同時測定をサポートしています。
- OSSPID および PID の意味を混同しないでください。
  - OSSPID は、多くの Measure エンティティレポートに表示される数値であり、PID は、CPU、PIN から構成される Guardian プロセス識別子です。
  - PID は Measure コマンドインターフェイスで広く使用されていますが､Measure コマンドインターフェイスは OSSPID をサポートしていません。
- OSS ファイルパス名はヌル終端された文字列です。Measure では、文字列フィールドの長さが必要であり、ヌル文字を含みます。

Guardian ファイル名の再利用

Guardian ファイル名の値は、一般に i ノード番号と呼ばれる、OSS ファイルシステムでの内部値またはセルに基づきます。ファイルとそのファイルに対するすべての参照を削除すると、i ノード番号とファイル名の再利用を行えます。

ファイルがディスク上で作成されると、一意の作成バージョンシリアル番号（CRVSN）がファイルに割り当てられます。CRVSN は、Guardian ファイル名のさまざまなインスタンスを区別するためのメカニズムを提供します。Measure はこのメカニズムを使用して、i ノード番号の再利用によるファイル名解決の問題を回避します。

Guardian ファイル名と CRVSN 値が、OSS ネームサーバーデータで現在関連付けられている値に一致しない場合、ファイル名の再利用が起きたと見なされます。ファイル名の再利用が行われると、関連付けられた OSS ファイルパス名は表示されず、`*** Translation Not Available ***` というメッセージが生成されます。

作成バージョンシリアル番号（CRVSN）の処理

作成バージョンシリアル番号（CRVSN）は、48 ビット値です。Measure は、Guardian 名に続く最大 15 文字の 10 進数文字列として CRVSN を表示します（たとえば$DATA01.ZYQ00001.Y00003D2:**340359**）。Measure は、6 文字の文字列（48 ビット）として CRVSN 値をデータレコードに格納します。

Measure ID（MID）

OSS ファイルパス名は、PATHID と呼ばれるデータ構造体の中に含まれています。PATHID は内部的なもので、制御ブロックの中で渡されます。OSS ファイルパス名が外部化されている場合は、ファイルパス名変換のために OSS ネームサーバーに渡されます。

OSS ファイルパス名のレポート機能をサポートするため、Measure は、各 OSS ファイルパス名の PATHID および CRVSN 情報を取得し、その情報を Measure データに保存します。Measure データ内の PATHID および CRVSN の値を結合したものが Measure ID（MID）です。MID 情報と内部ファイル名の表現は圧縮されているため、MID データを追加しても Measure データファイルのサイズは大きくなりません。

OSS ジャーナルセグメント

OSS ジャーナルセグメントは、Measure データの収集のために取り込まれた OSS ファイルパス名変換のセットです。これは、そのような変換が OSS ネームサーバーから使用できなくなったときに、Measure データファイルにあるデータレコードの OSS ファイルパス名変換を行います。OSS ネームサーバーが OSS ファイルパス名を変換できない原因は、時間の制約と、別のノードまたはネットワークへのデータファイルの再配置のためです。

Measure の OSS ファイルパス名のサポートの主な機能は、OSS ジャーナルセグメントの作成です。OSS ジャーナルセグメントによって、システム間と長期にわたる OSS ファイルパス名変換の移植性が保証されます。また、これによって、Measure データの OSS オブジェクトについての分析やレポートのために、測定されるシステムの OSS ネームサーバーにアクセスする必要もなくなります。

Measure データの OSS ファイルパス名を使用または表示するときや、ローカルシステム上の Measure データを分析するときは、OSS ジャーナルセグメントの構築は必要ありません。OSS ジャーナルセグメントの構築が必要になるのは、OSS ファイルパス名変換が構造化された Measure データ出力でレポート作成アプリケーションを使用できる必要があるときです。

Measure は、測定中に OSS ジャーナルセグメントを構築して、測定シャットダウン時に Measure データファイルに挿入します。分析のために Measure データファイルが開かれると、ジャーナルセグメントがメモリにロードされ、アプリケーションプロセスまたは MEASCOM に接続されます。ファイルパス名の変換の前に測定中のファイルが削除された場合、OSS ジャーナルセグメントはこのファイルに対応するファイルパス名を持たなくなり、データ分析には使用できなくなります。

この機能はオプションです。デフォルトはジャーナルセグメントなしです。

測定が終了する前に OSS ファイルが削除された場合は、OSS ジャーナルに変換が取り込まれないことがあります。その結果、MEASCOM リストに次のメッセージが生成される場合があります。*** Translation - Not Available ***.

エンティティレポートの形式

エンティティレポートの長さは一様ではありません。これは、OSS ファイルのパス名の表示において、1 行に表示できるよりも名前が長くなる場合、いくつかの行がレポートに追加されることがあるためです。

Measure エンティティレポートのスクロールを制御するには、PAGESIZE コマンドを使用します。PAGESIZE によって、別のプロンプトが出される前に表示する行の数を指定することができます。詳しくは、**PAGESIZE**(128 ページ)を参照してください。

ANSI SQL 名に対する Measure のサポート

テーブルやインデックスなどの SQL/MX オブジェクトは、基礎となる Guardian 名の他に ANSI SQL 名を持ちます。Measure G11 以降の PVU は、特定の Measure の出力に ANSI SQL 名を指定できます。

ANSI SQL 名および SQL/MX について詳しくは、*SQL/MX リファレンスマニュアル*および *SQL/MX インストレーションおよび管理ガイド*を参照してください。

ANSI SQL 名の処理

- ANSI SQL 名は常に表示され、一重引用符（'）で囲まれ、大文字と小文字が区別されます。二重引用符（"）で囲まれた ANSI SQL の区切られた識別子は、一重引用符の内部で使用できます。次に例を示します。

  ```
  'Catalog_12.Schema_34.Table_56'
  'Catalog_12."A.B.C PARTITION D$"" <>".Table_56'
  ```

  注記:

  ANSI SQL 名を呼び出し可能プロシージャーに引数として渡すときには、一重引用符で囲まないでください。**Measure 呼び出し可能プロシージャー**(458 ページ)を参照してください。

- ANSI SQL 名は、特定のオブジェクトを指す場合も、特定のスキーマおよびカタログ内の一連のオブジェクトを指す場合もあります。カタログが指定されている場合、そのカタログ直下のファイルだけが含まれます。指定されたカタログに従属したスキーマ内に格納されているファイルは含まれません。
- Measure は、二重引用符に囲まれていない一重引用符内のすべての空白文字（1 つまたは複数の空白）を圧縮します。
- Measure は、必要に応じて ANSI SQL 名をある行から次の行に折り返して表示します。折り返された ANSI SQL 名を Measure 表示からカットアンドペーストした場合、名前の一部ではないキャリッジリターン文字が含まれる場合があります。これを回避するには、一度に 1 行ずつカットアンドペーストしてください。
- 132 文字を超えるコマンドを入力するには、継続文字（「&」）を使用してコマンド入力行を小さく分割する必要があります。アンパサンド文字で終了するテキスト文字列を入力すると、次の入力行は入力の継続として扱われます。このように入力された連続した入力行は、1 つの文字列として入力された場合と同様に扱われます。次に例を示します。

  ```
  + ADD DISKFILE 'long_long_long_long_long_long_long_long_&amp;
  & long_catalog.short_schema.short_object PARTITION&amp;
  & short_partition2'
  ```

- ANSI SQL 識別子は長くできます（最大 128 文字）。区切られた形式での識別子の最大長は 258 文字です（識別子を区切る 2 つの二重引用符と、128 個の二重引用符文字から構成される識別子を表すための 256 個の二重引用符文字）。
  - 名前空間修飾子のない ANSI SQL オブジェクト名は、区切られない場合の最大長が 386 文字です（128 文字と 2 つのピリオドから構成される、3 つの区切られない識別子）。区切られた形式では、オブジェクト名の最大長は 776 文字です（258 個の文字と 2 つのピリオドから構成される、3 つの区切られた識別子）。
  - 名前空間修飾子のある ANSI SQL オブジェクト名は、区切られていない場合の最大長が 393 文字です（MODULE キーワードの 7 つの追加文字と 1 つのスペース）。区切られた形式では、オブジェクト名の最大長は 783 文字です（7 プラス 776）。

- 名前空間修飾子のない SQL/MX パーティション名は、区切られていない場合の最大長が 525 文字です（128 文字の区切られていない識別子フィールド 4 つ、2 つのピリオド、PARTITION キーワードの 9 文字、および 2 つのスペース）。区切られた形式では、パーティション名の最大長は 1045 文字です（258 文字の区切られた識別子 4 つ、2 つのピリオド、PARTITION キーワードの 9 文字、および 2 つのスペース）。
- 名前空間修飾子のある SQL/MX パーティション名は、区切られていない場合の最大長が 531 文字です（スペースが続いた TABLE または INDEX のキーワードを表す 6 つの追加文字）。区切られた形式では、パーティション名の最大長は 1051 文字です（6 プラス 1045）。
- 区切られた形式で表された最大長のパーティション名の場合、ANSI SQL 名の最大長は 1051 文字です。

SQL/MX は、これらの理論上の制限をすべてサポートしているわけではありません。SQL/MX が構文エラーを返すと、Measure は構文エラーを報告します。*SQL/MX Reference Manual* を参照してください。

SQL ジャーナルセグメント

測定データファイルには、ANSI SQL、Guardian 名、および ANS UID 間の変換に使用するための SQL ジャーナルセグメントを含めることができます。SQL ジャーナルセグメントによって、システム間と長期にわたる SQL 名変換の移植性が保証されます。

ジャーナルセグメントには、SQL/MX のオブジェクトとパーティション名、Guardian 名、ANS UID、ANSI 名前空間、SQLMID（SQL/MX ファイルの特定のインスタンスを表す内部 Measure ハンドル）などの情報が含まれます。

SQL ジャーナルセグメントの構築は、測定が行われたシステムで Measure データを分析する際は必要ありません。SQL ジャーナルセグメントの構築が必要になるのは、構造化された Measure データ出力でレポート作成アプリケーションが名前変換を使用できる必要があるときです（**MEASLISTEXTNAMES**(491 ページ)を参照）。

Measure は、測定中に SQL ジャーナルセグメントを構築して、測定のシャットダウン時に Measure データファイルに挿入します。分析のために Measure データファイルが開かれると、ジャーナルセグメントがメモリにロードされ、アプリケーションプロセスまたは MEASCOM に接続されます。SQL 名の変換の前に測定中のファイルが削除された場合、SQL ジャーナルセグメントはこのファイルに対応する SQL 名を持たなくなり、データ分析には使用できなくなります。

この機能はオプションです。デフォルトはジャーナルセグメントなしです。

ANSI SQL オブジェクト名には完全修飾名または部分修飾名を指定できます。部分修飾の場合、省略されたカタログとスキーマのフィールドは、MEASCOM セッション環境値を使用して解決されます（**SQLCATALOG**(149 ページ)および **SQLSCHEMA**(149 ページ)を参照）。

エンティティレポートの形式

エンティティレポートの長さは一様ではありません。これは、ANSI SQL 名の表示において、1 行に表示できるよりも名前が長くなる場合、いくつかの行がレポートに追加されることがあるためです。

Measure エンティティレポートのスクロールを制御するには、PAGESIZE コマンドを使用します。PAGESIZE によって、別のプロンプトが出される前に表示する行の数を指定することができます。詳しくは、**PAGESIZE**(128 ページ)を参照してください。

DLL に対する Measure のサポート

Measure G12 以降の PVU は DLL の測定をサポート

コード領域指定

位置独立コード（PIC）で実装される DLL の導入により、以下に示す従来のコード領域モデルは適用されなくなりました。

- ユーザーコード
- ユーザーライブラリ
- システムコード
- システムライブラリ

レガシーの TNS コード（高速化 TNS コードを含む）を除き、コード領域は PROCESSH エンティティを追加する場合に不要になり、PROCESSH データを表示するコマンドで表示されなくなりました。

以下の MEASCOM コマンドは、この変更による影響を受けます。

- ADD PROCESSH コマンドは、レガシーの TNS コード（高速化 TNS コードを含む）を除いてコード領域を指定する必要がなくなりました。
- 以下のコマンドの出力は、レガシーの TNS コード（高速化 TNS コードを含む）を除いて、コード領域ではなくコードファイルの周りに編成されます。
  - INFO PROCESSH
  - INFO MEASUREMENT
  - LIST PROCESSH
  - LISTALL PROCESSH

集計データの測定

DLL 実行に関する情報を収集、分析、および表示するために、Measure は同じバージョンのライブラリコードを実行しているすべてのプロセスにわたって、ライブラリコードについての集計データの収集と表示をサポートするようになりました。この機能強化には、以下の MEASCOM コマンドを通じて TOTALS 属性を PROCESSH エンティティに拡張することが必要でした。

- **LIST entity-type**(75 ページ)
- **LISTALL entity-type**(115 ページ)
- **RESET REPORT**(130 ページ)
- **SET REPORT**(137 ページ)

以下の設定を調整することができます。

- TOTALS ONLY

  集計カウントのみが表示されます。
- TOTALS INCLUDE

  プロセスごとのカウントと集計カウントの両方が表示されます。
- TOTALS SUPPRESS

  集計カウントは表示されません。

測定で指定されたコードを実行しているプロセスが 1 つだけの場合、プロセスごとのカウントと集計カウントは同じです。

このマニュアルの DLL に関する追加情報

コマンドインターフェイス

- **LIST entity-type**(75 ページ)
- **LISTALL entity-type**(115 ページ)
- **LISTACTIVE entity-type**(104 ページ)
- **RESET REPORT**(130 ページ)

- **SET REPORT**(137 ページ)
- **SHOW REPORT**(148 ページ)

情報の表示

- **INFO entity-type**(68 ページ)
- **INFO MEASUREMENT**(71 ページ)
- **LIST entity-type**(75 ページ)
- **LISTACTIVE entity-type**(104 ページ)
- **LISTALL entity-type**(115 ページ)

エンティティ指定

PROCESSH(387 ページ)

外部レコードの定義

- **PROCESSH エンティティの DDL レコード（ZMS 形式）**(391 ページ)にある PROCESSH 外部レコード定義を参照してください。
- **PROCESSH エンティティの DDL レコード（ZMS 形式）**(391 ページ)にある zmsproch-id.外部レコード定義を参照してください。

呼び出し可能プロシージャー

MEAS_CODERANGENAME_DEMANGLE_(468 ページ)

ZMS 形式のレコードへのアクセス（MEASDDLZ）

MEASDDLZ ファイルにより、アプリケーションは、影響を最小限に抑えて ZMS 形式のデータレコードにアクセスできます。

ZMS 形式のレコード構造体は、レコードの各コンポーネントに命名レベルを追加します。Legacy 形式には、この追加の命名レベルはありません。たとえば、Legacy 形式のレコードのフィールド cpu.dispatches は、ZMS 形式のレコード構造体の zmscpu.ctr.dispatches と同じです。

既存のアプリケーションおよび ENFORM レポートをレガシー命名形式から ZMS 形式の命名形式に変換することが、レコード参照の重要な編集になります。これには、プログラムで参照される可能性のあるフィールドと、2 つのサイズで表示された同等のフィールドの考えられる組み合わせ（process.sent-bytes と process.sent-bytes-f など）の削除も伴います。

このプロセスを簡素化するには、MEASDDLZ ファイルを使用します。

MEASDDLZ は、Legacy 形式のレコード命名規則を使用して、ZMS 形式のカウンター幅とレコード内の位置を指定します。レコードテンプレート名では、Legacy 形式の名前が使用されます（たとえば、ZMSPROC ではなく PROCESS）。組み合わされたフィールドは再定義により参照されます。

MEASDDLZ および ZMS 形式のフォーマットは、Measure G11 PVU で利用できます。H01 以降の PVU では、ZMS 形式のインターフェイスがデフォルトです。レガシーインターフェイスは機能強化されず、MEASDDLZ は ZMS 形式のレコードに変更を反映するように維持されなくなりました。カウンターは ZMS 形式のレコードでのみ追加または変更されます。

CLUSTER

CLUSTER エンティティは、ローカルシステム上のすべてのプロセスによって送信および受信される FOX メッセージの数に関する情報を提供します。このマニュアルでは、FOX は以下のことを指します。

コンポーネント	サーバー
FOX II	Cyclone または VLX
TorusNet	NonStop K シリーズ
Servernet/FX	NonStop S シリーズ

FOX リンクには、リンクを開始したリンカーと、リンクを受け入れたリスナーの少なくとも 2 つのプロセスが別々のシステムに含まれます。Measure サブシステムは、ローカルシステムしか測定できないので、 測定データは、各 FOX リンクの片側（リンカーまたはリスナー）だけを反映し、両方は反映しません。

トピック
CLUSTER エンティティのエンティティ指定構文
CLUSTER エンティティの DDL レコード
CLUSTER エンティティの DDL フィールドの定義
CLUSTER エンティティの使用上の注意

CLUSTER エンティティは、H シリーズ以降の RVU と関係ありません。H シリーズ以降の RVU に類似したエンティティは、**SERVERNET**(399 ページ)です。

CLUSTER エンティティのエンティティ指定構文

CLUSTER エンティティを記述するには、次の構文を使用します。

```
CLUSTER entity-spec [, entity-spec ] ...
```

CLUSTER

FOX トラフィックに関する情報を収集します。

entity-spec

次のように指定します。

```
{ * | \system[ ( cpu ) ] }
```

指定する値は次の通りです。

*****

ローカルシステムとすべてのリモートシステム間の FOX トラフィックを指定します。

system

システム名またはリモートシステムの番号です。*system* を指定した場合、測定にはローカルシステムとこのリモートシステムの間のすべての FOX トラフィックが含まれます。

cpu

ローカルシステム上の CPU の番号です。*cpu* を指定した場合、測定にはこの CPU が受信または送信するすべて FOX メッセージが含まれます。デフォルトは、すべての CPU です。

CLUSTER エンティティの DDL レコード（ZMS 形式）

CLUSTER エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsclstr-id.
 02 remote-system-name         type character 8.
 02 remote-system              type binary 16 unsigned.
 02 reserved-1                 type character 6.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsclstr-ctrs.
 02 messages-sent              type binary 64.
 02 sent-bytes                 type binary 64.
 02 returned-bytes             type binary 64.
 02 messages-received          type binary 64.
 02 received-bytes             type binary 64.
 02 reply-bytes                type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsclstr. FILE is "zmsclstr" ENTRY-SEQUENCED.
 02 hdr                        type zmsheader.
 02 ctr                        type zmsclstr-ctrs.
 02 id                         type zmsclstr-id.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

CLUSTER エンティティの DDL レコード（Legacy 形式）

CLUSTER エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G シリーズエンティティは、次の点を除いて D シリーズエンティティのレコードと同じです。

- フィールドのオーバーフローの可能性を減らすために、長い byte-count フィールドが提供されます。32 ビットの各 byte-count フィールドには、対応する 64 ビットがあります。64 ビットのフィールドの使用については、**G シリーズの CLUSTER エンティティの使用上の注意**(191 ページ)を参照してください。
- ERROR フィールドが、32 ビットの byte-count フィールドのフィールドオーバーフローを示すことがあります。

```
RECORD cluster. FILE is "cluster" ENTRY-SEQUENCED.
                        .
                        .
                        .
(error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド (177 ページ) を参照してください)
                        .
                        .
```

```
                    .
* entity identification items:
 02 remote-system              type binary 16 unsigned.
 02 remote-system-name         type character 8.

* counter value items:
 02 messages-sent              type binary 32 unsigned.
 02 sent-bytes                 type binary 32 unsigned.
 02 returned-bytes             type binary 32 unsigned.
 02 messages-received          type binary 32 unsigned.
 02 received-bytes             type binary 32 unsigned.
 02 reply-bytes                type binary 32 unsigned.

* F40 counter value items:
 02 sent-bytes-f               type binary 64.
 02 returned-bytes-f           type binary 64.
 02 received-bytes-f           type binary 64.
 02 reply-bytes-f              type binary 64.
end
```

CLUSTER エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。

REMOTE-SYSTEM

FOX トラフィックが測定されたリモートシステムのシステム番号。

REMOTE-SYSTEM-NAME

FOX トラフィックが測定されたリモートシステムのシステム名。

CLUSTER エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。

MESSAGES-RECEIVED

ローカルシステムで実行中のすべてのリスナープロセスによって受信された FOX メッセージ要求の数。

カウンタータイプ：増分。

MESSAGES-SENT

ローカルシステムで実行中のすべてのリンカープロセスによって送信された FOX メッセージの数。

カウンタータイプ：増分。

RECEIVED-BYTES

ローカルシステムで実行中のすべてのリスナープロセスによって受信された FOX メッセージバイトの数。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。

G シリーズの RVU では、RECEIVED-BYTES-F フィールドが RECEIVED-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

REPLY-BYTES

リモートのリンカープロセスからのメッセージに応答して、ローカルシステムで実行中のすべてのリスナープロセスから送信された FOX メッセージのバイト数。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。

G シリーズの RVU では、REPLY-BYTES-F フィールドが REPLY-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

RETURNED-BYTES

ローカルシステムで実行中のすべてのリンカープロセスによって受信された FOX メッセージバイトの数。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。

G シリーズの RVU では、RETURNED-BYTES-F フィールドが RETURNED-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

SENT-BYTES

ローカルシステムで実行中のすべてのリンカープロセスによって送信された FOX メッセージのバイト数。

プロセスが FOX メッセージを受信または送信すると、そのプロセスの PROCESS レコードの MESSAGES-SENT または MESSAGES-RECEIVED カウンターが加算されます。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。

G シリーズの RVU では、SENT-BYTES-F フィールドが SENT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

CLUSTER エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードだけで使用されます。

フィールドはアルファベット順に一覧表示されます。

RECEIVED-BYTES-F

G シリーズの RVU では、RECEIVED-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

REPLY-BYTES

G シリーズの RVU では、REPLY-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

RETURNED-BYTES-F

G シリーズの RVU では、RETURNED-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

SENT-BYTES-F

G シリーズの RVU では、SENT-BYTES と同じですが、より大きい値（32 ビットではなく 64 ビット）に対応します。

すべての CLUSTER エンティティの使用上の注意

- CLUSTER カウンターは、FOX リンク経由で別のシステムに直接データを送信するネットワーク操作の実行中に、メッセージシステムによって加算されます。

  CLUSTER は、ネットワーク回線ハンドラーを介して送信されるメッセージはカウントしません。そのようなメッセージは、NETLINE および SYSTEM カウンターレコードでカウントされます。FOX リンクで接続されているシステムの場合、ネットワーク回線ハンドラーを介して送信されるのは、セキュリティチェックを必要とするメッセージのみです。

  たとえば、FUP COPY を使用して、FOX リンクで接続されているリモートシステムにファイルをコピーすると仮定します。ファイルのセキュリティをチェックする必要があるため、OPEN メッセージはネットワーク回線ハンドラーを介して送信され、NETLINE および SYSTEM カウンターによってカウントされます。ファイルデータは FOX 回線を介して送信され、CLUSTER エンティティによってカウントされます。
- プロセスが FOX メッセージを受信または送信すると、そのプロセスの PROCESS レコードの MESSAGES-SENT または MESSAGES-RECEIVED カウンターが加算されます。
- FOX による転送と Expand 回線ハンドラーについて詳しくは、*Expand Configuration and Management Manual* および *Expand Network Management and Troubleshooting Manual* を参照してください。

H シリーズ以降の CLUSTER エンティティの使用上の注意

基礎となるコンポーネントがシステムでサポートされなくなったため、H シリーズ以降の RVU では CLUSTER エンティティがサポートされません。CLUSTER エンティティを設定しても、MEASCOM は構文エラーを返しません（そのため、CLUSTER エンティティを設定する古いスクリプトをすぐに変更する必要はありません）。ただし、レコードは設定されず、返されません。

G シリーズの CLUSTER エンティティの使用上の注意

64 ビットのバイトカウントフィールド（-F で終わるフィールド）は、従来の 32 ビットのバイトカウントフィールドと同じデータを収集します。たとえば、64 ビットフィールドの SENT-BYTES-F は、32 ビットフィールドの SENT-BYTES と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

32 ビットフィールドはまだアクティブであり、引き続き値を返します フィールドのオーバーフローが発生しない場合、32 ビットフィールドと 64 ビットのフィールドは同じ値を返します。32 ビットフィールドがオーバーフローした場合、対応する 64 ビットフィールドは正しい値を返し、32 ビットフィールドは-1 の値を返します。測定対象のエンティティの ERROR フィールドも、オーバーフロー状態を示す-1 を返します。

64 ビットフィールドを使用するには、アプリケーションを変換してください。32 ビットのフィールドは、今後の PVU では使用できなくなる可能性があります。

MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットフィールドの名前を使用してください。たとえば、LIST DEVICE BY SENT-BYTES-F ではなく LIST DEVICE BY SENT-BYTES のコマンドを発行してください。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。

CONTROLLER

CONTROLLER エンティティは、D シリーズ RVU を実行中のシステムの、ディスク、テープ、端末、およびその他のデバイスについて、コントローラーの I/O アクティビティを測定します。Measure D45 以降の PVU で、デフォルトでは CONTROLLER エンティティは無効になっています。このデフォルト設定について詳しくは、**CONTROLLER エンティティの使用上の注意**(194 ページ)を参照してください。

Measure サブシステムを開始する前に、CONTROLLER エンティティを有効にするには、以下のようにします。

```
ADD DEFINE =_MEASURE_CONTROLLERS, FILE A
```

G シリーズ RVU を実行中のシステムで、ServerNet アドレス指定可能コントローラー（SAC）を測定するには、**SERVERNET**(399 ページ)を参照してください。

注記:

すべてのコントローラーを指定している場合（ADD CONTROLLER *）や、CPU 番号だけを指定している場合（ADD CONTROLLER 2）は、D シリーズの測定アプリケーションで、ServerNet アドレス指定可能コントローラーを測定できます。特定の SAC を測定するには、エンティティ識別子を変更する必要があります。

注記:

CONTROLLER エンティティについての ZMS Style DDL レコードはありません。

トピック

CONTROLLER エンティティのエンティティ指定構文

CONTROLLER エンティティの DDL レコード

CONTROLLER エンティティの DDL フィールドの定義

CONTROLLER エンティティの使用上の注意

CONTROLLER エンティティのエンティティ指定構文

CONTROLLER エンティティを記述するには、次の構文を使用します。

```
CONTROLLER entity-spec [, entity-spec ] ...
```

CONTROLLER

システム上の 1 つまたは複数のコントローラーに関する情報を収集します。

entity-spec

次のように指定します。

```
{ * [ ( type ) ]                                }
{ cpu [ , channel [ , ctrl ] ] [ ( type ) ] }
```

指定する値は次の通りです。

*****

すべての CPU 内のすべてのコントローラーを測定します。

type

コントローラーの Hewlett Packard Enterprise 製品番号（3128 など）です。デフォルトは、指定された cpu、channel、ctrl のセットに一致するすべてのコントローラータイプです。

cpu

測定対象のコントローラーが構成されている CPU の番号です。

channel

測定対象のコントローラーのチャネル番号です。すべてのチャネルを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのチャネルです。

ctrl

測定対象のコントローラーのコントローラー番号（0～31）です。すべてのコントローラーを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのコントローラーです。

CONTROLLER エンティティの DDL レコード（Legacy 形式）

CONTROLLER エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD contrl.  FILE is "contrl" ENTRY-SEQUENCED.
                        .
                        .
                        .
(error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)
                        .
                        .
                        .
* entity identification items:
 02 channel                     type binary 16 unsigned.
 02 ctrl                        type binary 16 unsigned.
 02 ctrl-type                   type binary 16 unsigned.

* counter value items:
 02 requests                    type binary 32 unsigned.
 02 total-io-bytes              type binary 64.
 02 io-qtime                    type binary 64.
 02 io-qlen-max                 type binary 16 unsigned.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

CONTROLLER エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードのみで使用されます。

フィールドはアルファベット順に一覧表示されます。

CHANNEL

測定が実行されたチャネルの数。

CTRL

測定が実行されたコントローラーの数。

CTRL-TYPE

3128 などの製品番号。

CONTROLLER エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードだけで使用されます。

フィールドはアルファベット順に一覧表示されます。

IO-QLEN-MAX

IO-QTIME カウンターで記述されるコントローラーキューの未処理 I/O の最大数。

カウンタータイプ：最大キュー。

IO-QTIME

コントローラーが I/O の実行にビジーになっている時間。I/O の平均数が 1 より小さい場合、この数値はコントローラーのビジーパーセンテージとして解釈できます。

カウンタータイプ：キュー。

REQUESTS

このコントローラーパスに接続されているすべての I/O プロセス（IOP）によるコントローラーへの要求の合計数。

カウンタータイプ：増分。

TOTAL-IO-BYTES

このコントローラーパスに接続されているすべての IOP によってコントローラーとの間で転送された合計バイト数。

カウンタータイプ：累積。

CONTROLLER エンティティの使用上の注意

- インプリメンテーションが異なるため、CONTROLLER カウンターはデバイスレベルのカウンターとは直接比較できません。たとえば、CONTROLLER REQUESTS カウンター値は、そのコントローラーを共有するすべてのデバイスの REQUESTS カウンター値の合計と等しくなりません。
- コントローラーのアクティビティの測定は実用性に問題があり、一般的には推奨されません。通信サブシステムとディスクサブシステムのアーキテクチャーの違いによって、通信デバイスに関する正しいデータは収集できません。さらに、コントローラーエンティティ測定とオンライン設定ユーティリティプログラム（COUP）との間の互換性の問題によって、システムの安定性に深刻な問題（たとえば、システム停止）が発生する可能性があります。そのため、COUP で CONTROLLER エンティティは使用しないでください。
- ストレージコントローラーから返される要求とデータバイトは有効であり、システムのパフォーマンスを評価する上で役に立つ可能性があります。テスト環境で CONTROLLER エンティティを有効にするには、**CONTROLLER**(191 ページ)に記載されている DEFINE を設定してください。

CPU

CPU エンティティは、ローカルシステムの 1 つまたは複数の CPU についての情報を提供します。

トピック

CPU エンティティのエンティティ指定構文

CPU エンティティの DDL レコード

CPU エンティティの DDL フィールドの定義

CPU エンティティの使用上の注意

CPU エンティティのエンティティ指定構文

CPU エンティティを記述するには、次の構文を使用します。

```
CPU entity-spec [, entity-spec ] ...
```

CPU

ローカルシステム上の 1 つまたは複数の CPU に関する情報を収集します。

entity-spec

次のように指定します。

```
{ * | cpu }
```

指定する値は次の通りです。

*****

すべての CPU を示します。

cpu

測定対象の CPU の番号です。複数の CPU を測定するには、CPU 番号のカンマ区切りリストを指定します。

デフォルト値は、すべての CPU です。

CPU エンティティの DDL レコード（ZMS 形式）

CPU エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmscpu-id.

 02 cpu-type                 type binary 16 unsigned.
 02 cpu-subtype               type binary 16 unsigned.
 02 mem-mb                   type binary 16 unsigned.
 02 pcbs                      type binary 16 unsigned.
 02 page-size-bytes           type binary 32 unsigned.
 02 mem-frames                type binary 32 unsigned.
 02 mem-initial-lock          type binary 32 unsigned.
 02 ipus                      type binary 16 unsigned.
 02 flags                     type binary 16 unsigned.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmscpu-ctrs.
 02 cpu-busy-time             type binary 64.
 02 cpu-qtime                 type binary 64.
 02 dispatches                type binary 64.
 02 swaps                     type binary 64.
 02 intr-busy-time            type binary 64.
 02 process-ovhd              type binary 64.
 02 disc-ios                  type binary 64.
 02 cache-hits                type binary 64.
 02 transactions              type binary 64.
 02 response-time             type binary 64.
 02 comp-traps                type binary 64.
 02 native-busy-time          type binary 64.
 02 accel-busy-time           type binary 64.
 02 tns-busy-time             type binary 64.
 02 page-requests             type binary 64.
 02 page-scans                type binary 64.
 02 mm-page-scans             type binary 64.
 02 Starting-free-mem         type binary 32 unsigned.
 02 Ending-free-mem           type binary 32 unsigned.
 02 Starting-UCME             type binary 32 unsigned.
 02 Ending-UCME               type binary 32 unsigned.
 02 Starting-UCL              type binary 32 unsigned.
 02 Ending-UCL                type binary 32 unsigned.
 02 Starting-SCL              type binary 32 unsigned.
 02 Ending-SCL                type binary 32 unsigned.
 02 Starting-Free-CIDs        type binary 32 unsigned.
 02 Ending-Free-CIDs          type binary 32 unsigned.
 02 unsp-pages-qtime          type binary 64.
 02 unsp-pages-start          type binary 32 unsigned.
 02 unsp-pages-end            type binary 32 unsigned.
 02 link-prepush-msgs         type binary 64.
 02 link-readlink-msgs        type binary 64.
 02 link-large-msgs           type binary 64.
 02 readlinkcache-all         type binary 64.
 02 readlinkcache-ctrl        type binary 64.
 02 readlinkcache-none        type binary 64.
 02 replyctrlcache-msgs       type binary 64.
 02 processh-samples          type binary 64.

* New Counters for H01
 02 Starting-timer-cells      type binary 32 unsigned.
 02 Ending-timer-cells        type binary 32 unsigned.

* New Counters for H06
 02 locked-pages-qtime        type binary 64.
 02 locked-pages-start        type binary 64.
 02 locked pages-end          type binary 64.

* New counters for H03
 02 ipu                       occurs 16 times.
    03 ipu-busy-time          type binary 64.
    03 ipu-qtime              type binary 64.
    03 ipu-dispatches         type binary 64.

 02 svnet                     occurs 16 times.
```

```
    03 read-requests          type binary 64.
    03 write-requests         type binary 64.
    03 read-bytes             type binary 64.
    03 write-bytes            type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmscpu. FILE is "zmscpu" ENTRY-SEQUENCED.
 02 hdr                           type zmsheader.
 02 ctr                           type zmscpu-ctrs.
 02 id                            type zmscpu-id.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

CPU エンティティの DDL レコード（Legacy 形式）

CPU エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G シリーズエンティティの CPU DDL レコードは、次の点を除いて D シリーズエンティティの CPU レコードと同じです。

- ダイレクトバルク I/O 転送をサポートするためのカウンターが追加されます。
- 既存の DISK-IOS カウンターにはダイレクトバルク I/O 要求も含まれます。
- SEND-BUSY-TIME フィールドは使用されなくなりました。
- メモリ処理の変更をサポートするためのフィールドが追加され、いくつかの既存のフィールドによって測定されるアクティビティが変更されました。
- フィールドのオーバーフローの可能性を減らすために、長い byte-count フィールドが提供されます。32 ビットの各 byte-count フィールドには、対応する 64 ビットがあります。64 ビットのフィールドの使用については、**G シリーズの CPU エンティティの使用上の注意**(213 ページ)を参照してください。
- ERROR フィールドが、32 ビットの byte-count フィールドのフィールドオーバーフローを通知することがあります。
- G05 では、CPU-QTIME はプロセスがページフォールトのためにレディーリストで費やす時間を測定しません。そのため、MEM-QTIME は常にゼロです。
- G08 では、ServerNet クラスターの計測が追加されます。

```
RECORD cpu. FILE is "cpu" ENTRY-SEQUENCED.
                        .
                        .
                        .
```

(error、time items、および measurement identification items については、**一般的なエンティティヘッダーフィールド**(177 ページ) を参照してください)

```
                        .
                        .
                        .
* entity identification items:
 02 cpu-type                  type binary 16 unsigned.

* redefinition for D25
 02 memory-pages              type binary 16 unsigned.
 02 mem-mb                    type binary 16 unsigned
                               redefines memory-pages.

 02 pcbs                      type binary 16 unsigned.
```

```
 02 lcbs                      type binary 16 unsigned.

* counter value items:
 02 cpu-busy-time             type binary 64.
 02 cpu-qtime                 type binary 64.
 02 cpu-qlen-max              type binary 16 unsigned.
 02 mem-qtime                 type binary 64.
 02 mem-qlen-max              type binary 16 unsigned.
 02 dispatches                type binary 32 unsigned.
 02 swaps                     type binary 32 unsigned.
 02 intr-busy-time            type binary 64.
 02 process-ovhd              type binary 64.
 02 send-busy-time            type binary 64.
 02 disc-ios                  type binary 32 unsigned.
 02 cache-hits                type binary 32 unsigned.
 02 transactions              type binary 32 unsigned.
 02 response-time             type binary 64.
 * redefinition for D25:
 02 memory-pages32            type binary 32 unsigned.
 02 mem-frames                type binary 32 unsigned.
                              redefines memory-pages32.

 02 cpu-subtype               type binary 16 unsigned.

* fields for TNS/R specific counters:
 02 accel-busy-time           type binary 64.
 02 tns-busy-time             type binary 64.
 02 comp-traps                type binary 32 unsigned.

* Native Mode busy time
 02 tnsr-busy-time            type binary 64.

* New memory counters for D25:
 02 page-size-bytes           type binary 16 unsigned.
 02 mem-initial-lock          type binary 32 unsigned.
 02 page-requests             type binary 32 unsigned.
 02 page-scans                type binary 32 unsigned.
 02 Starting-free-mem         type binary 32 unsigned.
 02 Ending-free-mem           type binary 32 unsigned.
 02 Starting-UCME             type binary 32 unsigned.
 02 Ending-UCME               type binary 32 unsigned.

 02 Starting-UDS              type binary 32 unsigned.
 02 Ending-UDS                type binary 32 unsigned.
 02 Starting-UDS-lock         type binary 32 unsigned.
 02 Ending-UDS-lock           type binary 32 unsigned.

 02 Starting-SDS              type binary 32 unsigned.
 02 Ending-SDS                type binary 32 unsigned.
 02 Starting-SDS-lock         type binary 32 unsigned.
 02 Ending-SDS-lock           type binary 32 unsigned.

 02 Starting-UCL              type binary 32 unsigned.
 02 Ending-UCL                type binary 32 unsigned.
 02 Starting-UCL-lock         type binary 32 unsigned.
 02 Ending-UCL-lock           type binary 32 unsigned.
```

```
 02 Starting-SCL              type binary 32 unsigned.
 02 Ending-SCL                type binary 32 unsigned.
 02 Starting-SCL-lock         type binary 32 unsigned.
 02 Ending-SCL-lock           type binary 32 unsigned.
 02 Ending-Free-CIDs          type binary 32 unsigned
                              redefines Ending-SCL-Lock.

* New counters for G04
 02 unsp-pages-qtime          type binary 64.
 02 unsp-pages-qlen-max       type binary 16 unsigned.
 02 unsp-pages-start          type binary 32 unsigned.
 02 unsp-pages-end            type binary 32 unsigned.

* New identifier in G05:
 02 processor-status          type binary 32 unsigned.

* New counters for G05:
 02 disc-ios-f                type binary 64.
 02 cache-hits-f              type binary 64.
 02 svnet                     occurs 16 times.
    03 read-requests          type binary 32 unsigned.
    03 write-requests         type binary 32 unsigned.
    03 read-bytes             type binary 64.
    03 write-bytes            type binary 64.

* New counters for G08:
 02 link-prepush-msgs         type binary 64.
 02 link-readlink-msgs        type binary 64.
 02 link-large-msgs           type binary 64.
 02 readlinkcache-all         type binary 64.
 02 readlinkcache-ctrl        type binary 64.
 02 readlinkcache-none        type binary 64.
 02 replyctrlcache-msgs       type binary 64.
 02 ending-free-cids          type binary 32 unsigned.
                              redefines ending-scl-lock
 02 reserved                  type character 40.

 end
```

CPU エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。

CPU-SUBTYPE

CPU のサブタイプを示すバイナリ値です。

CPU-TYPE

プロセッサータイプ。PROCESSOR_GETNAME_プロシージャーを使用して、プロセッサータイプを判別できます。プロセッサータイプと PROCESSOR_GETNAME_プロシージャーについて詳しくは、*Guardian プロシージャーコールリファレンスマニュアル*を参照してください。

FLAGS

（ZMS 形式のみ）測定が NSMA システムまたは L シリーズ以降の RVU で行われたかどうかを示すために使用される 16 ビットフィールド。

flag.0:13	今後使用するために予約されています。
flags.14	1 = 測定は L シリーズ以降の RVU で行われました**
flags.15	1 = 測定は NSMA システム上で行われました*

* H06.24/J06.13 およびこれ以降の RVU

** L01、H07/J05 およびこれ以降の PVU

IPUS

（ZMS 形式のみ）Measure J01 以降の PVU では、論理 CPU を構成する IPU の数のレポートをサポートしています。

MEM-FRAMES

CPU 構成の使用可能な物理ページ数です。（この値は PEEK ユーティリティの PHYSCL 値です。）

MEM-INITIAL-LOCK

SYSGEN プログラムによってロックされているページフレームの数です。MEM-FRAMES と MEM-INITIAL-LOCK の差は、PEEK では SWAPBL と呼ばれます。この値は、システムのロード中にロックされ、オペレーティングシステムが使用できるメモリページフレームの数です。この値はオペレーティングシステムによって異なります。

MEM-MB

合計メモリサイズ（メガバイト単位）。

この 16 ビットフィールドのみが、最大 64GB のサイズをサポートします。サイズが大きい場合、Measure は 0xFFFF を入力します。その値が MEM-MB で見つかった場合は、MEM-FRAMES（CPU 構成の使用可能な物理ページの数）を PAGE-SIZE-BYTES（測定対象の CPU のメモリページフレームのバイト数）で乗じることで、合計メモリサイズを計算します。MEM-FRAMES は PEEK ユーティリティの PHYSCL 値です。

PAGE-SIZE-BYTES

測定対象の CPU のメモリページフレーム内のバイト数。

PCBS

SYSGEN プログラムによって構成されたプロセス制御ブロック（PCB）の数。

CPU エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

ACCEL-BUSY-TIME

CPU が高速化されたコードの実行でビジー状態だった時間。このカウンターは、TNS/R システムまたは TNS/E システムで行われる測定にのみ適用されます。

H シリーズ以降の CPU エンティティの使用上の注意(212 ページ)を参照してください。

カウンタータイプ：ビジー。

CACHE-HITS

I/O 操作中に必要なブロックがキャッシュ内で見つかった回数。

読み取り操作（または書き込み操作の読み取り部分）の場合、キャッシュヒットによってディスク I/O が節約されます。書き込み操作の場合、キャッシュヒットの結果はファイル I/O のタイプによって異なります。バッファー I/O の場合、キャッシュヒットによってディスク I/O が節約されます。ライトスルー I/O の場合、節約されません。

書き込み操作では、いくつかの場合に読み取りが発生します。たとえば、キーシーケンスファイルでは、データブロックに書き込むときに、1 つまたは複数のインデックスブロックの読み込みが必要になる場合があります。また、部分的に使用されたブロックをディスクに書き込む場合、ディスクプロセスはディスクからブロックを読み取り、新しいデータをブロックにマージし、ブロックをディスクに書き込む必要があります。

DISC エンティティは、追加のキャッシュ情報および I/O 情報を提供します。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、CACHE-HITS-F フィールドは CACHE-HITS の 64 ビットバージョンです。

カウンタータイプ：増分。

COMP-TRAPS

プロセッサーで互換性トラップが発生した回数。互換性トラップは、データのミスアライメントイベント、高速化または非高速化されたコードとの間での予期しない移行、または相対セグメント 2 または 3 の問題です。詳しくは、*EPTRACE Manual* を参照してください。

カウンタータイプ：増分。

CPU-BUSY-TIME

CPU がビジー状態だった時間。

TNS/R システムでは、CPU-BUSY-TIME は ACCEL-BUSY-TIME、TNS-BUSY-TIME、TNSR-BUSY-TIME、および INTR-BUSY-TIME の合計です。

TNS/E システムでは、CPU-BUSY-TIME は ACCEL-BUSY-TIME、TNS-BUSY-TIME、および NATIVE-BUSY-TIME の合計です。CPU-BUSY-TIME の値は常に表示されます。ただし、ACCEL-BUSY-TIME、TNS-BUSY-TIME、および NATIVE-BUSY-TIME の各カウンター値は、測定中に PROCESSH 測定がアクティブだった場合にのみ計算され表示されます。ACCEL-BUSY-TIME、TNS-BUSY-TIME、および NATIVE-BUSY-TIME の各カウンターの動作について詳しくは、これらの各カウンターの説明を参照してください。

Measure J01 以降の PVU の NSMA システムの場合、CPU-BUSY-TIME は論理 CPU を構成する個々の IPU の BUSY-TIME の集計です。REPORT RATE OFF の場合、この値はすべての IPU-BUSY-TIME の合計です。REPORT RATE ON、STYLE ZMS の場合、CPU-BUSY-TIME の最大値は 100%で、この場合 CPU-BUSY-TIME はすべての IPU-BUSY-TIME の AVERAGE に近づきます。REPORT RATE ON、STYLE LEGACY の場合、CPU-BUSY-TIME の最大値は IPU の数に 100%を乗じたものです。

カウンタータイプ：ビジー。

CPU-QTIME

レディーリストでプロセスが費やした時間（マイクロ秒）。Measure G05 以前の PVU では、レディーリストはページフォールトで待機中のプロセスを含んでいます。CPU MEM-QTIME カウンターは、メッセージシステムデータの到着を待機するのに費やした時間を示しています。

Measure G05 以降の PVU では、プロセスはレディーリストタイマー内でアクティブではないため、MEM-QTIME の値はゼロです。

カウンタータイプ：キュー。

DISC-IOS

測定された CPU 内のディスクプロセスによって実行された I/O ディスクの転送回数です。このカウントには、この CPU が別の CPU に対して開始したダイレクトバルク I/O 要求が含まれています。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、DISC-IOS-F フィールドは DISC-IOS の 64 ビットバージョンです。

カウンタータイプ：増分。

DISPATCHES

プロセスがレディーリストから選択され、CPU で実行された回数。PROCESS エンティティの DISPATCHES カウンターは、特定のプロセスがディスパッチ（選択および実行）された回数を示します。

カウンタータイプ：増分。

ENDING-FREE-CIDS

報告された測定間隔の最後にこのプロセッサーの新しいカウンターレコード用に使用可能なカウンター ID 制御ブロックの数。

カウンタータイプ：スナップショット。

ENDING-FREE-MEM

測定間隔の終了時の空きメモリフレームの数。

カウンタータイプ：スナップショット。

ENDING-SCL

測定間隔の終了時に物理メモリに割り当てられたシステムプロセスコードとライブラリセグメントのフレームの数。

カウンタータイプ：スナップショット。

ENDING-TIMER-CELLS

（ZMS 形式のみ）測定間隔の終了時に使用中のインストゥルメンテーションタイマーセルの数。

カウンタータイプ：スナップショット。

ENDING-UCL

測定間隔の終了時に物理メモリに割り当てられたユーザープロセスコードセグメントフレームとライブラリセグメントフレームの数。

カウンタータイプ：スナップショット。

ENDING-UCME

測定間隔の終了時の訂正不能メモリエラー（UCME）フレームの数。H シリーズ以降の RVU では無効です。

カウンタータイプ：スナップショット。

INTR-BUSY-TIME

CPU が割り込みハンドラーの実行に費やした時間。ほとんどの割り込みはメッセージと I/O 操作の処理で発生します。該当するシステムの説明マニュアルで、割り込みの各タイプについて説明しています。

TNS/E システムの場合、これは、システム割り込みプロセス（IP）とシステム補助プロセス（AP）の実行に費やされた合計時間です。

カウンタータイプ：ビジー。

IPU

（ZMS 形式のみ）Measure J01 以降の PVU における NSMA システムの場合、IPU 専用のカウンター。このフィールドは以下の 3 つのサブフィールドに分かれています。

サブフィールド	説明	カウンタータイプ
IPU-BUSY-TIME	この IPU がビジー状態だった時間。	ビジー
IPU-DISPATCHES	この IPU のディスパッチ数。	増分
IPU-QTIME	レディーリストでプロセスが費やした時間（マイクロ秒）。CPU-QTIME と同じですが、IPU 用です。	キュー

LINK-LARGE-MSGS

H/J シリーズ：readlinkcache メッセージサイズを超えた、CPU に転送されたメッセージの数。

L シリーズ：送信操作サイズを超えた、別の CPU に転送されたリクエストの数。要求データのバイトは、IB RDMA 読み取り操作で、サーバー CPU によりプルされます。

カウンタータイプ：増分。

LINK-PREPUSH-MSGS

H/J シリーズ：prepush プロトコルを使用して別の CPU に転送されたメッセージの数。

L シリーズ：送信操作を介して別の CPU に転送された要求の数。

カウンタータイプ：増分。

LINK-READLINK-MSGS

H/J シリーズ：prepush プロトコルを使用せずに、readlinkcache メッセージサイズ以内で別の CPU に転送されたメッセージの数。

L シリーズ：別の CPU から受信した応答の数。

カウンタータイプ：増分。

LOCKED-PAGES-END

（ZMS 形式のみ）Measure H06/J04 以降の PVU で、測定間隔の最後にメインメモリ内にロックされたページフレームの数。値は、CPU で実行しているすべてのプロセスの集約値です。

このカウンターは、NonStop OS が提供する CPU サブシステムバージョン（ZMSCPU.HDR.SUBSYSTEM-VERSION）が 3 以上の場合に有効で、MEASCOM によって ZMS 形式で表示されます。それ以外の場合は、カウンターにはゼロ値が格納され、MEASCOM によって表示されません。

カウンタータイプ：スナップショット。

LOCKED-PAGES-QTIME

（ZMS 形式のみ）今後使用するために予約されています。

LOCKED-PAGES-START

（ZMS 形式のみ）Measure H06/J04 以降の PVU で、測定間隔の最初にメインメモリ内にロックされたページフレームの数。値は、CPU で実行しているすべてのプロセスの集約値です。

このカウンターは、NonStop OS が提供する CPU サブシステムバージョン（ZMSCPU.HDR.SUBSYSTEM-VERSION）が 3 以上の場合に有効で、MEASCOM によって ZMS 形式で表示されます。それ以外の場合は、カウンターにはゼロ値が格納され、MEASCOM によって表示されません。

カウンタータイプ：スナップショット。

MM-PAGE-SCANS

（ZMS 形式のみ）メモリ要求によるものではなかったメモリのクリーンアップを実行するためにメモリマネージャーが開始したメモリページのスキャンの数。メモリマネージャーは、削除可能なページのメモリをバックグラウンドで定期的にスキャンします。このアクティビティでは、パフォーマンスの問題は示されません。**PAGE-SCANS**(204 ページ)も参照してください。

カウンタータイプ：増分。

NATIVE-BUSY-TIME

（ZMS 形式のみ）プロセッサーがネイティブコードの実行中にビジー状態だった時間。

H シリーズ以降の CPU エンティティの使用上の注意(212 ページ)を参照してください。

カウンタータイプ：ビジー。

PAGE-REQUESTS

ページフレームを取得するためにメモリマネージャーが呼び出された回数。（PEEK ユーティリティでは、この値は CLOCK:CALLS と呼ばれます。）

カウンタータイプ：増分。

PAGE-SCANS

置換可能なページを取得するためにメモリマネージャーが調べたページフレームの数。（PEEK ユーティリティは、置換可能なページを見つけるために調べられた平均フレーム数の平均値を示す SCANS/CALLS 値を提供します。）

カウンタータイプ：増分。

PROCESS-OVHD

オペレーティングシステムがプロセスの初期化と終了に費やした時間。PROCESS-OVHD 値は、CPU エンティティの全体的な CPU-BUSY-TIME のサブセットです。PROCESS-OVHD 値は、PROCESS エンティティの CPU-BUSY-TIME カウントには含まれません。このカウンターは、オペレーティングシステムによるプロセスの作成（プロセス制御データ構造のセットアップ、メモリの初期化、修正など）および終了（メモリとデータ構造のクリーンアップ）によって発生したコストを測定し、プロセスがそれ自体の動作の実行に費やした時間は含まれません。

カウンタータイプ：ビジー。

PROCESSH-SAMPLES

（ZMS 形式のみ）このカウンターは、H シリーズ、J シリーズ、およびそれ以降の RVU でのみ有効です。PROCESSH 測定がアクティブの場合は、プロセッサーの PROCESSH サンプルの数の未加工のカウントを報告します。PROCESSH 測定がアクティブでない場合、このカウンターは表示されず、レコードではその値はゼロです。PROCESSH サンプルのサンプリングレートを速くする方法について詳しくは、**H シリーズ以降の PROCESSH エンティティの使用上の注意**(397 ページ)を参照してください。

Measure J01 以降の PVU の NSMA システムでは、PROCESSH-SAMPLES はすべての IPU でのすべてのサンプル数の集計値を報告するため、通常はその CPU の n*サンプリング間隔に等しくなります。

カウンタータイプ：サンプリング。

READLINKCACHE-ALL

H/J シリーズ：制御バイトとデータバイトの両方が readlinkcache バッファーにキャッシュされた受信メッセージの数。

L シリーズ：制御バイトとデータバイトの両方が受信バッファーにキャッシュされた受信要求の数。

カウンタータイプ：増分。

READLINKCACHE-CTRL

H/J シリーズ：制御バイトだけが readlinkcache バッファーにキャッシュされた受信メッセージの数。

L シリーズ：制御バイトだけが受信バッファーにキャッシュされた受信要求の数。

カウンタータイプ：増分。

READLINKCACHE-NONE

H/J シリーズ：制御バイトとデータバイトのどちらも readlinkcache バッファーにキャッシュされなかった受信メッセージの数。

L シリーズ：RDMA 書き込み操作を使用した別の CPU に転送された応答の数。

カウンタータイプ：増分。

REPLYCTRLCACHE-MSGS

H/J シリーズ：制御が割り込みパケットに渡されたメッセージ応答の数。

L シリーズ：送信操作を使用して別の CPU に転送されたメッセージ応答の数。

カウンタータイプ：増分。

RESPONSE-TIME

CPU の端末プロセスとアクティブな測定の端末プロセスが端末応答に費やした時間。トランザクションとは、端末からの読み取りと、その後の端末への書き込を表します。応答時間とは、読み取りが終わってから書き込みが始まるまでの間隔を表します。

関連するカウンター	機能
TRANSACTIONS	トランザクションの数を測定します
TERMINAL RESPONSE-TIME および TRANSACTIONS	特定の端末の応答時間とトランザクションを測定します

カウンタータイプ：応答時間。

STARTING-FREE-CIDS

（ZMS 形式のみ）報告された測定間隔の最初に、このプロセッサーの新しいカウンターレコードで使用可能なカウンター ID 制御ブロックの数。

カウンタータイプ：スナップショット。

STARTING-FREE-MEM

測定間隔の開始時の空きメモリフレームの数。

カウンタータイプ：スナップショット。

STARTING-SCL

測定間隔の開始時に物理メモリに割り当てられたシステムプロセスコードとライブラリセグメントのフレームの数。このカウンターには、システムプロセスコードセグメントの TNS コード、高速化されたコード、およびネイティブコードに割り当てられたフレームの数と、システムコード（SC）およびシステムライブラリ（SL）として指定されたショートアドレス空間に割り当てられたフレームの数が含まれます。また、割り込みプロシージャーおよびミリコードに割り当てられているフレームも含まれます（TNS/R システムの場合）。

カウンタータイプ：スナップショット。

STARTING-TIMER-CELLS

（ZMS 形式のみ）測定間隔の開始時に使用中の計測タイマーのセル数。

カウンタータイプ：スナップショット。

STARTING-UCL

測定間隔の開始時に物理メモリに割り当てられたユーザープロセスコードとライブラリセグメントフレームの数。このカウンターには、ユーザープロセスの TNS コード、高速化コード、およびネイティブコードのセグメントに割り当てられたフレームの数と、ユーザーコード（UC）およびユーザーライブラリ（UL）として指定された短いアドレス空間に割り当てられたフレームの数が含まれます。

カウンタータイプ：スナップショット。

STARTING-UCME

測定間隔の開始時の訂正不能メモリエラー（UCME）フレームの数。H シリーズ以降の RVU では無効です。

カウンタータイプ：スナップショット。

SVNET

測定対象のプロセッサーのシステム相互接続アクティビティ。このフィールドは以下のサブフィールドに分かれています。

サブフィールド	説明	カウンタータイプ
READ-REQUESTS	測定対象のプロセッサーで開始され、指定されたプロセッサー（CPU *n*）から情報を取得したシステム相互接続転送の数。測定対象のプロセッサー内での転送の場合、このカウントは標準 I/O 読み取り要求の数です。他のプロセッサーでの転送の場合、このカウントには DBIO 読み取り要求の数が含まれます。	増分
WRITE-REQUESTS	測定対象のプロセッサーで開始され、指定されたプロセッサー（CPU *n*）に情報を送信したシステム相互接続転送の数。測定対象のプロセッサー内での転送の場合、このカウントは、標準 I/O 書き込み要求の数と、測定対象のプロセッサーから開始されたメッセージシステム送信要求の数です。他のプロセッサーへの転送の場合、このカウントには DBIO 書き込み要求の数が含まれます。	増分
READ-BYTES	指定されたプロセッサー（CPU *n*）に対する読み取り要求によって転送されたバイト数。	累積
WRITE-BYTES	指定されたプロセッサー（CPU *n*）に対する書き込み要求によって転送されたバイト数。	累積

Measure のレポートでは、システム相互接続アクティビティの値は、測定対象のプロセッサーとの間でデータを転送したプロセッサーごとに提供されます。

すべてのプロセッサーのレポートを生成して TOTALS オプションを使用すると、レポートにはプロセッサーごとのシステム相互接続データ転送アクティビティの合計が示されます。特定の各プロセッサーのフィールドは、そのプロセッサーによるシステム相互接続データ要求の合計と解釈できます。I/O 要求はこれらの全体で 1 回カウントされますが、メッセージシステムプロセッサ間通信（IPC）の要求は転送に関係するプロセッサーごとに 1 回カウントされるため、2 回カウントされます。

SWAPS

メモリマネージャーによって実行されたスワップ操作（メモリへのスワップインとメモリからのスワップアウトの両方）の数。メモリマネージャーは、一度に 1 ページまたは 2 ページ分をスワップします。CPU SWAPS カウンターは、スワップされるページ数に関係なく、スワップごとに 1 回加算されます （CPU SWAPS カウンターは DISC SWAPS カウンターとは異なります）。

スワップされるのは更新されたページのみです。コードページは、メモリにはスワップされますが、更新できないため、スワップアウトされることはありません。データページは、メモリに対するスワップインとスワップアウトの両方が実行されます。ただし、初期データのないデータページは、使用されるまではスワップインされません。メモリ要件とスワップインおよびスワップアウトされるページのタイプに応じて、ページフォールト（PROCESS エンティティによってカウントされる）でスワップが行われない場合や、1 回のスワップが行われる場合、2 回のスワップが行われる場合があります。

状況	スワップ
コードが削除された、またはメモリ内で使用可能な空きページがある。初期データページのスワップインなし	0
コードが削除された、または使用可能な空きページがある。コードまたはデータページで初期データがスワップインされた	1
使用済みのデータページがスワップアウトされ、使用済みのデータページがスワップインされた	2

カウンタータイプ：増分。

TNS-BUSY-TIME

CPU が TNS コードの実行でビジー状態だった時間。このカウンターは、TNS/R または TNS/E システムで行われる測定にのみ適用されます。

H シリーズ以降の CPU エンティティの使用上の注意(212 ページ)を参照してください。

カウンタータイプ：ビジー。

TRANSACTIONS

現在アクティブな測定の測定対象 CPU で端末プロセスによって実行されるトランザクションの数。トランザクションとは、端末からの読み取りと、その後の端末への書き込みを表します。応答時間とは、読み取りが終わってから書き込みが始まるまでの間隔を表します。

関連するカウンター	機能
RESPONSE-TIME	応答時間を測定します
TERMINAL RESPONSE-TIME および TRANSACTIONS	特定の端末の応答時間とトランザクションを測定します

カウンタータイプ：増分。

UNSP-PAGES-END

測定間隔の終了時に UNSP-PAGES-QTIME キューにあるページの数。

カウンタータイプ：スナップショット。

UNSP-PAGES-QTIME

スポンサリング解除されたページ（どのプロセスにも所有されていない共有プロセス）がメインメモリで費やした時間。スポンサリング解除されたページの説明については、**G シリーズの CPU エンティティの使用上の注意**(213 ページ)を参照してください。

カウンタータイプ：キュー。

UNSP-PAGES-START

測定間隔の開始時に UNSP-PAGES-QTIME キューにあるページの数。

カウンタータイプ：スナップショット。

CPU エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードのみで使用されます。

フィールドはアルファベット順に一覧表示されます。

LCBS

（Legacy 形式のみ）現在は使用されていません。0 を返します。

MEMORY-PAGES

（Legacy 形式のみ）再定義されました。MEM-MB を参照してください。

MEMORY-PAGES32

（Legacy 形式のみ）再定義されました。MEM-FRAMES を参照してください。

PROCESSOR-STATUS

（Legacy 形式のみ）システム上のプロセッサーの数（すべての NonStop G シリーズシステムの場合は 16）と、測定開始時の各プロセッサーの状態（アップ状態は 1、ダウン状態は 0）です。Guardian の PROCESSORSTATUS プロシージャーコールから取得されます。H シリーズ以降の RVU を実行しているシステムには使用されません。

CPU エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードだけで使用されます。

フィールドはアルファベット順に一覧表示されます。

CACHE-HITS-F

（Legacy 形式のみ）G シリーズの RVU では、CACHE-HITS と同じですが、より大きい値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：増分。

CPU-QLEN-MAX

（Legacy 形式のみ）CPU-QTIME カウンターで記述されるキューの項目の最大数。システムがロードされたら、すべてのシステムプロセスはレディーリストに置かれます。このカウンターはシステムロード時にのみ初期化されるため、一般的に 1 回限りの非常に長い CPU キューを反映します。

カウンタータイプ：最大キュー。

DISC-IOS-F

（Legacy 形式のみ）G シリーズの RVU では、DISC-IOS と同じですが、より大きい値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：増分。

ENDING-SCL-LOCK

（Legacy 形式のみ）現在は使用されていません。

ENDING-SDS

（Legacy 形式のみ）測定間隔の終了時に物理メモリに割り当てられたシステムプロセスのデータセグメントのフレーム数。Measure G05 以降の PVU に対する SPR では、ENDING-SDS フレームは ENDING-SCL カウンターに含まれます。

カウンタータイプ：スナップショット。

ENDING-SDS-LOCK

（Legacy 形式のみ）現在は使用されていません。

ENDING-UCL-LOCK

（Legacy 形式のみ）現在は使用されていません。

ENDING-UDS

（Legacy 形式のみ）測定間隔の終了時に物理メモリに割り当てられたユーザープロセスのデータセグメントのフレーム数。T9050G05 以降の PVU に対する SPR では、ENDING-UDS フレームは ENDING-UCL カウンターに含まれます。

カウンタータイプ：スナップショット。

ENDING-UDS-LOCK

（Legacy 形式のみ）現在は使用されていません。

MEM-QLEN-MAX

（Legacy 形式のみ）MEM-QTIME カウンターで記述されるキューの項目の最大数。

Measure G05 以降の PVU では、ゼロです。以前の PVU では、プロセッサーがロードされてからキューに入れられたページフォールトの最大数です。

カウンタータイプ：最大キュー。

MEM-QTIME

（Legacy 形式のみ）測定対象のプロセッサーにあるプロセスがページフォールトで待機するのに費やした CPU 時間です。（ページフォールトによってシステムがレディリストからプロセスを削除することはありません。）

Measure G05 以降の PVU では、メモリ処理が変更されたため、このカウンターによって測定されるアクティビティは多少変更されました。以前の G シリーズの RVU（およびすべての D シリーズの RVU）では、ページフォールト要求はメモリマネージャー（$VIRTUAL）によって処理されました。同時要求は処理のためにキューに入れる必要がありました。

Measure G05 以降の PVU では、ページフォールトの処理はシステムライブラリの機能です。複数の要求が同時に処理されるため、キューイングは行われません。その結果、MEM-QTIME は常にゼロです。

関連するカウンター	機能
CPU-QTIME	すべてのプロセスがレディリストで費やした時間（ページフォールトで待機するのに費やした時間を含む）を測定します。
PROCESS MEM-QTIME	プロセスがページフォールトで待機するのに費やした時間を測定します。
PROCESS PAGE-FAULTS	プロセスによって生成されるページフォールトの数をカウントします。

カウンタータイプ：キュー。

SEND-BUSY-TIME

（Legacy 形式のみ）D シリーズの RVU では、CPU が他の CPU へのデータの送信に費やした時間（つまり、SEND 命令の実行に費やした時間）。このカウンターは、INTR-BUSY-TIME カウンターと部分的に重複します。

G シリーズの RVU では、使用されなくなりました。

カウンタータイプ：ビジー。

STARTING-SCL-LOCK

（Legacy 形式のみ）現在は使用されていません。

STARTING-SDS

（Legacy 形式のみ）測定間隔の開始時に物理メモリに割り当てられたシステムプロセスのデータセグメントのフレーム数。このカウンターには、システムプロセススタックセグメント、拡張データセグメント、およびプロセスファイルセグメント（PFS）に割り当てられたフレーム数、および CPU システムグローバルデータセグメント（SG）とその他の拡張データセグメントに割り当てられたフレームの数、またはオペレーティングシステムが使用するために割り当てられたフレームの数が含まれます。システムプロセスは、Measure の PROCESS エンティティのシステムプロセスカテゴリに分類されるプロセスです。Measure G05 以降の PVU に対する SPR では、STARTING-SDS フレームは STARTING-SCL カウンターに含まれます。

カウンタータイプ：スナップショット。

STARTING-SDS-LOCK

（Legacy 形式のみ）現在は使用されていません。

STARTING-UCL-LOCK

（Legacy 形式のみ）現在は使用されていません。

STARTING-UDS

（Legacy 形式のみ）測定間隔の開始時に物理メモリに割り当てられたユーザープロセスデータセグメントのフレーム数。このカウンターには、ユーザープロセススタックセグメント、拡張データセグメント、およびプロセスファイルセグメント（PFS）に割り当てられたフレームの数が含まれます。T9050G05 以降の PVU に対する SPR では、STARTING-UDS フレームは STARTING-UCL カウンターに含まれます。

カウンタータイプ：スナップショット。

STARTING-UDS-LOCK

（Legacy 形式のみ）現在は使用されていません。

TNSR-BUSY-TIME

（Legacy 形式のみ）CPU が TNS/R ネイティブコードの実行でビジー状態だった時間。このカウンターは、TNS/R システムで行われる測定にのみ適用され、他のカウンターから取得されます。

```
CPU-BUSY-TIME - (INTR-BUSY-TIME + TNS-BUSY-TIME + ACCEL-BUSY-TIME) = TNSR-BUSY-TIME
```

カウンタータイプ：ビジー。

UNSP-PAGES-QLEN-MAX

（Legacy 形式のみ）UNSP-PAGES-QTIME カウンターで記述されるキューの最大項目数。

H シリーズ以降の RVU では、このカウンターの値は 1 です。

カウンタータイプ：最大キュー。

すべての CPU エンティティの使用上の注意

- CPU カウンターは、CPU がロードされるときに割り当てられて初期化されます。CPU カウンターは、広範な期間にわたるシステムのパフォーマンスの概要を提供します。

  CPU カウンターは、他のエンティティを使用してより詳細を調べることができる潜在的な問題を示すことができます。特に、PROCESS の CPU-BUSY-TIME カウンターは、特定のプロセスを実行するの

に費やした時間を測定します （PROCESS の CPU-NUM フィールドによってプロセスと CPU を照合することができます）。
- システム操作（システムがどのように I/O を実行し、プロセスを実行し、ページングを実行するかなど）について詳しくは、該当するシステムの説明マニュアルを参照してください。
- CPU のすべての PROCESS エンティティの IPU-BUSY-TIME と CPU-BUSY-TIME の相違について詳しくは、**All PROCESS エンティティの使用上の注意**(384 ページ)を参照してください。

J シリーズの CPU エンティティの使用上の注意

J06.13 以降の RVU では、有効な IPU/CPU の数を動的に増やすことができます。このような変更が行われているときに CPU 測定がアクティブである場合、Measure は自動的に測定を停止して、新しい IPU 数で測定を再開します。この動作によって、RATE が ON の場合に行われるスケーリングの計算がすべてのレコードで正確であることが保証されます。ただし、RATE が ON のときに TOTALS レコードが要求された場合、値の信頼性がなくなります。

RATE が ON のときの CPU TOTALS レコードが重要となる場合は、有効な IPU/CPU に変更を加える前に、すべての CPU 測定を停止することをお勧めします。変更が完了した後で CPU 測定を再開できます。このようにすることで、データファイル内のすべてのレコードが、有効な IPU の同じ番号を基準にすることになります。

アクティブな読み取りは特殊なケースです。というのも、外部レコードのデルタ時間と RATE が ON のときの MEASCOM の値を計算するために、測定が開始されたときのタイムスタンプではなく割り当てタイムスタンプ（基本的には CPU が最後に再ロードされた時間）を使用するためです。IPU カウントが変化している間にもほとんどのカウンターでこれらの計算を正確に行うために、元の割り当てタイムスタンプが内部カウンターレコードに保存され、これらのタイムスタンプを使用して外部レコードが作成されます。

ただし、一部のカウンター値は古い IPU カウントでスケーリングする必要があり、別の一部は新しい IPU カウントでスケーリングする必要があるため、いくつかのカウンターフィールドに対して行われた IPU スケーリングは正しくない可能性があります。また、CPU が再ロードされた後に IPU/CPU でいくつかの変更が行われた可能性もあります。その結果、Measure API によって返された外部レコードの値が正しいとしても、IPU/CPU で変更が行われた場合、RATE が ON のときの LISTACTIVE コマンドの以下の CPU カウンターフィールドは、MEASCOM では正しく表示されません。

- ACCEL-BUSY-TIME
- CPU-BUSY-TIME
- INTR-BUSY-TIME
- NATIVE-BUSY-TIME
- PROCESS-OVHD
- TNS-BUSY-TIME

FOR を指定して LISTACTIVE コマンドを発行した場合は、その期間の開始から終了までの間に IPU/CPU で変更が行われていない限り、正しい値が表示されます。

アクティブな読み取りに関するスケーリングの問題を修正する唯一の方法は、CPU を再ロードすることです。

H シリーズ以降の CPU エンティティの使用上の注意

- H シリーズ、J シリーズ、およびそれ以降の RVU では、すべての byte-count フィールドが 64 ビットに対応しています。ZMS 形式のカウンター名には接尾辞-F は表示されなくなりましたが、Legacy 形式のデータを要求するアプリケーションでは-F で終わる名前を持つカウンターを引き続き使用できます。
- Measure H06/J04 以降の PVU では、ZMS 形式の CPU および ZMS 形式の PROCESS で、LOCKED-PAGES-END と LOCKED-PAGES-START のカウンターがサポートされます。これらのカウンターを使用するときには、次の使用上の留意事項に注意してください。

- CPU のすべての PROCESS エンティティでロックされていると報告されたページフレームの合計は、対応する CPU エンティティでロックされていると報告されたページフレームと同じである必要はありません。
- ページフレームは、メモリ内に置かれ、存続するようにマークされると、ロックされます。このようなマークがすべて削除されたり、ページフレームの割り当てが解除されると、ロックが解除されます。
- ロックされたページフレームは、最初はそれをロックしたプロセスに関連付けられます（スポンサリングされます）。ページが複数のプロセス間で共有され、最初のプロセスが終了した場合、ページのスポンサリングが解除され、その後、ページを共有している残りのプロセスのいずれかによってスポンサリングされます。ページフレームが任意のプロセスの一意のアドレス空間ではなくグローバルアドレス空間にマップされている場合、そのページはメモリマネージャープロセス PIN 1 によってスポンサリングされるようになります。このプロセスは、システムによってシステム用に割り当てられたページもスポンサリングします。
- スポンサリングされておらずロックされたページフレームは、メモリマネージャープロセスによって次のルーチンに達するまでスポンサリング解除されたままになります（どのプロセスにも関連付けられません）。

- CPU エンティティの初期レコードは、測定の開始時ではなく、CPU のロード時に作成されます。取得がアクティブであるときには、初期レコードと現在のレコードの間のデルタ時間を考慮して、現在のカウンター値が表示されます。ACCEL-BUSY-TIME、NATIVE-BUSY-TIME、および TNS-BUSY-TIME のカウンターは、デルタ時間の間に CPU で PROCESSH 測定がアクティブであった場合にのみ計算されて返され、表示されます。そうでない場合、カウンターは表示されず、返されたレコードの値はゼロになります。この期間全体で PROCESSH 測定がアクティブでなかった場合（CPU エンティティである可能性が低い）、このカウンター値は正しい値を示しません。

  これらのカウンターは、該当するコード領域で観察された PROCESSH のサンプル数に基づいて計算されるため、より正確な数値を得るには、PROCESSH サンプルの頻度を増やします。PROCESSH-SAMPLES カウンターは、現在のサンプリング頻度を報告します。

  このカウンターは Measure H04/J02 以降の PVU で有効ですが、測定間隔の開始から終了まで PROCESSH サンプル数が変わらない場合、PROCESSH 測定がアクティブであっても、MEASCOM で、ACCEL-BUSY-TIME、NATIVE-BUSY-TIME、および TNS-BUSY-TIME のフィールドは表示されません。LISTALL コマンドでは、これらのフィールドは表示される間隔と表示されない間隔があります。

  MEASCOM コマンドの LISTACTIVE FOR *time* は、該当の期間全体にわたって PROCESSH 測定が実行されていた場合にのみ、ACCEL-BUSY-TIME、NATIVE-BUSY-TIME、および TNS-BUSY-TIME のカウンターの正しい値がもたらされます。

G シリーズの CPU エンティティの使用上の注意

- 64 ビットのバイトカウントフィールド（-F で終わるフィールド）は、従来の 32 ビットのバイトカウントフィールドと同じデータを収集します。たとえば、64 ビットフィールドの DISC-IOS-F は、32 ビットフィールドの DISC-IOS と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

  32 ビットフィールドは現在アクティブであり、引き続き値を返します フィールドのオーバーフローがない場合、32 ビットフィールドと 64 ビットのフィールドは同じ値を返します。32 ビットフィールドがオーバーフローした場合、対応する 64 ビットフィールドは正しい値を返し、32 ビットフィールドは-1 の値を返します。測定対象のエンティティの ERROR フィールドも、オーバーフロー状態を示す-1 を返します。

  64 ビットフィールドを使用するには、アプリケーションを変換してください。32 ビットのフィールドは、将来のリリースでは使用できなくなる可能性があります。

  MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットフィールドの名前を使用してください。たとえば、LIST CPU BY DISC-IOS-F ではなく LIST CPU BY DISC-IOS のコマンドを発行してください。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。
- Measure G05 以降の PVU では、スポンサリングされたメモリページとスポンサリング解除されたメモリページが使用可能です。ページを使用するプロセス（通常はセグメントを定義するプロセスですが、

必ずしもそうとは限りません）によって、ページはメモリ内でスポンサリングされます。共有ページのスポンサーであるプロセスが終了すると、ページは一時的にスポンサリング解除されます。スポンサリング解除されたページを参照した次の共有プロセスが、そのページのスポンサーになります。以下の CPU カウンターは、スポンサリング解除されたページに関する情報を提供します。

UNSP-PAGES-QTIME	UNSP-PAGES-START
UNSP-PAGES-QLEN-MAX	UNSP-PAGES-END

- さまざまなタイプのメッセージシステム転送プロトコルおよび概念（事前プッシュ、事後プル、リンカー、およびリスナーなど）の説明については、*NonStop S シリーズサーバーの説明マニュアル*を参照してください。
- Measure G08 以降の PVU では、ServerNet クラスターをサポートするためのいくつかの新しいカウンターが Measure に追加されています。
- プロセッサー内の空き CID の数は、使用可能な CID の最大数とすべてのアクティブな測定による現在の使用状況によって決まります。24 個の CID は内部リストヘッダーに使用されます。CID の数は DEFINE によって設定できます。この設定パラメーターは、START MEASSUBSYS コマンドが発行されたときに読み取られます。

D シリーズの CPU エンティティの使用上の注意

Measure D30 以降の PVU では、以下のカウンターがサポート対象外になりました。

STARTING-UDS-LOCK	ENDING-UDS-LOCK	STARTING-SDS-LOCK	ENDING-SDS-LOCK
STARTING-UCL-LOCK	ENDING-UCL-LOCK	STARTING-SCL-LOCK	ENDING-SCL-LOCK

DEVICE

DEVICE エンティティは、テープドライブやプリンターなどのデバイスに関する情報を提供します。G シリーズ以降の RVU では、DEVICE エンティティはテープドライブにのみ適用されます。

注記:

一部のデバイスは、特定のエンティティタイプで識別されます。ディスクの場合は DISC、通信回線の場合は LINE および NETLINE、サブデバイスおよび非同期端末の場合は TERMINAL です。

トピック

DEVICE エンティティのエンティティ指定構文

DEVICE エンティティの DDL レコード

DEVICE エンティティの DDL フィールドの定義

DEVICE エンティティの使用上の注意

DEVICE 測定の例

DEVICE エンティティのエンティティ指定構文

DEVICE エンティティを記述するには、次の構文を使用します。

```
DEVICE entity-spec [, entity-spec ] ...
```

DEVICE

ローカルシステム上の 1 つまたは複数のデバイスに関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                                                          }
{ $device [ ( cpu [, fabric [, group [, module [, slot
  [, scsi-id ] ] ] ] ] ) ]                                   }
```

または、CLIM 接続デバイスのみを指定するには、次のように指定します。

```
{ $device [ ( cpu [, fabric [, sac [, lun ] ] ] ) ]      }
```

また、D シリーズのデータのみを一覧表示する場合は、次のように指定します。

```
{ *                                                          }
{ $device [ ( cpu [ , chan [ , ctrl [ , unit ] ] ] ) ] }
```

指定する値は次の通りです。

*****

すべての CPU のすべてのデバイスを示します。

$*device*

測定対象のデバイスの名前です。すべてのデバイスを指定するには、アスタリスク（*）を使用します。

cpu

測定対象のデバイスが構成されている CPU の番号です。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

fabric

このデバイスで使用されるシステム相互接続ファブリックです。

Measure サブシステムは､指定されたファブリックに関連するすべてのデバイスレコードを測定します。値は、X ファブリックでは 0、Y ファブリックでは 1 です。デフォルトは、両方のファブリックのレコードを測定することです。

ServerNet/DA で接続されたデバイスには、X または Y ファブリックはありません。これらのディスクを指定するには、アスタリスク（*）を使用します。この場合、エンティティ指定の残りの部分が 1 つのデバイスを記述している場合に限り、LISTACTIVE コマンドでアスタリスク（*）を使用することができます。

FCSA または CLIM によって接続されているデバイスはデュアルファブリックのアダプターです。つまり、X と Y の両方のファブリックに接続されています。これらのディスクの場合、LIST コマンドでは「*」または「X」のみを指定し、LISTACTIVE コマンドでは「X」のみを指定する必要があります。

group

（G シリーズ、H シリーズ、および J シリーズで、CLIM デバイスには使用しない）測定するデバイスのグループ番号です。（グループは物理エンクロージャーに相当します）。すべてのグループを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのグループです。

module

（G シリーズ、H シリーズ、および J シリーズで、CLIM デバイスには使用しない）測定対象のデバイスのモジュール番号です。すべてのモジュールを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのモジュールです。

slot

（G シリーズ、H シリーズ、および J シリーズで、CLIM デバイスには使用しない）測定対象のデバイスのスロット番号です。すべてのスロットを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのスロットです。

scsi-id

（G シリーズ、H シリーズ、および J シリーズで、CLIM デバイスには使用しない）測定対象のデバイスの SCSI ポート識別子です。すべての SCSI ポートを指定するには、アスタリスク（*）を使用します。デフォルトは、すべての SCSI ポートです。この値は ServerNet/DA および FCSA デバイスについて使用されますが、CLIM デバイスには使用されません。

sac

CLIM の名前です。DEVICE エンティティについては 1 つの CLIM 名のみがサポートされています。

lun

CLIM 内のディスクに割り当てられている論理ユニット番号です。NonStop オペレーティングシステムでは、*lun* について 0～65535 の範囲をサポートしています。Measure では、ワイルドカード（*）用に 65535（-1）を予約した後の 0～65534 の範囲をサポートしています。*lun* は CLIM デバイスについてのみ指定できます。デフォルトは、すべての lun です。

chan

（D シリーズ）測定対象のデバイスのチャネル番号です。デフォルトは、すべてのチャネルです。

ctrl

（D シリーズ）測定対象のデバイスのコントローラー番号（0～31）です。デフォルトは、すべてのコントローラーです。

unit

（D シリーズ）測定対象のデバイスのユニット番号です。デフォルトは、すべてのユニットです。

DEVICE エンティティの DDL レコード（ZMS 形式）

DEVICE エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsdev-id.
 02 pin                          type binary 16 unsigned.
 02 device-type                  type binary 16 unsigned.
 02 device-subtype               type binary 16 unsigned.
 02 fabric                       type binary 16 unsigned.
 02 servernet                    type binary 16 unsigned redefines fabric.
 02 device-name                  type character 8.
 02 logical-device               type binary 32 unsigned.
 02 GMS.
    03 group                     type binary 32 unsigned.
    03 module                    type binary 32 unsigned.
    03 slot                      type binary 32 unsigned.
```

```
 02 SCSI-id                      type binary 64.
 02 plpt                         redefines SCSI-id.
    03 plpt-flags                type binary 8 unsigned.
    03 path                      type binary 8 unsigned.
    03 lun                       type binary 16 unsigned.
    03 partition                 type binary 16 unsigned.
    03 target-id                 type binary 16 unsigned.
 02 config-name                  type character 64.
 02 adapter-name                 type character 64.
 02 SAC-name                     type character 64.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsdev-ctrs.
 02 requests                     type binary 64.
 02 reads                        type binary 64.
 02 writes                       type binary 64.
 02 input-bytes                  type binary 64.
 02 output-bytes                 type binary 64.
 02 read-qbusy-time              type binary 64.
 02 read-qtime                   type binary 64.
 02 write-qbusy-time             type binary 64.
 02 write-qtime                  type binary 64.
 02 device-qbusy-time            type binary 64.
 02 DBIO-reads                   type binary 64.
 02 DBIO-writes                  type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsdev. FILE is "zmsdev" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 id                           type zmsdev-id.
 02 ctr                          type zmsdev-ctrs.
end
```

DEVICE エンティティの DDL レコード（Legacy 形式）

DEVICE エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています（要求は G シリーズ概要レポートでのみ表示されます）。

G シリーズ以降の DEVICE エンティティの DDL レコードは、次の点を除いて D シリーズ DEVICE エンティティのレコードと同じです。

- CHANNEL フィールドが SERVERNET に再定義されます。
- 別のエンティティ識別フィールドが使用されます。CTRL フィールドおよび UNIT フィールドは使用されなくなります。
- ServerNet/DA または FCSA によって接続されるデバイスを識別するための SCSI ID 識別子が追加されます。
- キューに基づく ServerNet 環境でビジー時間を測定するためのカウンターが追加されます。READ-BUSY-TIME カウンターおよび WRITE-BUSY-TIME カウンターが使用されなくなります。
- ダイレクトバルク I/O 操作を測定するためのカウンターが追加されます。
- READS や WRITES などの既存の I/O カウンターはダイレクトバルク I/O 操作も含みます。
- フィールドのオーバーフローの可能性を減らすために、長い byte-count フィールドが提供されます。G シリーズの RVU では、32 ビットの各 byte-count フィールドに対応する 64 ビットがあります。H

シリーズ以降の RVU では、すべての byte-count フィールドが 64 ビットに対応しています。64 ビットのフィールドの使用については、**G シリーズの DEVICE エンティティの使用上の注意**(225 ページ)を参照してください。

- ERROR フィールドが、32 ビットの byte-count フィールドのフィールドオーバーフローを通知することがあります。

```
RECORD device. FILE is "device" ENTRY-SEQUENCED.
                        .
                        .
                        .
(error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)
                        .
                        .
                        .
* entity identification items:
 02 pin                           type binary 16 unsigned.
 02 device-name                   type character 8.
 02 logical-device                type binary 16 unsigned.
 02 ctrl                          type binary 16 unsigned.
 02 unit                          type binary 16 unsigned.
 02 device-type                   type binary 16 unsigned.
 02 device-subtype                type binary 16 unsigned.

* counter value items:
 02 requests                      type binary 32 unsigned.
 02 read-busy-time                type binary 64.
 02 write-busy-time               type binary 64.
 02 reads                         type binary 32 unsigned.
 02 writes                        type binary 32 unsigned.
 02 input-bytes                   type binary 32 unsigned.
 02 output-bytes                  type binary 32 unsigned.

* new entity identification item for D10:
 02 channel                       type binary 16 unsigned.
 02 servernet                     type binary 16 unsigned
                                  redefines channel.

* F40 new entity identification items:
 02 config-name                   type character 64.
 02 adapter-name                  type character 64.
 02 SAC-name                      type character 64.
 02 GMS.
    03 group                      type binary 32 unsigned.
    03 module                     type binary 32 unsigned.
    03 slot                       type binary 32 unsigned.

* F40 new counter value items:
 02 read-qbusy-time               type binary 64.
 02 read-qtime                    type binary 64.
 02 read-qlen-max                 type binary 16 unsigned.
 02 write-qbusy-time              type binary 64.
 02 write-qtime                   type binary 64.
 02 write-qlen-max                type binary 16 unsigned.
 02 device-qbusy-time             type binary 64.
 02 input-bytes-f                 type binary 64.
 02 output-bytes-f                type binary 64.
```

```
* New identifier in G05:
 02 scsi-id                      type binary 64.
* New counters for G05:
 02 dbio-reads                   type binary 32 unsigned.
 02 dbio-writes                  type binary 32 unsigned.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

DEVICE エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

ADAPTER-NAME

SAC が存在するアダプターに関連付けられた論理名です。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

CONFIG-NAME

このパスでアクセスする物理デバイスの論理名です。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

DEVICE-NAME

測定対象のデバイスの名前。

DEVICE-SUBTYPE

DEVICE-TYPE の追加の識別子。デバイスサブタイプのリストについては、*Guardian Operations Reference Summary* を参照してください。

DEVICE-TYPE

測定したデバイスのタイプ（テープ、プリンターなど）。デバイスタイプの完全なリストについては、*Guardian Operations Reference Summary* を参照してください。G シリーズ以降の RVU では、DEVICE エンティティはテープ（DEVICE-TYPE 4）およびオープン SCSI（DEVICE-TYPE 8）のみを測定します。

FABRIC

このデバイスで使用されるシステム相互接続ファブリックです。

Measure サブシステムは、指定されたファブリックに関連するすべてのデバイスレコードを測定します。値は、X ファブリックでは 0、Y ファブリックでは 1 です。デフォルトは、両方のファブリックのレコードを測定することです。

ServerNet/DA で接続されたデバイスには、X または Y ファブリックはありません。これらのディスクを指定するには、アスタリスク（*）を使用します。この場合、エンティティ指定の残りの部分が 1 つのデバイスを記述している場合に限り、LISTACTIVE コマンドでアスタリスク（*）を使用することができます。

FCSA または CLIM によって接続されているデバイスはデュアルファブリックのアダプターです。つまり、X と Y の両方のファブリックに接続されています。これらのディスクの場合、LIST コマンドでは「*」または「X」のみを指定し、LISTACTIVE コマンドでは「X」のみを指定する必要があります。

GMS

物理的位置のアドレス（グループ、モジュール、スロット）。GMS フィールドは、次の 3 つのサブフィールドに分かれています。

GROUP	グループ番号です。
MODULE	モジュール番号です。
SLOT	スロット番号です。

GROUP、MODULE、および SLOT のキーワードは、LIST コマンドおよび LISTALL コマンドの IF 句と BY 句で使用できます。

注記:

CLIM デバイスではなく、G、H、および J シリーズで使用します。

LOGICAL-DEVICE

測定デバイスの論理デバイス番号。

PIN

I/O プロセスのプロセス識別番号。

PLPT

（Measure H03 以降の PVU の ZMS 形式のみ）PLPT 構造体は、SCSI-ID フィールドに代わるものです。CLIM デバイスを識別するにはより多くの属性が必要だったため、SCSI-ID フィールド（0〜999 の範囲）は 16 ビットに減らされ、PLPT.TARGET-ID フィールドになりました。この構造体は、記述子とデータレコードで使用されます。PLPT 構造体は、次の 5 つのサブフィールドで構成されます。

PLPT-FLAGS	MEAS_CLIM_REL	このフィールドを、SCSI-ID フィールドではなく PLPT 構造体として識別します。*scsi-id* はそこに-1F（ワイルドカード）が指定されている可能性があるため（記述子の場合）、このフラグに意味があるのは PLPT-FLAGS フィールドが 1s だけではない場合のみです。下位互換性を維持するため、デバイス構成が CLIM デバイス専用であった場合を除き、このビットは MEASLISTCONFIG を使用して返される記述子に対しては設定されません。
	MEAS_PATH_SEL	（DEVICE エンティティと組み合わせて使用されることはありません。）データレコードではなく、記述子でのみ使用されます。
	MEAS_CLIM_DEVICE	この記述子またはデータレコードが CLIM デバイス用であることを指定します。
	MEAS_MULTIPLE_CLIMS	（DEVICE エンティティと組み合わせて使用されることはありません。）データレコードではなく、記述子でのみ使用されます。
PATH	（DEVICE エンティティと組み合わせて使用されることはありません）。	

表は続く

LUN	論理ユニット番号。この番号は、ストレージ CLIM 内のディスクドライブに割り当てられている一意の番号です。この番号は、特定の CLIM 内でのみ一意です。CLIM より前のデバイスでは、この番号は 0 です。CLIM デバイスでは、有効な範囲は 0～65534 です。このフィールドは符号なし数値として扱われます。*lun* が記述子にある場合、ワイルドカードはすべて 1s です
PARTITION	（DEVICE エンティティと組み合わせて使用されることはありません）。
TARGET-ID	SCSI インターフェイスマネージャーによって返される SCSI デバイス ID。Measure H03 の PVU より前では、この値は 64 ビットの SCSI-ID フィールドに格納されていました。この値だけが 16 ビットを必要とするため、Measure H03 の PVU 以降では、*lun* と *path* に空いた上位の 48 ビットを使用するために、この値はこの小さいフィールドに格納されます。*target-id* の有効な値は 0～999 です。*target-id* が記述子にある場合、ワイルドカードはすべての 1s です。

SAC-NAME

（G、H、および J シリーズの RVU のみ）このパスに対してこのデバイスをサポートする ServerNet アドレス可能コントローラー（SAC）の論理名。論理名はシステム構成データベースによって保持されます。デバイスが CLIM デバイスの場合、SAC-NAME は CLIM の名前です。この名前は null で埋められ null で終了します。

SCSI-ID

このデバイスの SCSI ポート識別子。

SERVERNET

詳しくは、**FABRIC**(219 ページ)を参照してください。

DEVICE エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。

DBIO-READS

要求側のアプリケーションプロセッサーへのダイレクトバルク I/O のために有効化された測定対象デバイスへの読み取り操作の数。このカウンターは READS のサブセットです。

カウンタータイプ：増分。

DBIO-WRITES

要求側のアプリケーションプロセッサーからのダイレクトバルク I/O のために有効化された測定対象デバイスへの書き込み操作の数。このカウンターは WRITES のサブセットです。

カウンタータイプ：増分。

DEVICE-QBUSY-TIME

任意の数またはタイプの要求が、このプロセッサーのこのデバイスのキューに入れられた状態で経過した時間。

書き込みキューと読み取りキュー内の並列動作のため、WRITE-QBUSY-TIME と READ-QBUSY-TIME を加算しても DEVICE-QBUSY-TIME に等しくなりません。

カウンタータイプ：キュービジー。

INPUT-BYTES

デバイスから読み取られたバイト数。I/O プロセスは I/O 操作の前にこのカウンターを変更するため、書き込みが失敗するとこのカウンターは正確ではなくなることがあります。プログラムによる読み取り操作だけでなく、内部操作（I/O 操作の再試行など）もこのカウンターを変更します。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、INPUT-BYTES-F フィールドが INPUT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

OUTPUT-BYTES

デバイスに書き込まれたバイト数。I/O プロセスは I/O 操作の前にこのカウンターを変更するため、書き込みが失敗するとこのカウンターは正確ではなくなることがあります。プログラムによる書き込み操作だけでなく、ボリュームラベルの書き込みなどの内部操作もこのカウンターを変更します。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、OUTPUT-BYTES-F フィールドが OUTPUT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

READ-QBUSY-TIME

任意の数の読み取り要求が、このプロセッサーのこのデバイスのキューに入れられた状態で経過した時間。

カウンタータイプ：キュービジー。

READ-QTIME

このデバイスのキューに入れられた読み取り要求が費やした合計時間。

カウンタータイプ：キュー。

READS

I/O プロセスによって実行された（デバイスからメモリへの）読み取り操作の数。プログラムによる読み取り操作だけでなく、内部操作（I/O 操作の再試行など）もこのカウンターを変更します。データブロックの前にインデックスブロックを読み取る必要があるため、キーシーケンスファイルへの読み取り操作では複数の読み取りが行われる可能性があります。

カウンタータイプ：増分。

REQUESTS

I/O プロセスが受信した要求の数。I/O プロセスが読み取るまで要求がキューに入れられていた時間を確認するには、I/O プロセスを測定し、PROCESS RECV-QTIME カウンターを調べます。

カウンタータイプ：増分。

WRITE-QBUSY-TIME

任意の数の書き込み要求が、このプロセッサーのこのデバイスのキューに入れられた状態で経過した時間。

カウンタータイプ：キュービジー。

WRITE-QTIME

このデバイスのキューに入れられた書き込み要求が費やした合計時間。

カウンタータイプ：キュー。

WRITES

デバイスプロセスによって実行された（メモリからデバイスへの）書き込み操作の数。プログラムによる書き込み操作だけでなく、ボリュームラベルの書き込みなどの内部操作もこのカウンターを変更します。ブロック分割やインデックスブロックへの更新のためにキーシーケンスファイルへの書き込み操作が発生すると、結果として複数の書き込みが加算される可能性があります。

カウンタータイプ：増分。

DEVICE エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードのみで使用されます。

フィールドはアルファベット順に一覧表示されます。

CHANNEL

（Legacy 形式のみ）D シリーズの RVU で、デバイスのチャネル番号。G シリーズの RVU では、再定義されます。SERVERNET を参照してください。

CTRL

（Legacy 形式のみ）D シリーズの RVU では、測定されたデバイスのコントローラー番号。G シリーズの RVU では現在使用されていません。ゼロを返します。

UNIT

（Legacy 形式のみ）D シリーズの RVU では、測定対象のデバイスのユニット番号を表します。G シリーズ以降の RVU では現在使用されていません。ゼロを返します。

DEVICE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードだけで使用されます。

フィールドはアルファベット順に一覧表示されます。

INPUT-BYTES-F

（G シリーズ、Legacy 形式のみ）INPUT-BYTES と同じですが、より大きな値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

OUTPUT-BYTES-F

（G シリーズ、Legacy 形式のみ）OUTPUT-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

READ-BUSY-TIME

（Legacy 形式のみ）D シリーズの RVU では、デバイスからの読み取りに費やした時間。G シリーズ以降の RVU では現在使用されていません。ゼロを返します。

カウンタータイプ：ビジー。

READ-QLEN-MAX

（Legacy 形式のみ）カウンターレコードが割り当てられた後の任意の時点で、このデバイスのキューに入れられた読み取り要求の最大数。

カウンタータイプ：最大キュー。

WRITE-BUSY-TIME

（Legacy 形式のみ）D シリーズの RVU では、デバイスへの書き込みに費やした時間を表します。G シリーズの RVU では現在使用されていません。ゼロを返します。

カウンタータイプ：ビジー。

WRITE-QLEN-MAX

（Legacy 形式のみ）カウンターレコードが割り当てられた後の任意の時点で、このデバイスのキューに入れられた書き込み要求の最大数。

カウンタータイプ：最大キュー。

すべての DEVICE エンティティの使用上の注意

- DEVICE カウンターは、すべてのデバイスアクティビティを追跡します。デバイスアクティビティには、ユーザープロセスとシステムプロセスの両方からの I/O が含まれます。また、空き領域テーブルへの書き込みやファイルラベルとボリュームラベルへの書き込みなどの内部アクティビティも含まれます。2 つ以上のデバイスのワークロードを比較してバランスを調整するには、DEVICE カウンターを使用します。
- ZMSDEV レコードの SUBSYSTEM-VERSION は、デバイス I/O プロセス、たとえばテーププロセス（OTPPROCP）によって提供されます。
- DEVICE レポートには、DEVICE カウンターから計算されるフィールドが含まれます。これらのフィールドは、DEVICE DDL レコードでは定義されていません。

  DEVICE-AST

  （ZMS 形式のみ）各要求が待機または DP2 による処理に費やした平均時間。DEVICE-QBUSY-TIME は READS と WRITES の合計で除算されます。

  カウンタータイプ：サービス時間。

H シリーズ以降の DEVICE エンティティの使用上の注意

- Measure H03/J01 以降の PVU の CLIM 接続デバイスに対して DEVICE エンティティを指定する場合、*lun* および *path* を含め、*scsi id* を使用しないように更新されています。FCSA デバイスと同様に、このエンティティ指定は ADD、DELETE、LIST、LISTACTIVE、および LISTALL コマンドで使用できます。

注記:

path フィールドは ZMS 形式の DDL DEVICE レコード構文の PLPT 構造体にも引き続き表示されますが、H06.23/J06.12 以降の RVU では、*path* フィールドは DEVICE エンティティでは使用できず、CLIM 接続の DISC エンティティに対してのみサポートされています。

- H シリーズ以降の RVU では、すべての byte-count フィールドが 64 ビットに対応しています。-F で終わるフィールド名は、ZMS 形式のレコードでは使用されなくなりましたが、Legacy 形式でデータを要求するアプリケーションでは引き続き使用できます。

G シリーズの DEVICE エンティティの使用上の注意

G シリーズでの Measure の DEVICE エンティティは、テープに対してのみ使用されます。TIL および THL は廃止され、すべてのプリンターのサポートは LAN 接続のデバイスから行われます。テープは SCSI デバイスとしてサポートされ、モジュールドライバーコードをディスク操作と共有します。

64 ビットのバイトカウントフィールド（-F で終わるフィールド）は、従来の 32 ビットのバイトカウントフィールドと同じデータを収集します。たとえば、64 ビットフィールドの INPUT-BYTES-F は、32 ビットフィールドの INPUT-BYTES と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

32 ビットフィールドは現在アクティブであり、引き続き値を返します フィールドのオーバーフローがない場合、32 ビットフィールドと 64 ビットのフィールドは同じ値を返します。32 ビットフィールドがオーバーフローした場合、対応する 64 ビットフィールドは正しい値を返し、32 ビットフィールドは-1 の値を返します。測定対象のエンティティの ERROR フィールドも、オーバーフロー状態を示す-1 を返します。

64 ビットフィールドを使用するには、アプリケーションを変換してください。32 ビットのフィールドは、将来のリリースでは使用できなくなる可能性があります。

MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットのフィールドの名前を使用してください。たとえば、LIST DEVICE BY INPUT-BYTES-F ではなく LIST DEVICE BY INPUT-BYTES のコマンドを発行してください。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。

D シリーズの DEVICE エンティティの使用上の注意

アプリケーションがすべてのデバイス（ADD DEVICE *）を指定している場合や*$device* または*$device (cpu)*のみを指定している場合は、D シリーズの測定アプリケーションを使用して NonStop S シリーズサーバーのデバイスを測定することができます。アプリケーションが *controller*、*channel*、または *unit* を指定している場合に NonStop S シリーズのデバイスを測定するには、エンティティ識別子を変更する必要があります。

例

コマンド	説明
ADD DEVICE $TAPE	$TAPE という名前のデバイスを測定します
ADD DEVICE *	すべてのデバイスを測定します
ADD DEVICE $* (2)	CPU 2 のすべてのデバイスを測定します

表は続く

コマンド	説明
ADD DEVICE $* (*, *, 110, 3, 1, 2)	GMS が 110,3,1 で、*scsi-id* が 2 のコントローラーに接続されているすべてのデバイスを測定します（ServerNet/DA または FCSA ストレージデバイスのみ）
ADD DEVICE $* (*,*,CLIM3)	CLIM3 という名前の SAC に接続されているすべてのデバイスを測定します

DISC

DISC エンティティは、ローカルシステムの 1 つまたは複数のディスクについての情報を提供します。キャッシュのサイズ変更があったとき、対応する DISC 測定が再開され、新しいプライマリディスクレコードおよびバックアップディスクレコードのセットが生成されます。パスの切り替えがあった場合、つまりプライマリディスクプロセスによって構成済みのプライマリ CPU およびバックアップ CPU が切り替わるとき、これらのプロセスについてのプライマリディスクレコードの新しいセットが作成されます。古いプライマリディスクレコードは更新されません。DISC 容量情報はプライマリ DP2 プロセスの DISC レコードからのみ入手できます。この情報は、バックアップ DP2 プロセスレコードでは通常は入手できません。ただし、そのプロセスが測定中にプライマリプロセスである場合を除きます。

注記:

MEASCOM コマンドでは、キーワード DISK は DISC と交換可能です。

トピック

DISC エンティティのエンティティ指定構文

DISC エンティティの DDL レコード

DISC エンティティの DDL フィールドの定義

DISC エンティティの使用上の注意

DISC 測定の例

DISC エンティティのエンティティ指定構文

DISC エンティティを記述するには、次の構文を使用します。

```
DISC entity-spec [, entity-spec ] ...
```

DISC

ローカルシステム上の 1 つまたは複数のディスクに関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                                                                    }
{ $disc [ ( cpu [ , fabric [ , group [ , module  [ , slot
  [, scsi-id ] ] ] ] ] ) ]                                             }
```

または、複数のパスのみを持つディスクの場合は、次のように指定します。

```
{ $disk[path]                                                    }
```

または、CLIM 接続ディスクのみを指定するには、次のように指定します。

```
{ $disc [ ( cpu [, fabric [, sac [, lun [, partition ] ] ] ] ) ]  }
```

また、D シリーズのデータのみを一覧表示する場合は、次のように指定します。

```
{ $disc [ ( cpu [, channel [, controller [, unit ] ] ] ) ]       }
```

指定する値は次の通りです。

*****

すべての CPU のすべてのディスクを測定します。いずれかの指定子がアスタリスク（*）に従う場合、$*を使用する必要があります。

$*disc*

測定対象のディスクの名前です。すべてのディスクを指定するには、アスタリスク（*）を使用します。

cpu

ディスクが構成されている CPU の番号です。デフォルトは、すべての CPU です。すべての CPU を指定するには、アスタリスク（*）を使用します。

fabric

このデバイスで使用されるシステム相互接続ファブリックです。

Measure サブシステムは､指定されたファブリックに関連するすべてのディスクレコードを測定します。値は、X ファブリックでは 0、Y ファブリックでは 1 です。デフォルトは、両方のファブリックのレコードを測定することです。

ServerNet/DA で接続されたディスクには、X または Y ファブリックはありません。これらのディスクを指定するには、アスタリスク（*）を使用します。この場合、エンティティ指定の残りの部分が 1 つのディスクを記述している場合に限り、LISTACTIVE コマンドでアスタリスク（*）を使用することができます。

FCSA または CLIM によって接続されているディスクはデュアルファブリックのアダプターです。つまり、X と Y の両方のファブリックに接続されています。これらのディスクの場合、LIST コマンドでは「*」または「X」のみを指定し、LISTACTIVE コマンドでは「X」のみを指定する必要があります。

group

（G シリーズ、H シリーズ、および J シリーズで、CLIM デバイスには使用しない）測定対象のディスクのグループ番号です。（グループ番号は物理エンクロージャーに相当します）。すべてのグループを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのグループです。

module

（G シリーズ、H シリーズ、および J シリーズで、CLIM デバイスには使用しない）測定対象のディスクのモジュール番号です。すべてのモジュールを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのモジュールです。

slot

（G シリーズ、H シリーズ、および J シリーズで、CLIM デバイスには使用しない）測定対象のディスクのスロット番号です。すべてのスロットを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのスロットです。

scsi-id

（G シリーズ、H シリーズ、および J シリーズで、CLIM デバイスには使用しない）測定対象のディスクの SCSI ポート識別子です。すべての SCSI ポートを指定するには、アスタリスク（*）を使用します。デフォルトは、すべての SCSI ポートです。この値は ServerNet/DA および FCSA ディスクに使用され、CLIM ディスクには使用されません。

path

ストレージ CLIM ディスクだけでなく FCSA ストレージディスクにも適用されます。*パス*の有効な値は「-P」、「-B」、「-M」または「-MB」です（それぞれ、プライマリ、バックアップ、ミラー、ミラーバックアップを指定します）。path が指定された場合、svnet、sac、lun、および partition に指定された値は無視されます。

sac

ディスクが接続されている SAC の名前です。デバイスが CLIM デバイスの場合、*sac* は CLIM の名前です。2 つまでの CLIM 名をサポートします。2 つの CLIM 名が指定されている場合、カッコで囲んで 2 つの名前をカンマで区切る必要があります。デフォルトは、すべての CLIM です。

lun

CLIM 内のディスクに割り当てられている論理ユニット番号です。NonStop オペレーティングシステムでは、*lun* について 0～65535 の範囲をサポートしています。Measure では、ワイルドカード（*）用に 65535（-1）を予約した後の 0～65534 の範囲をサポートしています。デフォルトは、すべての lun です。

partition

パーティション分割されたディスク内のパーティションに割り当てられている番号です。デフォルトは、すべてのパーティションです。H06.23/J06.12 以降の RVU の CLIM 接続ディスクに対してのみ有効です。

channel

（D シリーズ）測定対象のディスクのチャネル番号です。デフォルトは、すべてのチャネルです。

ctrl

（D シリーズ）測定対象のディスクのコントローラー番号です。デフォルトは、すべてのコントローラーです。

unit

（D シリーズ）測定対象のディスクのユニット番号です。デフォルトは、すべてのユニットです。

DISC エンティティの DDL レコード（ZMS 形式）

DISC エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsdisc-id.
 02 pin                          type binary 16 unsigned.
 02 device-type                  type binary 16 unsigned.
 02 device-subtype               type binary 16 unsigned.
 02 fabric                       type binary 16 unsigned.
 02 servernet                    type binary 16 unsigned redefines fabric.
 02 device-name                  type character 8.
 02 logical-device               type binary 32 unsigned.
 02 GMS.
    03 group                     type binary 32 unsigned.
    03 module                    type binary 32 unsigned.
```

```
    03 slot                     type binary 32 unsigned.
 02 SCSI-id                     type binary 64.
 02 plpt                        redefines SCSI-id.
    03 plpt-flags               type binary 8 unsigned.
    03 path                     type binary 8 unsigned.
    03 lun                      type binary 16 unsigned.
    03 partition                type binary 16 unsigned.
    03 target-id                type binary 16 insigned.
 02 config-name                 type character 64.
 02 adapter-name                type character 64.
 02 SAC-name                    type character 64.
 02 capacity                    type binary 64.
 02 storage-pool                type character 8.
 02 disc-process-type           type binary 16 unsigned.
 02 reserved-1                  type character 6.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsdisc-ctrs.
 02 request-qtime               type binary 64.
 02 requests                    type binary 64.
 02 reads                       type binary 64.
 02 writes                      type binary 64.
 02 input-bytes                 type binary 64.
 02 output-bytes                type binary 64.
 02 swaps                       type binary 64.
 02 control-points              type binary 64.
 02 control-point-writes        type binary 64.
 02 free-space-ios              type binary 64.
 02 requests-blocked            type binary 64.
 02 starting-free-space         type binary 64.
 02 ending-free-space           type binary 64.
 02 starting-free-blocks        type binary 32 unsigned.
 02 ending-free-blocks          type binary 32 unsigned.
 02 volsem-qtime                type binary 64.
 02 read-qbusy-time             type binary 64.
 02 read-qtime                  type binary 64.
 02 write-qbusy-time            type binary 64.
 02 write-qtime                 type binary 64.
 02 device-qbusy-time           type binary 64.
 02 DBIO-reads                  type binary 64.
 02 DBIO-writes                 type binary 64.
 02 defreqs                     type binary 64.
 02 defreq-qtime                type binary 64.
 02 deferred-qtime              type binary 64.
 02 c                           occurs 8 times.
    03 blks                     type binary 32 unsigned.
    03 block-size               type binary 32 unsigned.
    03 blocks-inuse-start       type binary 32 unsigned.
    03 blocks-inuse-end         type binary 32 unsigned.
    03 block-splits             type binary 64.
    03 hits                     type binary 64.
    03 misses                   type binary 64.
    03 faults                   type binary 64.
    03 audit-buf-forces         type binary 64.
    03 blks-dirty-qtime         type binary 64.
```

```
    03 write-cleans              type binary 64.
    03 write-dirtys              type binary 64.
    03 write-misses              type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsdisc. FILE is "zmsdisc" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 ctr                          type zmsdisc-ctrs.
 02 id                           type zmsdisc-id.
end
```

DISC エンティティの DDL レコード（Legacy 形式）

DISC エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G シリーズ DISC エンティティの DDL レコードは、次の点を除いて D シリーズ DISC エンティティのレコードと同じです。

- CHANNEL フィールドが SERVERNET に再定義されます。
- 別のエンティティ識別フィールドが使用されます。CTRL フィールドおよび UNIT フィールドは使用されなくなります。
- ServerNet/DA または FCSA によって接続されるディスクを識別するための SCSI ID 識別子が追加されます。
- キューに基づく ServerNet 環境でビジー時間を測定するためのカウンターが追加されます。READ-BUSY-TIME カウンターおよび WRITE-BUSY-TIME カウンターが使用されなくなります。
- ダイレクトバルク I/O 操作を測定するためのカウンターが追加されます。
- READS や WRITES などの既存の I/O カウンターはダイレクトバルク I/O 操作を含みます。
- フィールドのオーバーフローの可能性を減らすために、長い byte-count フィールドが提供されます。G シリーズの RVU では、32 ビットの各 byte-count フィールドに対応する 64 ビットがあります。同様に、16 ビットの各 byte-count フィールドには、対応する 32 ビットがあります。H シリーズ以降の RVU では、すべての byte-count フィールドが 64 ビットに対応しています。64 ビットまたは 32 ビットのフィールドの使用については、**G シリーズの DISC エンティティの使用上の注意**(247 ページ)

  を参照してください。
- ERROR フィールドが、32 ビットの byte-count フィールドのフィールドオーバーフローを通知することがあります。

```
RECORD disc. FILE is "disc" ENTRY-SEQUENCED.
                        .
                        .
                        .
```

(error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)

```
                        .
                        .
                        .
* entity identification items:
 02 pin                          type binary 16 unsigned.
 02 device-name                  type character 8.
 02 logical-device               type binary 16 unsigned.
 02 ctrl                         type binary 16 unsigned.
 02 unit                         type binary 16 unsigned.
 02 device-type                  type binary 16 unsigned.
 02 device-subtype               type binary 16 unsigned.
```

```
 02 disc-process-type             type binary 16 unsigned.

* counter value items:
 02 request-qtime                  type binary 64.
 02 request-qlen-max               type binary 16 unsigned.
 02 requests                       type binary 32 unsigned.
 02 read-busy-time                 type binary 64.
 02 write-busy-time                type binary 64.
 02 seek-busy-time                 type binary 64.
 02 reads                           type binary 32 unsigned.
 02 writes                          type binary 32 unsigned.
 02 seeks                           type binary 32 unsigned.
 02 input-bytes                    type binary 32 unsigned.
 02 output-bytes                   type binary 32 unsigned.
 02 swaps                           type binary 32 unsigned.
 02 cblks-inuse-qtime              type binary 64.
 02 cblks-inuse-max                type binary 16 unsigned.
 02 ablks-inuse-qtime              type binary 64.
 02 ablks-inuse-max                type binary 16 unsigned.
 02 c                               occurs 4 times.
    03 hits                         type binary 32 unsigned.
    03 misses                       type binary 32 unsigned.
    03 faults                       type binary 32 unsigned.
    03 audit-buf-forces             type binary 32 unsigned.
    03 blks                         type binary 16 unsigned.
    03 blks-dirty-qtime             type binary 64.
    03 blks-dirty-max               type binary 16 unsigned.
 02 control-points                  type binary 32 unsigned.
 02 control-point-writes           type binary 32 unsigned.
 02 free-space-ios                 type binary 32 unsigned.
 02 requests-blocked                type binary 32 unsigned.

* new entity identification item for D10:
 02 channel                         type binary 16 unsigned.
 02 servernet                       type binary 16 unsigned.
 02 capacity                        type binary 64.

* new counter value items for D10:
 02 starting-free-space            type binary 64.
 02 ending-free-space              type binary 64.
 02 starting-free-blocks           type binary 32 unsigned.
 02 ending-free-blocks             type binary 32 unsigned.
 02 cw                              occurs 4 times.
    03 write-dirtys                 type binary 32 unsigned.
    03 write-cleans                 type binary 32 unsigned.
    03 write-misses                 type binary 32 unsigned.
    03 block-splits                 type binary 32 unsigned.
 02 volsem-qtime                    type binary 64.
 02 volsem-qlen-max                type binary 16 unsigned.

* SMS changes:
 02 storage-pool                    type character 8.

* new entity identification item for F40:
 02 config-name                     type character 64.
 02 adapter-name                    type character 64.
 02 SAC-name                        type character 64.
```

```
 02 GMS.
    03 group                       type binary 32 unsigned.
    03 module                      type binary 32 unsigned.
    03 slot                        type binary 32 unsigned.

* new counter value items for F40:
 02 read-qbusy-time               type binary 64.
 02 read-qtime                    type binary 64.
 02 read-qlen-max                 type binary 16 unsigned.
 02 write-qbusy-time              type binary 64.
 02 write-qtime                   type binary 64.
 02 write-qlen-max                type binary 16 unsigned.
 02 device-qbusy-time             type binary 64.
 02 input-bytes-f                 type binary 64.
 02 output-bytes-f                type binary 64.

* New counters for G05
 02 scsi-id                       type binary 64.

* New counters in G05:
 02 DBIO-reads                    type binary 32 unsigned.
 02 DBIO-writes                   type binary 32 unsigned.

* New counters in G07:
 02 cn                            occurs 4 times.
    03 blks                       type binary 32 unsigned.
    03 blks-inuse-start           type binary 32 unsigned.
    03 blks-inuse-end             type binary 32 unsigned.

* New counters in G08:
 02 defreqs                       type binary 32 unsigned.
 02 defreq-qtime                  type binary 64.
 02 defreq-qlen-max               type binary 16 unsigned.
 02 deferred-qtime                type binary 64.
 02 deferred-qlen-max             type binary 16 unsigned.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

DISC エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

ADAPTER-NAME

SAC が存在するアダプターに関連付けられた論理名です。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

CAPACITY

ディスクボリューム容量（バイト単位）。項目は IF 句および BY 句の場合のみ使用できます。

CONFIG-NAME

この物理ディスクに関連付けられた論理名です。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

DEVICE-NAME

デバイスの名前。

DEVICE-SUBTYPE

DEVICE-TYPE の追加の識別子。ディスクのサブタイプ値のリストについては、*System Generation Manual* を参照してください。

DEVICE-TYPE

値は 3 で、ディスクボリュームを示します。

DISC-PROCESS-TYPE

ディスクプロセスタイプ DP2 です。

FABRIC

このデバイスで使用されるシステム相互接続ファブリックです。

Measure サブシステムは、指定されたファブリックに関連するすべてのディスクレコードを測定します。値は、X ファブリックでは 0、Y ファブリックでは 1 です。デフォルトは、両方のファブリックのレコードを測定することです。

ServerNet/DA で接続されたディスクには、X または Y ファブリックはありません。これらのディスクを指定するには、アスタリスク（*）を使用します。この場合、エンティティ指定の残りの部分が 1 つのディスクを記述している場合に限り、LISTACTIVE コマンドでアスタリスク（*）を使用することができます。

FCSA または CLIM によって接続されているディスクはデュアルファブリックのアダプターです。つまり、X と Y の両方のファブリックに接続されています。これらのディスクの場合、LIST コマンドでは「*」または「X」のみを指定し、LISTACTIVE コマンドでは「X」のみを指定する必要があります。

GMS

物理的位置のアドレス（グループ、モジュール、スロット）。GMS フィールドは、次の 3 つのサブフィールドに分かれています。

GROUP	グループ番号です。
MODULE	モジュール番号です。
SLOT	スロット番号です。

GROUP、MODULE、および SLOT のキーワードは、LIST コマンドおよび LISTALL コマンドの IF 句と BY 句で使用できます。

注記:

CLIM デバイスではなく、G、H、および J シリーズで使用します。

LOGICAL-DEVICE

デバイスの論理デバイス番号。

PIN

ディスクプロセスのプロセス識別番号。

PLPT

（ZMS 形式のみ）Measure H03 以降の PVU では、PLPT 構造体は SCSI-ID フィールドに代わるものです。CLIM デバイスを識別するにはより多くの属性が必要だったため、SCSI-ID フィールド（0～999 の範囲）は 16 ビットに減らされ、PLPT.TARGET-ID フィールドになりました。この構造体は、記述子とデータレコードで使用されます。PLPT 構造体は、次の 5 つのサブフィールドで構成されます。

PLPT-FLAGS	MEAS_CLIM_REL	このフィールドを、SCSI-ID フィールドではなく PLPT 構造体として識別します。*scsi-id* はそこに-1F（ワイルドカード）が指定されている可能性があるため（記述子の場合）、このフラグに意味があるのは PLPT-FLAGS フィールドが 1s だけではない場合のみです。下位互換性を維持するため、デバイス構成が CLIM デバイス専用であった場合を除き、このビットは MEASLISTCONFIG を使用して返される記述子に対しては設定されません。
	MEAS_PATH_SEL	選択が *path* とデバイス名のみによるもので、*lun*、*partition*、または *scsi-id* などの他の属性によるものではないことを指定します。このフラグが意味を持つのは、PLPT-FLAGS フィールドが 1s だけではなく、データレコードではなく記述子でのみ使用される場合のみです。
	MEAS_CLIM_DEVICE	この記述子またはデータレコードが CLIM デバイス用であることを指定します。
	MEAS_MULTIPLE_CLIMS	この記述子の CONFIG-NAME フィールドに複数の CLIM 名が含まれることを指定します。H06.23/J06.12 以降の RVU でのみ有効です。データレコードではなく、記述子でのみ使用されます。
PATH	NonStop オペレーティングシステムとディスクドライブ間の接続。NonStop オペレーティングシステムには、ディスクドライブに 2 つのパス（バックアップとプライマリ）がある可能性があります。CLIM デバイスの場合は、フォールトトレランス要件のために、これには各 CLIM を通る 1 つのパスを持つ 2 つのストレージ CLIM が必要です。ミラー化された NonStop オペレーティングシステムのボリュームは、実際には 2 つのディスクドライブで構成されるため、このようなボリュームには 4 つのパスが関連付けられている可能性があります。有効な範囲は、プライマリ、バックアップ、ミラー、ミラーバックアップのそれぞれで 0～3 です。*path* が記述子にある場合、ワイルドカードはすべての 1s です。	
LUN	論理ユニット番号。この番号は、ストレージ CLIM 内のディスクドライブに割り当てられている一意の番号です。この番号は、特定の CLIM 内でのみ一意です。CLIM より前のデバイスでは、この番号は 0 です。CLIM デバイスでは、有効な範囲は 0～65534 です。このフィールドは符号なし数値として扱われます。*lun* が記述子にある場合、ワイルドカードはすべての 1s です。	

表は続く

PARTITION	H06.23/J06.12 以降の RVU では、パーティション分割されたディスク内のパーティションに割り当てられているパーティション番号。CLIM 接続のディスクに対してのみ有効です。
TARGET-ID	SCSI インターフェイスマネージャーによって返される SCSI デバイス ID。Measure H03 の PVU より前では、この値は 64 ビットの SCSI-ID フィールドに格納されていました。この値だけが 16 ビットを必要とするため、Measure H03 の PVU 以降では、*lun* と *path* に空いた上位の 48 ビットを使用するために、この値はこの小さいフィールドに格納されます。*target-id* の有効な値は 0〜999 です。*target-id* が記述子にある場合、ワイルドカードはすべての 1s です。

SAC-NAME

物理ボリュームへのこのパスで使用される SAC に関連付けられている論理名。論理名はシステム構成データベースによって保持されます。デバイスが CLIM デバイスの場合、SAC-NAME は CLIM の名前です。デフォルトは、すべての CLIM です。この名前は null で埋められ null で終了します。

SCSI-ID

このディスクの SCSI ポート識別子。

SERVERNET

詳しくは、**FABRIC**(233 ページ)を参照してください。

STORAGE-POOL

測定対象のディスクが割り当てられている Storage Management Foundation（SMF）ストレージプールの名前（ストレージプールプロセスのプロセス名）。

DISC エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

C

（ZMS 形式のみ）ディスクキャッシュメトリック。C のフィールドは ZMS 形式の DDL と Legacy 形式の DDL レコードの両方に含まれますが、サブフィールドの内容およびサブフィールドのシーケンスはレコードのスタイルに基づいて異なります。Legacy 形式の DDL レコードのサブフィールドの内容と C フィールドのシーケンスについては、**DISC エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義**(242 ページ)を参照してください。ZMS 形式の DDL レコードで、このフィールドは以下のサブフィールドに分かれています。

サブフィールド	説明	カウンタータイプ
BLKS	割り当てられているキャッシュブロックの数。TOTALS ONLY または TOTALS INCLUDE が使用されている場合、この値は割り当てられているキャッシュブロックの最大数です。BLKS はカウンターではなく、内部のディスクプロセスデータ構造から取得した値です。	なし
BLOCK-SIZE	（ZMS 形式のみ）キャッシュのブロックサイズ（バイト単位）。	なし
BLOCKS-INUSE-START	（ZMS 形式のみ）測定の最初に使用されていたキャッシュブロックの数。この値は構造化された DISC レコードの一部ですが、Legacy 形式の DISC レポートには表示されません。TOTALS ONLY または TOTALS INCLUDE が使用されている場合、この値は測定の最初に使用されるキャッシュブロックの合計数です。	スナップショット
BLOCKS-INUSE-END	（ZMS 形式のみ）測定の最後に、または測定間隔コピー時間に使用されていたキャッシュブロックの数。TOTALS ONLY または TOTALS INCLUDE が使用されている場合、この値は測定の最後にまたは測定間隔コピー時間に使用されるキャッシュブロックの合計数です。	スナップショット
BLOCK-SPLITS	（ZMS 形式のみ）ボリュームの（ブロックサイズあたりの）ブロック分割の数。	増分
HITS	読み取り操作またはバッファー書き込み操作で必要なブロックがキャッシュ内に見つかり、ディスク I/O が節約された回数。このカウンターには、読み取りおよび書き込みの両方の操作のキャッシュヒットが含まれます。	増分
MISSES	読み取り操作時に必要なブロックがキャッシュ内に見つからず、ディスク I/O を発生させた回数。キャッシュを経由しない読み取り操作はこのカウンターに含まれません。	増分
FAULTS	必要なブロックがキャッシュ内に存在することがキャッシュテーブルで示されたが、メモリマネージャーがそれを別の目的で使用していたためブロックが存在しなかった回数。FAULTS は、特定のアプリケーションに対してキャッシュが大きすぎるか物理メモリが小さすぎる場合に発生します。FAULTS 発生率がすべての I/O 操作の 5%を超える場合、一般的にメモリを増やす必要があります。	増分
AUDIT-BUF-FORCES	ディスクプロセスがダーティキャッシュブロックに書き込む前にオーディットトレイルボリュームに書き込むことで、そのブロックを別の用途のために解放する必要があった回数。この値はゼロに近い値にする必要があります。	増分

表は続く

サブフィールド	説明	カウンタータイプ
BLKS-DIRTY-QTIME	キャッシュブロックがダーティになっていた時間。キャッシュブロックがダーティになるのは、バッファーを使用した書き込み操作がデータをキャッシュブロックに書き込むときです。ディスクプロセスがダーティキャッシュブロックをディスクに書き込んだ後、ブロックはクリーンになります。	キュー
WRITE-CLEANS	（ZMS 形式のみ）書き込み操作で、必要なブロックがキャッシュ内に見つかった回数。これにより、書き込みあたりの検出済みのクリーンなディスクの読み取りが回数が節約されます。	累積
WRITE-DIRTYS	（ZMS 形式のみ）書き込み操作で、必要なブロックがキャッシュ内でダーティであった回数。	累積
WRITE-MISSES	（ZMS 形式のみ）書き込み操作で必要なブロックがキャッシュに見つからない回数。書き込みを実行する前にディスク読み取りが必要です。	累積

Measure 出力では、各サブフィールドには DP2 キャッシュブロックのサイズごとに 1 つの 8 つの値があります。

フィールドラベル	キャッシュブロックサイズ
C0	512 バイト
C1	1024 バイト
C2	2048 バイト
C3	4096 バイト
C4	8192 バイト
C5	16346 バイト
C6	32768 バイト
C7	65536 バイト

CONTROL-POINT-WRITES

TMF 制御ポイントでディスクプロセスがディスクに書き込むダーティキャッシュブロックの数。TMF 監査の詳細については、*TMF Operations and Recovery Guide* を参照してください。

カウンタータイプ：増分。

CONTROL-POINTS

ディスクプロセスが TMF 制御ポイントレコードを監査証跡ファイルに書き込んだ回数。TMF 監査の詳細については、*TMF Operations and Recovery Guide* を参照してください。

カウンタータイプ：増分。

DBIO-READS

要求側のアプリケーションプロセッサーへのダイレクトバルク I/O のために有効化された測定対象デバイスへの読み取り操作の数。このカウンターは READS のサブセットです。

カウンタータイプ：増分。

DBIO-WRITES

要求側のアプリケーションプロセッサーからのダイレクトバルク I/O のために有効化された測定対象デバイスへの書き込み操作の数。このカウンターは WRITES のサブセットです。

カウンタータイプ：増分。

DEFERRED-QTIME

遅延の影響を受ける要求が実際に費やした遅延時間（マイクロ秒単位）。このカウンターは DEFREQ-QTIME のサブセットです。

カウンタータイプ：キュー。

DEFREQ-QTIME

遅延の影響を受ける要求がディスクプロセス内部キューで費やした時間（マイクロ秒単位）（現在アクティブな要求または遅延した要求を含む）。このカウンターは REQUEST-QTIME のサブセットです。

カウンタータイプ：キュー。

DEFREQS

遅延の影響を受けるディスクプロセスによって受信された要求の数。このカウンターは REQUESTS のサブセットです。

カウンタータイプ：増分。

DEVICE-QBUSY-TIME

任意の数またはタイプの要求が、このプロセッサーのこのディスクのキューに入れられた状態で経過した時間。

書き込みキューと読み取りキュー内の並列動作のため、WRITE-QBUSY-TIME と READ-QBUSY-TIME を加算しても DEVICE-QBUSY-TIME に等しくなりません。詳しくは、**G シリーズの DISC エンティティの使用上の注意**(247 ページ)を参照してください。

カウンタータイプ：キュービジー。

ENDING-FREE-BLOCKS

TO time で指定された時間におけるボリューム上の空きブロックの数。カウンターはプライマリパスレコードに対してのみ更新されます。RATE 属性は、このカウンターの表示方法には影響がありません。プライマリパスは LIST DISC コマンド（MeasureH03 以降の PVU）または `SCF INFO`*`DISCNAME`*`, DETAIL` コマンドで取得されます。

カウンタータイプ：スナップショット。

ENDING-FREE-SPACE

TO time で指定された時間におけるボリューム上の空きバイト数。このカウンターはプライマリパスレコードに対してのみ更新されます。RATE 属性は、このカウンターの表示方法には影響がありません。プライマリパスは LIST DISC コマンド（MeasureH03 以降の PVU）または `SCF INFO`*`DISCNAME`*`, DETAIL` コマンドで取得されます。

カウンタータイプ：スナップショット。

FREE-SPACE-IOS

ディスクの空き領域テーブルに対する I/O 操作の数。

ディスクの空き領域テーブルには、ディスク上で利用可能なそれぞれの空きブロックが示されます。このディスクプロセスは、ファイルエクステントが割り当てられたり割り当て解除されたときにテーブルを変更します。適切なエクステントサイズを選択し、不要なファイルを作成しないようにして、不要なファイルを削除することで、空き領域 I/O 数を軽減できます。

カウンタータイプ：増分。

INPUT-BYTES

ディスクから読み取られたバイト数。プログラムによる読み取り操作に加え、書き込み操作やさまざまなコマンド（LISTHEADERS や LISTDEFECTS など）でもディスクからバイトが読み取られます。

ディスクプロセスは、ファイルの構造と読み込まれるデータの量に応じて、読み取り操作中に 1 つまたは複数のデータブロックを読み取ります。

- 非構造化ファイルの場合、ディスクプロセスは、少なくとも 1 つのデータブロックを読み取ります。レコードがブロック境界をまたいでいる場合、ディスクプロセスは 2 つのデータブロックを読み取る必要があります。また、部分的に使用されたブロックをディスクに書き込む場合、ディスクプロセスはブロックを読み取り、新しいデータをブロックにマージし、ブロックをディスクに書き込む必要があります。
- 構造化ファイルの場合、ディスクプロセスは、ファイルブロックサイズで指定されたとおりにデータブロックを読み取ります。
- キーシーケンスファイルの場合、データブロックを読み取るために、ディスクプロセスは 1 つまたは複数のインデックスファイルを読み取らなければならない可能性もあります。

I/O 操作前に I/O プロセスがこのカウンターを変更するので、書き込みが失敗した場合、バイト数が正確でなくなることがあります。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、INPUT-BYTES-F フィールドが INPUT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

OUTPUT-BYTES

ディスクに書き込まれたバイト数。プログラムによる書き込み操作だけでなく、多くのコマンド（FORMAT や LOADMICROCODE など）もディスクに書き込みを行います。

ディスクプロセスは、ファイルの構造と書き込まれるデータの量に応じて、書き込み操作中に 1 つまたは複数のブロックのデータを書き込みます。

- 非構造化ファイルの場合、ディスクプロセスは、少なくとも 1 ブロックのデータを書き込みます。レコードがブロック境界をまたいでいる場合、ディスクプロセスは 2 ブロックのデータを書き込む必要があります。さらに、部分的に埋められているブロックをディスクに書き込む場合、ディスクプロセスは、ブロックを読み取り（INPUT-BYTES によってカウントされる）、新しいデータをそのブロックにマージしてから、そのブロックをディスクに書き込む必要があります。
- 構造化ファイルの場合、ディスクプロセスは、ファイルブロックサイズで指定されたとおりに、1 ブロックのデータを読み取ります。
- キーシーケンスファイルの場合、ディスクプロセスは、データブロックに書き込みを行い、インデックスブロックの更新が必要な場合もあります。

ディスクプロセスは、更新を収集して必要なときにのみそれらを書き込むため、バッファーに格納される書き込みは I/O を節約できます。

I/O プロセスは I/O 操作の前にこのカウンターを変更するため、書き込みが失敗するとこのバイトカウントは正確ではなくなることがあります。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、OUTPUT-BYTES-F フィールドが OUTPUT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

READ-QTIME

このディスクのキューに入れられた読み取り要求が費やした合計時間。

カウンタータイプ：キュー。

READ-QBUSY-TIME

任意の数の読み取り要求が、このプロセッサーのこのディスクのキューに入れられた状態で経過した時間。

カウンタータイプ：キュービジー。

READS

ディスクプロセスによって実行された（ディスクからメモリへの）物理的な読み取り操作の数。プログラムによる読み取り操作だけでなく、多くのコマンド（LISTHEADERS や LISTDEFECTS など）もディスクからの読み取りを行います。

カウンタータイプ：増分。

REQUEST-QTIME

要求がディスクプロセスの内部キューで費やした時間（現在アクティブな要求を含む）。要求を読み取るとき、ディスクプロセスはその要求を内部キューに入れます。要求を処理した後、ディスクプロセスは内部キューから要求を削除します。ディスクプロセスの外部キュー（ディスクプロセスによって読み取られるのを待機している要求）の長さを確認するには、ディスクプロセスを測定し、PROCESS エンティティの RECV-QTIME カウンターを調べます。

カウンタータイプ：キュー。

REQUESTS

ディスクプロセスが受信した I/O 要求（READ、WRITE、FILEINFO）の数。FILE エンティティは、これらの要求のほとんどをファイルごとに記録します。ディスクプロセスが読み取るまで要求がキューに入れられていた時間を確認するには、プロセスを測定し、PROCESS エンティティの RECV-QTIME カウンターを調べます。要求を読み取った後、ディスクプロセスはその要求を内部キューに入れます。REQUEST-QTIME は内部キューの長さを測定します。

カウンタータイプ：増分。

REQUESTS-BLOCKED

要求されたファイルまたはレコードがすでにロックされていたために、操作が待機しなければならなかった回数。通常、レコードは、LOCKFILE 要求、LOCKREC 要求、または TMF 監査操作によってロックされます。

カウンタータイプ：増分。

STARTING-FREE-BLOCKS

測定開始時にボリュームの空き領域テーブルにある空きエントリーの数。空き領域テーブルは、DP2 によって維持される内部データ構造です。このカウンターは、REPORT STYLE が ZMS のときに MEASCOM によって表示されます。IF 句と BY 句だけでなく、PLOT コマンドでも使用できます。

このカウンターはプライマリパスレコードに対してのみ更新されます。プライマリパスは、LIST DISC コマンド（Measure H03 以降の PVU）または SCF INFO *DISCNAME*, DETAIL コマンドで取得できます。

カウンタータイプ：スナップショット。

STARTING-FREE-SPACE

測定開始時のボリューム上の空きバイト数。このカウンターは、REPORT STYLE が ZMS のときに MEASCOM によって表示されます。IF 句と BY 句だけでなく、PLOT コマンドでも使用できます。

このカウンターはプライマリパスレコードに対してのみ更新されます。プライマリパスは、LIST DISC コマンド（Measure H03 以降の PVU）または SCF INFO *DISCNAME*, DETAIL コマンドで取得できます。

カウンタータイプ：スナップショット。

SWAPS

メモリマネージャーで実行されたスワップ操作（メモリへのスワップインとメモリからのスワップアウトの両方）の数。

ファイルブロックサイズに依存しない CPU スワップとは異なり、ディスクプロセスはファイルのブロックサイズに基づいて I/O を実行するため、必要なデータをスワップするために複数回の I/O を実行しなければならない場合があります。DISC SWAPS カウンターは I/O ごとに加算されます。そのため、ディスクスワップの合計は CPU スワップの合計よりも大きくなることがあります。更新されたページのみがスワップされます。コードページは更新されることがないために、メモリからスワップアウトされることはありません。

カウンタータイプ：増分。

VOLSEM-QTIME

要求がボリュームセマフォ（排他モードおよび共有モード）のために待機するのに費やした時間。

REPORT RATE が OFF の場合、このカウンターの値はボリュームセマフォの合計キュー時間に等しくなります。REPORT RATE が ON の場合、このカウンターの値はキュー内の要求の平均数に等しくなります。

通常の状況では、このカウンターの値はゼロに近い値になるはずです。そうでない場合は、一部の操作によってボリュームセマフォが排他モードのままになっているため、共有モードのボリュームセマフォを必要とする他のすべての要求操作を待機させていることを示します。

カウンタータイプ：キュー。

WRITE-QBUSY-TIME

任意の数の書き込み要求が、このプロセッサーのこのディスクのキューに入れられた状態で経過した時間。

カウンタータイプ：キュービジー。

WRITE-QTIME

このディスクのキューに入れられた書き込み要求が費やした合計時間。

カウンタータイプ：キュー。

WRITES

ディスクプロセスによって実行された（メモリからディスクへの）物理的な書き込み操作の数。プログラムによる書き込み操作だけでなく、多くのコマンド（FORMAT や LOADMICROCODE など）もディスクに書き込みを行います。

カウンタータイプ：増分。

DISC エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードのみで使用されます。

フィールドはアルファベット順に一覧表示されます。

CHANNEL

（Legacy 形式のみ）D シリーズの RVU で、デバイスのチャネル番号。G シリーズ以降の RVU では、再定義されます。SERVERNET を参照してください。

CTRL

（Legacy 形式のみ）D シリーズの RVU では、デバイスのコントローラー番号。G シリーズの RVU では現在使用されていません。ゼロを返します。

UNIT

（Legacy 形式のみ）D シリーズの RVU では、デバイスのユニット番号を表します。G シリーズの RVU では現在使用されていません。ゼロを返します。

DISC エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードだけで使用されます。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

ABLKS-INUSE-MAX

（Legacy 形式のみ）現在は使用されていません。

ABLKS-INUSE-QTIME

（Legacy 形式のみ）現在は使用されていません。

C

（Legacy 形式のみ）ディスクキャッシュメトリック。C のフィールドは Legacy 形式の DDL と ZMS 形式の DDL レコードの両方に含まれますが、サブフィールドの内容およびサブフィールドのシーケンスはレコードのスタイルに基づいて異なります。ZMS 形式の DDL レコードのサブフィールドの内容と C フィールドのシーケンスについては、**DISC エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義**(235 ページ)を参照してください。Legacy 形式の DDL レコードで、このフィールドは以下のサブフィールドに分かれています。

サブフィールド	説明	カウンタータイプ
HITS	読み取り操作またはバッファー書き込み操作で必要なブロックがキャッシュ内に見つかり、ディスク I/O が節約された回数。このカウンターには、読み取りおよび書き込みの両方の操作のキャッシュヒットが含まれます。	増分
MISSES	読み取り操作時に必要なブロックがキャッシュ内に見つからず、ディスク I/O を発生させた回数。キャッシュを経由しない読み取り操作はこのカウンターに含まれません。	増分

表は続く

サブフィールド	説明	カウンタータイプ
FAULTS	必要なブロックがキャッシュ内に存在することがキャッシュテーブルで示されたが、メモリマネージャーがそれを別の目的で使用していたためブロックが存在しなかった回数。FAULTS は、特定のアプリケーションに対してキャッシュが大きすぎるか物理メモリが小さすぎる場合に発生します。FAULTS 発生率がすべての I/O 操作の 5%を超える場合、一般的にメモリを増やす必要があります。	増分
AUDIT-BUF-FORCES	ディスクプロセスがダーティキャッシュブロックに書き込む前にオーディットトレイルボリュームに書き込むことで、そのブロックを別の用途のために解放する必要があった回数。この値はゼロに近い値にする必要があります。	増分
BLKS	割り当てられているキャッシュブロックの数。TOTALS ONLY または TOTALS INCLUDE が使用されている場合、この値は割り当てられているキャッシュブロックの最大数です。BLKS はカウンターではなく、内部のディスクプロセスデータ構造から取得した値です。	なし
BLKS-DIRTY-QTIME	キャッシュブロックがダーティになっていた時間。キャッシュブロックがダーティになるのは、バッファーを使用した書き込み操作がデータをキャッシュブロックに書き込むときです。ディスクプロセスがダーティキャッシュブロックをディスクに書き込んだ後、ブロックはクリーンになります。	キュー
BLKS-DIRTY-MAX	BLKS-DIRTY-QTIME カウンターで記述されるキューの項目の最大数。	最大キュー

Measure 出力では、各サブフィールドには DP2 キャッシュブロックのサイズごとに 1 つの 4 つの値があります。

フィールドラベル	キャッシュブロックサイズ
C0	512 バイト
C1	1024 バイト
C2	2048 バイト
C3	4096 バイト

CBLKS-INUSE-MAX

（Legacy 形式のみ）現在は使用されていません。

CBLKS-INUSE-QTIME

（Legacy 形式のみ）現在は使用されていません。

CN

（Legacy 形式のみ）ディスクキャッシュメトリック。このフィールドは以下のサブフィールドに分かれています。

サブフィールド	説明	カウンタータイプ
BLKS	割り当てられているキャッシュブロックの数。TOTALS ONLY または TOTALS INCLUDE が使用されている場合、この値は割り当てられているキャッシュブロックの最大数です。BLKS はカウンターではなく、内部のディスクプロセスデータ構造から取得した値です。	なし
BLKS-INUSE-START	測定の最初に使用されていたキャッシュブロックの数。この値は構造化された DISC レコードの一部ですが、Legacy 形式の DISC レポートには表示されません。TOTALS ONLY または TOTALS INCLUDE が使用されている場合、この値は測定の最初に使用されるキャッシュブロックの合計数です。	スナップショット
BLKS-INUSE-END	測定の最後に、または測定間隔コピー時間に使用されていたキャッシュブロックの数。TOTALS ONLY または TOTALS INCLUDE が使用されている場合、この値は測定の最後にまたは測定間隔コピー時間に使用されるキャッシュブロックの合計数です。	スナップショット

Measure 出力では、各サブフィールドには DP2 キャッシュブロックのサイズごとに 1 つの 4 つの値があります。

フィールドラベル	キャッシュブロックサイズ
C0	512 バイト
C1	1024 バイト
C2	2048 バイト
C3	4096 バイト

CW

（Legacy 形式のみ）キャッシュ書き込みメトリック。このフィールドは以下のサブフィールドに分かれています。

サブフィールド	説明	カウンタータイプ
WRITE-DIRTYS	書き込み操作で必要なブロックがキャッシュ内でダーティであった回数。	累積
WRITE-CLEANS	書き込み操作で、必要なブロックがキャッシュ内でクリーンであった回数。これにより、書き込みあたりのディスク読み取りが節約されます。	累積

表は続く

サブフィールド	説明	カウンタータイプ
WRITE-MISSES	書き込み操作で必要なブロックがキャッシュに見つからない回数。書き込みを実行する前にディスク読み取りが必要です。	累積
BLOCK-SPLITS	ボリュームの（ブロックサイズあたりの）ブロック分割の数。	増分

Measure 出力では、各サブフィールドには DP2 キャッシュブロックのサイズごとに 1 つの 4 つの値があります。

フィールド名	キャッシュブロックサイズ
C0	512 バイト
C1	1024 バイト
C2	2048 バイト
C3	4096 バイト

DEFERRED-QLEN-MAX

（Legacy 形式のみ）DEFERRED-QTIME カウンターで記述されるキューの項目の最大数。

H シリーズ以降の RVU では、このカウンターの値は 1 です。

カウンタータイプ：最大キュー。

DEFREQ-QLEN-MAX

（Legacy 形式のみ）DEFREQ-TIME カウンターで記述されるキューの項目の最大数。

H シリーズ以降の RVU では、このカウンターの値は 1 です。

カウンタータイプ：最大キュー。

INPUT-BYTES-F

（Legacy 形式のみ）G シリーズの RVU では、INPUT-BYTES と同じですが、より大きな値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

OUTPUT-BYTES-F

（Legacy 形式のみ）G シリーズの RVU では、OUTPUT-BYTES と同じですが、より大きな値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

READ-BUSY-TIME

（Legacy 形式のみ）D シリーズの RVU では、ディスクからの読み取りに費やした時間。このカウンターには、データの読み取りと、データを読み取るためのディスクヘッドの位置決めに費やした時間が含まれます。

G シリーズの RVU では現在使用されていません。ゼロを返します。

カウンタータイプ：ビジー。

READ-QLEN-MAX

（Legacy 形式のみ）このディスクのキューに入れられた読み取り要求の最大数。

カウンタータイプ：最大キュー。

REQUEST-QLEN-MAX

（Legacy 形式のみ）REQUEST-QTIME カウンターで記述されるキューの項目の最大数。

カウンタータイプ：最大キュー。

SEEK-BUSY-TIME

（Legacy 形式のみ）現在は使用されていません。ゼロを返します。読み取り操作と書き込み操作では、すべてのシーク操作は暗黙的です。

SEEKS

（Legacy 形式のみ）現在は使用されていません。ゼロを返します。読み取り操作と書き込み操作では、すべてのシーク操作は暗黙的です。

VOLSEM-QLEN-MAX

（Legacy 形式のみ）VOLSEM-QTIME カウンタが記述するキュー内の要求の最大数。

カウンタータイプ：最大キュー。

WRITE-BUSY-TIME

（Legacy 形式のみ）D シリーズの RVU では、ディスクへの書き込みに費やした時間を表します。このカウンターには、データの書き込みと、データを書き込むためのディスクヘッドの位置決めに費やした時間が含まれます。

G シリーズの RVU では現在使用されていません。ゼロを返します。

カウンタータイプ：ビジー。

WRITE-QLEN-MAX

（Legacy 形式のみ）測定が開始された後の任意の時点で、このディスクのキューに入れられた書き込み要求の最大数。

カウンタータイプ：最大キュー。

すべての DISC エンティティの使用上の注意

- DISC カウンターは、ファイルアクティビティと内部アクティビティ（空き領域テーブルへの書き込み、取り消し領域への書き込み、ファイルラベルやボリュームラベルの書き込みなど）の両方を含む、すべてのディスクアクティビティを追跡します。2 つ以上のディスクのワークロードを比較してバランスを調整するには、DISC カウンターを使用します。

  I/O およびキャッシュ機能については、該当するシステムの説明マニュアルを参照してください。TMF 監査およびディスクに書き込まれる監査証跡については、*TMF Operations and Recovery Guide* を参照してください。
- Measure は、SMF 仮想ディスクプロセス（VDP）を DISC エンティティとして測定しません。たとえば、$V1 が VDP の場合、コマンド$ADD DISK $V1 は、$V1 のデータも$V1 の下に構成された物理ディスクのデータも収集しません。ただし、エラーは返されません。

- ZMSDISC レコードの SUBSYSTEM-VERSION は DP2 によって提供されます。
- DISC レポートには、DISC カウンターから計算されるフィールドが含まれます。これらのフィールドは、DISC DDL レコードでは定義されていません。

  DISC-BUSY-TIME

  （Legacy 形式のみ）DISC WRITE-BUSY-TIME および READ-BUSY-TIME カウンターの合計。

  DISC-RATE

  物理ドライブに対する READ 操作と WRITE 操作の合計速度。この値は BRIEF 形式のレポートに含まれています。

  カウンタータイプ：増分。

  REQUEST-ART

  （ZMS 形式のみ）各要求が待機または DP2 による処理に費やした平均時間。REQUEST-QTIME は REQUESTS で除算されます。

  カウンタータイプ：応答時間。

  DEVICE-AST

  （ZMS 形式のみ）各要求が待機または DP2 による処理に費やした平均時間。DEVICE-QBUSY-TIME は READS と WRITES の合計で除算されます。

  カウンタータイプ：サービス時間。

H シリーズ以降の DISC エンティティの使用上の注意

- H06.23/J06.12 以降の RVU の CLIM 接続のデバイスに対して DISC エンティティを指定する場合、オプションのパラメーターとして *partition* を含めるように更新されています。このエンティティ指定は ADD、DELETE、LIST、LISTACTIVE、および LISTALL コマンドで使用できます。さらに、測定時に *partition* を指定した場合、Measure は INFO および INFO MEASUREMENT コマンドでパーティション番号を表示します。
- Measure H03/J01 以降の PVU の CLIM 接続のデバイスに対して DISC エンティティを指定する場合、*lun* および *path* が追加され、*scsi id* を使用しないように更新されています。FCSA デバイスと同様に、このエンティティ指定は ADD、DELETE、LIST、LISTACTIVE、および LISTALL コマンドで使用できます。
- H シリーズ以降の RVU では、すべての byte-count フィールドが 64 ビットに対応しています。-F で終わるフィールド名は、ZMS 形式では使用されなくなりましたが、Legacy 形式でデータを要求するアプリケーションでは引き続き使用できます。

G シリーズの DISC エンティティの使用上の注意

- デバイスのビジー時間を測定するには、DEVICE-QBUSY-TIME カウンターを使用します。DEVICE-QBUSY-TIME は D シリーズのフィールド DISC-BUSY-TIME に代わるものです。

  D シリーズの RVU では、DISC-BUSY-TIME は DISC WRITE-BUSY-TIME と READ-BUSY-TIME カウンターの値を加算することによって計算されていました。一方、G シリーズの RVU の DEVICE-QBUSY-TIME は、WRITE-QBUSY-TIME と READ-QBUSY-TIME の合計とは等しくなりません。ServerNet 環境では、読み取りキューと書き込みキューにアクティブな要求を同時に保持することができます。そのため、READ-QBUSY-TIME カウンターと WRITE-QBUSY-TIME カウンターの間に重複が存在することがあり、それらの値を加算しても正確なビジー時間の測定は得られません。
- 64 ビットのバイトカウントフィールド（-F で終わるフィールド）は、従来の 32 ビットのバイトカウントフィールドと同じデータを収集します。たとえば、64 ビットフィールドの INPUT-BYTES-F は、32 ビットフィールドの INPUT-BYTES と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

  32 ビットフィールドは現在アクティブであり、引き続き値を返します フィールドのオーバーフローが発生しない場合、32 ビットフィールドと 64 ビットのフィールドは同じ値を返します。32 ビットフィールドがオーバーフローした場合、対応する 64 ビットフィールドは正しい値を返し、32 ビットフィ

ールドは-1 の値を返します。測定対象のエンティティの ERROR フィールドも、オーバーフロー状態を示す-1 を返します。

64 ビットフィールドを使用するには、アプリケーションを変換してください。32 ビットのフィールドは、今後の PVU では使用できなくなる可能性があります。

MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットフィールドの名前を使用してください。たとえば、LIST DEVICE BY INPUT-BYTES-F ではなく LIST DEVICE BY INPUT-BYTES のコマンドを発行してください。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。

- ダイレクトバルク I/O 要求は、ディスクへの読み取りおよび書き込み要求としてカウントされます。READ-BUSY-QTIME、READ-QTIME、INPUT-BYTES、および類似するカウンターは、ダイレクトバルク I/O と標準 I/O の両方に対して更新されます。

  ディスクが標準とダイレクトバルクの両方の I/O に対して使用される場合は、DISC カウンターだけからダイレクトバルク I/O のバイトカウントを判別することはできません。この情報は、DBIO 転送を要求するアプリケーションプロセスごとの FILE レコードの詳細な分析によって提供される場合があります。
- 32 ビットの CN[N].BLKS フィールドは、以前の 16 ビットの C[N].BLKS フィールドと同じデータを収集します。フィールドのオーバーフローが発生しない場合、16 ビットフィールドと 32 ビットのフィールドは同じ値を返します。16 ビットがオーバーフローした場合、対応する 32 ビットフィールドは正しい値を返し、16 ビットフィールドは-1 の値を返します。測定エンティティの ERROR フィールドは、オーバーフロー状態を示す-2 を返します。
- Measure G08 以降の PVU の新しいカウンターは、アプリケーションの処理とプロセスの優先度の割り当てを行うときに、DP2 の混合ワークロードアルゴリズムの有効性をより適切に評価します。Measure G08 のカウンター（DEFREQS、DEFREQ-QLEN-MAX、DEFERRED-QTIME、DEFERRED-QLEN-MAX）は、延期の対象となる要求の要求時間と延期時間を追跡します。

  混合ワークロード設計では、優先度の高いワークロードと同時に優先度の低い SQL クエリを実行する場合の影響をある程度制御できます。SQL クエリの優先度が低いほど、優先度の高い応答時間への影響は小さくなります。同じ CPU に複数のボリュームがある並列クエリでは特に、優先度が低くなるほどクエリの速度も遅くなります。

  ボリュームに対する優先度が高いワークロードが同時にアクティブである場合、ユーザーはクエリを延期することでこの混在ワークロードを制御できます。優先度が 150 の場合は延期が短く、クエリの優先度が低くなると長くなります。延期を無効にするには、150 より大きい優先度でクエリを実行します。

  延期可能 q-time と延期済み q-time の Measure カウンターは、延期の量の兆候を示します。SQL クエリを完了するのに必要な時間が長すぎる場合は、延期可能な q-time と延期済み q-time の比率が大きくなります。優先度の低い SQL クエリの動作中に応答時間がかなり長くなる場合は、延期可能 q-time と延期済み q-time の比率が低くなります。SQL クエリの優先度の調整は、この比率の変化に対応させてください。

  プラットフォームやアプリケーションの変更によって、一次的に SQL クエリアクティビティの優先度に対して何らかの調整が必要になる場合があります。クエリが最小限のデータの処理を終えるまで、この設計によってクエリアクティビティが延期されることはありません。ただし、OLTP タイプの SQL クエリアプリケーションの中には、延期を避けるために 150 より高い優先度で実行する必要があるものもあります。

  今後の RVU では、アプリケーションが CPU およびディスク容量の増加を最大限に活用できるように、混合ワークロード設計が調整される可能性があります。設計における調整では、混合ワークロード環境に対する優先度の調整が必要になる可能性があります。混合ワークロードアルゴリズムの最新の変更には、クエリの速度を上げるための調整が含まれていますが、応答時間の向上はわずかです。

  注記:

  優先度の設定を使用して混合ワークロードアプリケーションを制御するためのガイドラインについては、ホワイトペーパー Mixed Workload Priority Guide, Version Id.: 1.0, Date: September 2001 を参照してください。

D シリーズの DISC エンティティの使用上の注意

アプリケーションがすべてのディスク（ADD DISC *）を指定している場合や*$disk* または*$disk (cpu)*のみを指定している場合は、D シリーズの測定アプリケーションを使用して NonStop S シリーズサーバーのディスクを測定することができます。アプリケーションが *controller*、*channel*、または *unit* を指定している場合に NonStop S シリーズのディスクを測定するには、エンティティ識別子を変更する必要があります。

例

コマンド	説明
ADD DISC $SYSTEM	$SYSTEM という名前のディスクを測定します（すべてのパス）
ADD DISC *	すべてのディスクを測定します
ADD DISC $* (2)	CPU 2 のすべてのディスクを測定します
ADD DISC $* (*, *, 110, 3, 1, 2)	GMS が 110,3,1 で、scsi-id が 2 のコントローラーに接続されているすべてのディスクを測定します（ServerNet/DA または FCSA ストレージデバイスのみ）
ADD DISC $SYSTEM-B	$SYSTEM のバックアップパスを測定します
ADD DISC $* (*,*,CLIM3)	CLIM3 という名前の SAC に接続されているすべてのディスクを測定します

DISCOPEN

DISCOPEN エンティティは、指定されたファイルのすべての一意のオープンについてディスクプロセスによって実行された I/O 操作を測定します。測定されるファイルは、測定中のシステムに対してローカルである必要がありますが、測定されるファイルにアクセスするプロセスはローカルでもリモートでもかまいません。

注記:

MEASCOM コマンドでは、HELP を除いてキーワード DISKOPEN は DISCOPEN と交換可能です。

Measure G09 以降の PVU では、DISCOPEN エンティティで OSS ファイルのパス名の使用がサポートされます。これは DISCOPEN レポートで OSS ファイルパス名を表示し、外部の構造化レコードと OSSNAMES 構造化レコード出力との直接マッピングを提供します。

Measure H01 以降の PVU では、MEASCOM は DISCOPEN エンティティ指定の ANSI SQL 名を受け入れます。

トピック
DISCOPEN エンティティのエンティティ指定構文
DISCOPEN エンティティの DDL レコード
DISCOPEN エンティティの DDL フィールドの定義

表は続く

トピック

DISCOPEN エンティティの使用上の注意

DISCOPEN 測定の例

DISCOPEN エンティティのエンティティ指定構文

DISCOPEN エンティティを記述するには、次の構文を使用します。

```
DISCOPEN entity-spec [,entity-spec] ...
```

DISCOPEN

1 つまたは複数のディスクファイルオープンへの物理的アクセスについての情報を収集します。測定されるファイルはローカルである必要があります。ただし、そのファイルにアクセスするプロセスは、ローカル、リモート、またはその両方であってかまいません。

entity-spec

次のように定義されます。

```
diskfile [ ( [ \system-name, ] cpu, pin )]
```

指定する値は次の通りです。

diskfile

測定対象のファイルです。*diskfile* は、以下のいずれかのように指定します。

```
{ * }
{ [[$device.]subvolume.]filename[:crvsn] }
{ [$device.]#tempfile }
{ "pathname" }
{ '{TABLE|INDEX} [[catalog.]schema.]object[ PARTITION partition]' }
{ 'SCHEMA [catalog.]schema' }
{ 'CATALOG {catalog|*}'
```

指定する値は次の通りです。

*****

すべてのファイルオープンを測定します。

$*device*

測定対象のファイルが含まれるボリューム（デバイス）の名前です。すべてのボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのボリュームです。

subvolume

測定対象のファイルが含まれるサブボリュームの名前です。すべてのサブボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのサブボリュームです。

filename

測定対象のファイルの名前です。一時ファイルを除くすべてのファイルを指定するには、アスタリスク（*）を使用します。一時ファイルは別途指定されます。

:crvsn

（Measure G09 以降の PVU）一意のファイル名を形成するために必要な、タイムスタンプ、作成バージョンのシリアル番号（CRVSN）、またはファイル名拡張子です。このオプションを使

用して、ファイル名の一意性を保証します。CRVSN は Measure レポートおよび LISTGNAME コマンドから使用できます。

#tempfile

測定対象の一時ファイルのファイル識別番号です。すべての一時ファイルを指定するには、アスタリスク（*）を使用します。

"*pathname*"

（Measure G09 以降の PVU）完全修飾または部分修飾のいずれかの OSS ファイルパス名です。スラッシュ（/）で始まらない OSS ファイルパス名は、部分的なパス名と見なされ、OSSPATH の現在の設定を前に指定することで拡張されます。/G で始まる OSS ファイルパス名はサポートされていないため、代わりに Guardian ファイル名を使用してください。

注記:

OSS ファイルパス名は大文字と小文字が区別され、二重引用符（""）で囲んで指定する必要があります。有効な OSS ファイルパス名は、特定のファイル、または特定のディレクトリ内のファイルのセットを参照できます。ディレクトリが指定されている場合は、そのディレクトリ内のファイルのみが含まれます。指定したディレクトリの下位のディレクトリ内のファイルは含まれません。

catalog*、*schema*、*object*、*partition

（Measure H01 以降の PVU）ANSI SQL 識別子です。省略すると、catalog および schema フィールドは、MEASCOM の catalog および schema の環境値によって解決されます。名前空間キーワード TABLE または INDEX が省略された場合、デフォルトとして TABLE が推定されます。

ワイルドカードの構文は次のとおりです。

`TABLE [[catalogC.]schemaS.]tableT` – tableT 内のすべてのパーティション
`INDEX [[catalogC.]schemaS.]indexI` – indexI 内のすべてのインデックスパーティション
`SCHEMA [catalogC.]schemaS` – schemaS 内のすべてのオブジェクト
`CATALOG catalogC` – catalogC 内のすべてのオブジェクト
`CATALOG *` – ノード上で表示されるすべてのオブジェクト

system-name

オープナープロセスを実行するシステムの名前です。すべてのシステムを指定するには、アスタリスク（*）を使用します。*cpu* および *pin* が指定されない限り、デフォルトはすべてのシステムとなりますが、それらが指定された場合、デフォルトはローカルシステムです。

cpu

オープナープロセスが実行されている CPU の番号です。デフォルトは、すべての CPU です。オープナープロセスの所有者を指定するには *cpu* および *pin* を使用します。

pin

オープナープロセスのプロセス識別番号です。すべてのプロセスを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのプロセスです。

DISCOPEN エンティティの DDL レコード（ZMS 形式）

DISCOPEN エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsdopen-id.
 02 opener-cpu                type binary 16 unsigned.
 02 opener-pin                type binary 16 unsigned.
 02 ocb-number                type binary 16 unsigned.
 02 file-type                 type binary 16 unsigned.
 02 opener-system-name        type character 8.
 02 file-name.
    03 volume                 type character 8.
    03 subvol                 type character 8.
    03 filename               type character 8.
 02 device-name               type character 8.
 02 storage-pool              type character 8.
 02 file-name-MID.
    03 PATHID                 type character 24.
    03 CRVSN                  type character 6.
 02 record-type               type binary 16 unsigned.

 02 ocb32-number              type binary 32 unsigned.
 02 reserved-1                type character 12.

end
```

カウンターフィールドの DDL 定義

```
 02 driver-input-calls        type binary 64.
 02 driver-output-calls       type binary 64.
 02 cache-hits                type binary 64.
 02 cache-write-hits          type binary 64.
 02 block-splits              type binary 64.
 02 requests-blocked          type binary 64.
 02 requests                  type binary 64.
 02 lockwait-time             type binary 64.
 02 max-lockwait-time         type binary 64.
 02 lock-timeouts             type binary 64.
 02 lock-bounces              type binary 64.
 02 cache-write-cleans        type binary 64.
 02 DBIO-input-calls          type binary 64.
 02 DBIO-output-calls         type binary 64.
 02 SQL-operation-time        type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsdopen. FILE is "zmsdopen" ENTRY-SEQUENCED.
 02 hdr                       type zmsheader.
 02 ctr                       type zmsdopen-ctrs.
 02 id                        type zmsdopen-id.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

DISCOPEN エンティティの DDL レコード（Legacy 形式）

DISCOPEN エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G シリーズ DISCOPEN エンティティの DDL レコードは、次の点を除いて D シリーズ DISCOPEN エンティティのレコードと同じです。

- ダイレクトバルク I/O をサポートする追加されたカウンター（DRIVER-INPUT-CALLS などの既存の I/O カウンターもダイレクトバルク I/O 操作を含みます）
- SQL 操作時間の代わりとしての DP2 ビジー時間の表示
- OSS ファイルパス名のサポートのための新しい識別子

```
RECORD discopen. FILE is "discopen" ENTRY-SEQUENCED.
                        .
                        .
                        .
```

(error、time items、および measurement identification items については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください)

```
                        .
                        .
                        .
* entity identification items:
 02 opener-cpu                  type binary 16 unsigned.
 02 opener-pin                  type binary 16 unsigned.
 02 ocb-number                  type binary 16 unsigned.
 02 opener-system-name          type character 8.
 02 file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 file-type                   type binary 16 unsigned.

* counter value items:
 02 driver-input-calls          type binary 32 unsigned.
 02 driver-output-calls         type binary 32 unsigned.
 02 cache-hits                  type binary 32 unsigned.
 02 cache-write-hits            type binary 32 unsigned.
 02 block-splits                type binary 32 unsigned.
 02 requests-blocked            type binary 32 unsigned.

* new entity identification items for D10:
 02 record-type                 type binary 16 unsigned.

* new counter value items for D10:
 02 requests                    type binary 32 unsigned.
 02 lockwait-time               type binary 64.
 02 max-lockwait-time           type binary 64.
 02 lock-timeouts               type binary 32 unsigned.
 02 lock-bounces                type binary 32 unsigned.
 02 cache-write-cleans          type binary 32 unsigned.

* SMS changes:
 02 device-name                 type character 8.
 02 storage-pool                type character 8.

 02 dbio-input-calls            type binary 32 unsigned.
 02 dbio-output-calls           type binary 32 unsigned.

* New counters for G08:
 02 SQL-operation-time          type binary 64.
```

```
* New identifiers for OSS file pathname support:
 02 file-name-mid.
    03 pathid                  type character 24.
    03 crvsn                   type character 6.
end
```

DISCOPEN エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

DEVICE-NAME

測定ファイルが配置されているディスクデバイス。SMF ファイルの場合、このフィールドには、論理ファイル名に対応する物理的な場所が表示されます。NonStop ファイルの場合、このフィールドは、FILE-NAME VOLUME サブフィールドと同じです。

FILE-NAME

測定対象のファイルの名前。この名前は､NonStop ファイルか SMF ファイルのいずれかに適用できます。FILE-NAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、ディスクファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、ディスクファイルを開いたときに指定された物理ファイル名を表します。 SMF ファイルでは、ディスクファイルを開いたときに使用された、位置に依存しない論理ファイル名を表します。

FILE-NAME-MID

OSS ファイルまたは SQL/MX ファイルの内部形式表現。FILE-NAME-MID には次の 2 つのサブフィールドがあります。

PATHID	OSS ファイルまたは SQL/MX ファイルの内部形式表現。その他のファイルでは、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルまたは SQL/MX ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。その他のファイルでは、このフィールドにはゼロが含まれます。

FILE-TYPE

ファイルタイプを示す以下のいずれかの値。

0	非構造化
1	相対値

表は続く

2　エントリーシーケンス

3　キーシーケンス

OCB-NUMBER

65534 以下の場合は、オープン制御ブロック（OCB）の識別番号です。それ以外の場合、値は-1 になります。

OCB32-NUMBER

（ZMS 形式のみ、L02、J06 以降）オープン制御ブロック（OCB）の識別番号です。

OPENER-CPU

測定対象のファイルを開いたプロセスの CPU 番号。

OPENER-PIN

測定対象のファイルを開いたプロセスのプロセス識別番号。

OPENER-SYSTEM-NAME

測定対象のファイルを開いたシステムの名前。

RECORD-TYPE

DISCOPEN エンティティのインストゥルメンテーションを記録するために使用する内部カウンターレコードのタイプ。

値が 0 の場合、開かれたファイルは、一時ファイル、プログラム（オブジェクトファイル）、または EDIT ファイルのいずれかです。これらのタイプのファイルでは、短い形式の内部レコードが使用され、特定のカウンターの値は記録されません。詳しくは、**すべての DISCOPEN エンティティの使用上の注意**(257 ページ)を参照してください。

STORAGE-POOL

測定対象のディスクが割り当てられている SMF ストレージプールの名前（ストレージプールプロセスのプロセス名）。

DISCOPEN エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。

BLOCK-SPLITS

ファイルへの書き込み中に分割されたブロックの数。

ブロック分割は、レコードをキーシーケンスファイルに追加したりレコードを長くしたりして、データブロックにレコード全体を含めることができなくなった場合に発生します。データブロックを分割すると、データブロックの上にあるインデックスブロックも同時に分割されることがあります。

ブロック分割では複数の I/O 操作が必要になるため、ブロック分割を最小限に抑える必要がありますが、キーシーケンスファイルに多くの新しいレコードを挿入するときは、完全に回避することはできません。

カウンタータイプ：増分。

CACHE-HITS

読み取り操作で必要なブロックがキャッシュ内に見つかり、ディスク I/O が節約された回数。

部分的に使用されたブロックをディスクに書き込む場合、ディスクプロセスはブロックを読み取り（おそらくこのカウンターをインクリメントします）、新しいデータをブロックにマージし、ブロックをディスクに書き込む必要があります。新しいファイルに書き込む場合、ディスクプロセスは、スペアブロックがキャッシュにない場合を除き、キャッシュから新しいブロックを読み取ります（このカウンターをインクリメントします）。

DRIVER-INPUT-CALLS カウンターは、すべての読み取り操作（キャッシュヒットとキャッシュミスの両方）をカウントします。

カウンタータイプ：増分。

CACHE-WRITE-CLEANS

書き込み操作で必要なブロックがキャッシュ内に検出され、そのブロックがクリーンだった回数。

書き込み中に、ディスクプロセスは要求されたブロックをキャッシュ内で検索します。キャッシュ内にブロックがある場合、ブロックはクリーンまたはダーティな状態です。ディスクプロセスは、キャッシュブロックをディスクにフラッシュすることによってキャッシュブロックを定期的にクリーンアップします。

カウンタータイプ：増分。

CACHE-WRITE-HITS

書き込み操作で必要なブロックがキャッシュ内に検出され、ブロックがダーティだった回数。

ダーティブロックに書き込むことで、物理書き込み操作が節約されます。

カウンタータイプ：増分。

DBIO-INPUT-CALLS

要求側のプロセッサーへのダイレクトバルク I/O 読み取り操作の数。このカウンターは DRIVER-INPUT-CALLS のサブセットです。

カウンタータイプ：増分。

DBIO-OUTPUT-CALLS

要求側のプロセッサーからのダイレクトバルク I/O 書き込み操作の数。このカウンターは DRIVER-OUTPUT-CALLS のサブセットです。

カウンタータイプ：増分。

DRIVER-INPUT-CALLS

キャッシュヒットとキャッシュミスの両方の読み取り操作の数。物理的な読み取り操作の数は、DRIVER-INPUT-CALLS と CACHE-HITS の差です。

カウンタータイプ：増分。

DRIVER-OUTPUT-CALLS

書き込み操作数。書き込みカウントが 4096 バイト以下の非構造化ファイルへの書き込みおよび構造化ファイルの場合、物理ディスク書き込み操作の数は DRIVER-OUTPUT-CALLS と CACHE-WRITE-HITS の差です。

書き込みカウントが 4096 を超える（大規模な書き込み）非構造化ファイルへの書き込みの場合、物理ディスク書き込み操作の数は、DRIVER-OUTPUT-CALLS と、CACHE-WRITE-HITS および CACHE-WRITE-CLEANS の合計との差です。

カウンタータイプ：増分。

LOCK-BOUNCES

ディスクプロセスによりファイルシステムに返される返送ロック（FELOCKED またはエラー 73）の数。SQL テーブルの場合、返送ロックは、RETURN-IF-LOCKED オプションを付けて CONTROL TABLE コマンドを使用した結果です。Enscribe ファイルの場合、返送ロックは、Param1.<15> = 1（代替ロックモード）で SETMODE 関数 4 を使用した結果です。

カウンタータイプ：増分。

LOCK-TIMEOUTS

ロックでのタイムアウト（エラー 40）の数。要求のタイムアウト値に達すると、ファイルシステム（Enscribe および SQL/MP）は、ディスクプロセスにキャンセル要求を送信します。まだロックが承諾されるまで待機している要求がディスクプロセスで検出されると、その要求はロック待機キューから削除され、このカウンターが加算されます。

カウンタータイプ：増分。

LOCKWAIT-TIME

ロックの待機に費やした時間。

REQUESTS-BLOCKED カウンターは、ブロックされた要求の数、つまり要求がロックのキューに入れられた回数をカウントします。REPORT RATE 値がオンの場合、このカウンターは平均待機時間（LOCKWAIT-TIME を REQUESTS-BLOCKED で割った値）を返します。

カウンタータイプ：ロック待機。

MAX-LOCKWAIT-TIME

ロックあたりの最大待機時間。この数値は、アプリケーションでタイムアウト値を設定するガイドとして使用できます。REPORT RATE ON は、このカウンター値には影響しません。

カウンタータイプ：最大値。

REQUESTS

このオープンに対する要求の数。

カウンタータイプ：増分。

REQUESTS-BLOCKED

要求されたファイルまたはレコードがロックされていたために、I/O 操作が待機しなければならなかった回数。通常、レコードのロックは、LOCKFILE 要求、LOCKREC 要求、または TMF 監査操作によって行われます。

カウンタータイプ：増分。

SQL-OPERATION-TIME

この DISCOPEN の代わりに SQL サブセット要求で DP2 が動作していたときに累積された DP2 プロセスビジー時間。

カウンタータイプ：ビジー。

すべての DISCOPEN エンティティの使用上の注意

- Measure サブシステムは、ローカルエンティティだけを追跡します。そのため、DISCOPEN エンティティには、ローカルファイルで実行された I/O 操作のみが含まれます。ファイルにアクセスするプロセ

スは、測定の設定によっては、ローカルであることも、リモートであることも、その両方であることもあります。

- DISCOPEN カウンターレコードは、ディスクプロセスのオープン制御ブロック（OCB）エントリーに添付されます。バックアップディスクプロセスの OCB は測定されないため、プライマリディスクプロセスが無効か移動された場合は、DISCOPEN による測定を再起動してください。
- DISCOPEN エンティティの DDL レコードには、短い形式と長い形式の 2 つの形式があります。長いレコードには、ファイルロックを測定するいくつかの追加のカウンターが含まれています。すべてのファイルオープンがこれらの測定を必要とするわけではないため、MEASCTL プロセスは、ファイルタイプに基づいて測定時に短いレコードまたは長いレコードを割り当てます。短いレコードは、一時ファイル（$vol.#nnnnnnn）、EDIT ファイル（ファイルコード 101）、およびオブジェクトファイル（ファイルコード 100）に割り当てられます。長いレコードの形式は、その他のすべてのファイルタイプに割り当てられます。

  長いレコードの形式は、構造化された出力ファイルに使用されます。レコード内のデータが短い形式で収集されている場合、構造化ファイルでは次のカウンターフィールドにゼロが含まれます。

REQUESTS	MAX-LOCKWAIT-TIME	LOCK-BOUNCES
LOCKWAIT-TIME	LOCK-TIMEOUTS	CACHE-WRITE-CLEANS

- DISCOPEN を使用する場合と DISKFILE を使用する場合の留意事項は次のとおりです。
  - DISCOPEN は、ファイルオープンごとに別々のカウンターレコードを作成します。オープナープロセスは個別に識別されます。同じプロセスが 1 つのファイルを複数回開くと、複数の DISCOPEN レコードが作成されます。
  - DISKFILE は、1 つのファイルのディスクファイルアクセスを全体として測定します。DISKFILE は、測定対象のファイルごとに 1 つのカウンターレコードを作成し、個々のオープナープロセスは追跡しません。
  - 代替キーファイルのオープンなどの間接オープンの場合、DISCOPEN は別々のレコードを作成します。DISKFILE は、間接オープンの場合は別々のレコードを作成しません。
  - 測定中にファイルを閉じてから再度開いた場合､DISCOPEN と DISKFILE は両方とも新しいレコードを作成します。
  - DISCOPEN は、ディスクファイルアクティビティの詳細なビューを提供します。DISCOPEN カウンターは、キーシーケンスデータベースファイルの構造に関する潜在的な問題を検出する場合に特に便利です。
  - 高いレベルの詳細が不要である場合、DISKFILE では分析する必要があるデータ量を削減する一方で、ファイル構造と競合に関する潜在的な問題を検出することができます。

  ファイルへの論理アクセスを測定するには､FILE エンティティタイプを使用します。説明については、**FILE**(284 ページ)を参照してください。
- ミラーディスクへの書き込みでは、物理的な書き込みが 2 回行われた場合でも、要求のカウントは 1 回増分されます。

G シリーズの DISCOPEN エンティティの使用上の注意

- 1 つのディスクのすべての DISCOPEN レコードを合計する場合､SQL-OPERATION-TIME カウンターには、すべての DP2 処理時間およびすべての SQL 関連の処理時間が含まれません。多く DP2 処理アクティビティは、特定の DISCOPEN に関連付けることはできません。SQL‑OPERATION-TIME 以外の操作は測定するには期間が短すぎるため、SQL‑OPERATION-TIME カウンターは、DP2 プロセスビジー時間の主な消費要因である SQL 操作を特定することと、オープンする特定のプロセスに要求を関連付けることを目的としています。
- Guardian ファイルシステムでは、通常の Guardian ファイル名の指定によって、または OSS ファイルパス名（/G/*volume*/*subvol*/*filename*）によってファイルを開くことができます。
- DISCOPEN レコードを OSS ファイルシステムでオープンすると、ファイルのオープンに使用された OSS ファイルパス名が表示されます。DISCOPEN レコードを Guardian ファイルシステムでオープンすると、レコードの PATHID 情報は不完全で、情報を変換する要求を発行できません。この場合、

表示される OSS ファイルパス名は Guardian ファイル名と CRVSN 値を変換して取得されます。ファイル名に複数のパスが定義されている場合は、すべての OSS ファイルパス名が表示されます。

- LIST DISCOPEN *レポート要求または Guardian ファイルシステムからのレポート要求にはレコードの OSS ファイルパス名が表示されますが、その後の LIST 要求は同じ OSS ファイルパス名レコードを返しません。この動作は、ファイルシステムへのアクセスが Guardian ファイルシステムから行われたか、リストされるファイルが Guardian ファイル名であることを示しています。Guardian と OSS のファイルシステムと環境の違いについて詳しくは、*オープンシステムサービスユーザーガイド*を参照してください。
- Measure G11 以降の PVU では、ANSI SQL 名が表示されます。

例

DISCOPEN エンティティのレポート出力には、ANSI SQL 名を持つファイルについての ANSI SQL 名が含まれています。例を示します。

```
Disc File Open $DATA.ZSD12345.Z1234567
ANSI SQL Name 'TABLE Catalog_12.Schema_45.Table_56 PARTITION Partition_78'
Device Name $DATA                                  Pool
File Type Key Sequenced                            Local CPU  0
Opener Process: System \HURTS    CPU  0    Pin   285
Format Version:  H01  Data Version:  H01  Subsystem Version:  1
Local System \HURTS   From  12 Jan 2003, 19:35:28   For  46.9 Seconds
------------ Requests ------------------------------------------------------
Lockwait-Time                          Requests                         3 #
Max-Lockwait-Time                      Requests-Blocked
Lock-Timeouts                          Lock-Bounces
------------ Logical I/O ---------------------------------------------------
Cache-Hits                      6 #    Cache-Write-Hits
Block-Splits                           Cache-Write-Cleans
Driver-Input-Calls              6 #    Driver-Output-Calls
DBIO-Input-Calls                       DBIO-Output-Calls
------------ SQL -----------------------------------------------------------
SQL-Operation-Time
```

DISKFILE

DISKFILE エンティティは、指定されたディスクファイルのすべてのオープナープロセスによって実行された I/O 操作を測定します。DISKFILE は FCB を使用してオープナーを測定し、測定されるファイルごとに 1 つのカウンターレコードを作成します。DISKFILE エンティティタイプは、測定中にファイルがクローズされて再オープンしない限り、個々のオープナープロセスを追跡しません。

Measure G09 以降の PVU では、DISKFILE エンティティは、Guardian ファイル名に代わる OSS ファイルパス名の使用をサポートします。これは DISKFILE レポートで OSS ファイルパス名を表示し、外部の構造化レコードと新しい OSSNAMES 構造化レコード出力との直接マッピングを提供します。多くの OSS ファイルパス名が特定のディスクファイルを参照していますが、ファイルに DISKFILE レコードは 1 つしか存在しません。任意の有効な名前を使用すると、目的のレコードが取得されます。レコードのレポート機能では、レコードを参照するすべての有効な名前がリストされます。

Measure G11 以降の PVU では、ANSI SQL 名が表示されます。

Measure H01 以降の PVU では、MEASCOM は DISKFILE エンティティ指定の ANSI SQL 名を受け入れます。

トピック

DISKFILE エンティティのエンティティ指定構文

DISKFILE エンティティの DDL レコード

DISKFILE エンティティの DDL フィールドの定義

DISKFILE エンティティの使用上の注意

DISKFILE 測定の例

DISKFILE エンティティのエンティティ指定構文

DISKFILE エンティティを記述するには、次の構文を使用します。

```
DISKFILE entity-spec [,entity-spec] ...
```

DISKFILE

指定されたファイルへの物理アクセスおよび論理アクセスに関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                                                                    }
{ [[$device.]subvolume.]filename[:crvsn]                               }
{ [$device.]#tempfile                                                  }
{ "pathname"                                                           }
{ '{TABLE|INDEX} [[catalog.]schema.]object[ PARTITION partition]'      }
{ 'SCHEMA [catalog.]schema'                                            }
{ 'CATALOG {catalog|*}'                                                }
```

指定する値は次の通りです。

*****

すべてのディスクファイルのローカルおよびリモートアクセスを測定します。

$*device*

測定対象のファイルが含まれるボリューム（デバイス）の名前です。すべてのボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのボリュームです。

subvolume

測定対象のファイルが含まれるサブボリュームの名前です。すべてのサブボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのサブボリュームです。

filename

測定対象のファイルの名前です。一時ファイルを除くすべてのファイルを指定するには、アスタリスク（*）を使用します。一時ファイルは別途指定されます。

:*crvsn*

（Measure G09 以降の PVU）一意のファイル名を形成するために必要な、タイムスタンプ、作成バージョンのシリアル番号（CRVSN）、またはファイル名拡張子です。このオプションを使用して、ファイル名の一意性を保証します。CRVSN は Measure レポートおよび LISTGNAME コマンドから使用できます。

#tempfile

測定対象の一時ファイルのファイル識別番号です。すべての一時ファイルを指定するには、アスタリスク（*）を使用します。

"*pathname*"

（Measure G09 以降の PVU）完全修飾または部分修飾のいずれかの OSS ファイルパス名です。スラッシュ（/）で始まらない OSS ファイルパス名は、部分的なパス名と見なされ、OSSPATH の現在の設定を前に指定することで拡張されます。/G で始まる OSS ファイルパス名はサポートされていないため、代わりに Guardian ファイル名を使用してください。

注記:

OSS ファイルパス名は大文字と小文字が区別され、二重引用符（""）で囲んで指定する必要があります。有効な OSS ファイルパス名は、特定のファイル、または特定のディレクトリ内のファイルのセットを参照できます。ディレクトリが指定されている場合は、そのディレクトリ内のファイルのみが含まれます。指定したディレクトリの下位のディレクトリ内のファイルは含まれません。

catalog*、*schema*、*object*、*partition

（Measure H01 以降の PVU）ANSI SQL 識別子です。省略すると、*catalog* および *schema* フィールドは、MEASCOM の *catalog* および *schema* の環境値によって解決されます。名前空間キーワード TABLE または INDEX が省略された場合、デフォルトとして TABLE が推定されます。

ワイルドカードの構文は次のとおりです。

`TABLE [[catalogC.]schemaS.]tableT` – `tableT` 内のすべてのパーティション
`INDEX [[catalogC.]schemaS.]indexI` – `indexI` 内のすべてのインデックスパーティション
`SCHEMA [catalogC.]schemaS` – `schemaS` 内のすべてのオブジェクト
`CATALOG catalogC` – `catalogC` 内のすべてのオブジェクト
`CATALOG *` - ノード上で表示されるすべてのオブジェクト

DISKFILE エンティティの DDL レコード（ZMS 形式）

DISKFILE エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsdfile-id.
 02 FCB-number                type binary 16 unsigned.
 02 file-type                 type binary 16 unsigned.
 02 file-code                 type binary 16 unsigned.
 02 format                    type binary 16 unsigned.
 02 file-name.
    03 volume                 type character 8.
    03 subvol                 type character 8.
    03 filename               type character 8.
 02 device-name               type character 8.
 02 storage-pool              type character 8.
 02 file-name-MID.
```

```
    03 PATHID                  type character 24.
    03 CRVSN                   type character 6.

 02 reserved-1                 type character 10.

end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsdfile-ctrs.
 02 starting-eof               type binary 64.
 02 ending-eof                 type binary 64.
 02 transient-opens            type binary 64.
 02 open-qtime                 type binary 64.
 02 driver-input-calls         type binary 64.
 02 driver-output-calls        type binary 64.
 02 cache-read-hits            type binary 64.
 02 cache-write-hits           type binary 64.
 02 cache write-cleans         type binary 64.
 02 requests                   type binary 64.
 02 requests-blocked           type binary 64.
 02 lockwait-time              type binary 64.
 02 max-lockwait-time          type binary 64.
 02 lock-timeouts              type binary 64.
 02 lock-bounces               type binary 64.
 02 block-splits               type binary 64.
 02 extent-allocations         type binary 64.
 02 DBIO-input-calls           type binary 64.
 02 DBIO-output-calls          type binary 64.
 02 SQL-inserts                type binary 64.
 02 SQL-updates                type binary 64.
 02 SQL-deletes                type binary 64.
 02 starting-rows              type binary 64.
 02 ending-rows                type binary 64.
 02 OSS-Block-Writes           type binary 64.
 02 OSS-Block-Write-Bytes      type binary 64.
 02 OSS-Cache-Callbacks        type binary 64.
 02 OSS-Callback-Writes        type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsdfile. FILE is "zmsdfile" ENTRY-SEQUENCED.
 02 hdr                        type zmsheader.
 02 ctr                        type zmsdfile-ctrs.
 02 id                         type zmsdfile-id.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

DISKFILE エンティティの DDL レコード（Legacy 形式）

DISKFILE エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G シリーズ DISKFILE エンティティの DDL レコードは、カウンターがダイレクトバルク I/O および SQL アクションをサポートするために追加された点を除いて D シリーズ DISKFILE エンティティのレコード

と同じです。(DRIVER-INPUT-CALLS などの既存の I/O カウンターもダイレクトバルク I/O 操作を含みます。) さらに、OSS ファイルパス名のサポートのための新しい識別子が追加されました。

```
RECORD diskfile.  FILE is "diskfile" ENTRY-SEQUENCED.
                        .
                        .
                        .
```

(error、time items、および measurement identification items については、**一般的なエンティティヘッダーフィールド** (177 ページ) を参照してください)

```
                        .
                        .
                        .
* entity identification items:
 02 FCB-number                 type binary 16 unsigned.
 02 file-name.
    03 volume                   type character 8.
    03 subvol                   type character 8.
    03 filename                 type character 8.
 02 file-type                  type binary 16 unsigned.
 02 file-code                  type binary 16 unsigned.
* counter value items:
 02 starting-eof               type binary 64.
 02 ending-eof                 type binary 64.
 02 transient-opens            type binary 32 unsigned.
 02 open-qtime                 type binary 64.
 02 open-qlen-max              type binary 16 unsigned.
 02 driver-input-calls         type binary 32 unsigned.
 02 driver-output-calls        type binary 32 unsigned.
 02 cache-read-hits            type binary 32 unsigned.
 02 cache-write-hits           type binary 32 unsigned.
 02 cache write-cleans         type binary 32 unsigned.
 02 requests                   type binary 32 unsigned.
 02 requests-blocked           type binary 32 unsigned.
 02 lockwait-time              type binary 64.
 02 max-lockwait-time          type binary 64.
 02 lock-timeouts              type binary 32 unsigned.
 02 lock-bounces               type binary 32 unsigned.
 02 block-splits               type binary 32 unsigned.
 02 extent-allocations         type binary 32 unsigned.

* SMS changes:
 02 device-name                type character 8.
 02 storage-pool               type character 8.
* New counters in G05:
 02 dbio-input-calls           type binary 32 unsigned.
 02 dbio-output-calls          type binary 32 unsigned.
* New counters in G08:
 02 SQL-inserts                type binary 32 unsigned.
 02 SQL-updates                type binary 32 unsigned.
 02 SQL-deletes                type binary 32 unsigned.
 02 ending-rows                type binary 32 unsigned.
 02 format                     type binary 16 unsigned.
* New identifiers for OSS file pathname support:
 02 file-name-mid.
    03 pathid                   type character 24.
    03 crvsn                    type character  6.
* New counters in G10:
```

```
 02 OSS-Block-Writes           type binary 32 unsigned.
 02 OSS-Block-Write-Bytes      type binary 64.
 02 OSS-Cache-Callbacks        type binary 32 unsigned.
 02 OSS-Callback-Writes        type binary 32 unsigned.
end
```

DISKFILE エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

DEVICE-NAME

測定ファイルが配置されているディスクデバイス。SMF ファイルの場合、このフィールドには、論理ファイル名に対応する物理的な場所が表示されます。NonStop ファイルの場合、このフィールドは、FILE-NAME VOLUME サブフィールドと同じです。

FCB-NUMBER

測定されたファイルのファイル制御ブロック番号。ディスクプロセスの内部情報です。

FILE-CODE

3 桁のシステムファイルコード（オブジェクトファイルの場合は 100、編集ファイルの場合は 101 など）。ファイルコードの完全なリストについては、*File Utility Program (FUP) Reference Manual* を参照してください。

FILE-NAME

測定対象のファイルの名前。この名前は、NonStop ファイルか SMF ファイルのいずれかに適用できます。FILE-NAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、ディスクファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、ディスクファイルを開いたときに指定された物理ファイル名を表します。 SMF ファイルでは、ディスクファイルを開いたときに使用された、位置に依存しない論理ファイル名を表します。

FILE-NAME-MID

OSS ファイルまたは SQL/MX ファイルの内部形式表現。FILE-NAME-MID には次の 2 つのサブフィールドがあります。

PATHID	OSS ファイルまたは SQL/MX ファイルの内部形式表現。OSS または SQL/MX ファイル以外のファイルでは、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルまたは SQL/MX ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。その他のファイルでは、このフィールドにはゼロが含まれます。

FILE-TYPE

測定対象のファイルのタイプを示す以下のいずれかの値。

0	非構造化
1	相対値
2	エントリーシーケンス
3	キーシーケンス

STORAGE-POOL

測定対象のディスクが割り当てられている SMF ストレージプールの名前（ストレージプールプロセスのプロセス名）。

DISKFILE エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。

BLOCK-SPLITS

ファイルへの書き込み中のブロック分割の数。

カウンタータイプ：増分。

CACHE-READ-HITS

読み取り操作で必要なブロックがキャッシュ内に見つかり、ディスク I/O が節約された回数。DRIVER-INPUT-CALLS カウンターは、すべての読み取り操作（キャッシュヒットとキャッシュミスの両方）をカウントします。

カウンタータイプ：増分。

CACHE-WRITE-CLEANS

書き込み操作で必要なブロックがキャッシュ内に検出され、ブロックがクリーンだった回数。

書き込み中に、ディスクプロセスは要求されたブロックをキャッシュ内で検索します。キャッシュ内にブロックがある場合、ブロックはクリーンまたはダーティな状態です。ディスクプロセスは、キャッシュブロックをディスクにフラッシュすることによってキャッシュブロックを定期的にクリーンアップします。

カウンタータイプ：増分。

CACHE-WRITE-HITS

書き込み操作で必要なブロックがキャッシュ内に検出され、ブロックがダーティだった回数。

ダーティブロックに書き込むことで、書き込み I/O 操作が節約されます。

カウンタータイプ：増分。

DBIO-INPUT-CALLS

要求側のプロセッサーへのダイレクトバルク I/O 読み取り操作の数。このカウンターは DRIVER-INPUT-CALLS のサブセットです。

カウンタータイプ：増分。

DBIO-OUTPUT-CALLS

要求側のプロセッサーからのダイレクトバルク I/O 書き込み操作の数。このカウンターは DRIVER-OUTPUT-CALLS のサブセットです。

カウンタータイプ：増分。

DRIVER-INPUT-CALLS

キャッシュヒットとキャッシュミスの両方の読み取り操作の数。物理的な読み取り操作の数は、DRIVER-INPUT-CALLS と CACHE-READ-HITS の差です。

カウンタータイプ：増分。

DRIVER-OUTPUT-CALLS

書き込み操作数。書き込みカウントが 4096 バイト以下の非構造化ファイルへの書き込みおよび構造化ファイルの場合、物理ディスク書き込み操作の数は DRIVER-OUTPUT-CALLS と CACHE-WRITE-HITS の差です。

書き込みカウントが 4096 を超える（大規模な書き込み）非構造化ファイルへの書き込みの場合、物理ディスク書き込み操作の数は、DRIVER-OUTPUT-CALLS と、CACHE-WRITE-HITS および CACHE-WRITE-CLEANS の合計との差です。

カウンタータイプ：増分。

ENDING-EOF

測定停止時間または測定間隔コピー時間におけるファイルの EOF 値（バイト単位）。REPORT RATE 値は、MEASCOM のこの値の表示に影響しません。この項目はバイト単位の絶対値が表示された後に、STARTING-EOF に対する ENDING‑EOF の比率が表示されます。

カウンタータイプ：スナップショット。

ENDING-ROWS

測定間隔の終了時の SQL テーブルの行数。このカウンターは、ファイルのオープン以降のある時点でテーブルの行がゼロ行だった場合のみ有効です。

カウンタータイプ：スナップショット。

EXTENT-ALLOCATIONS

ファイルエクステントが割り当てられる回数です。

カウンタータイプ：増分。

LOCK-BOUNCES

ファイルシステムに返される返送ロック（FELOCKED = エラー 73）の数。SQL テーブルの場合、返送ロックは、RETURN‑IF‑LOCKED 制御オプションを付けて CONTROL TABLE コマンドを使用した結果です。Enscribe ファイルの場合、返送ロックは、Param.<15> = 1（代替ロックモード）で SETMODE 4 関数を使用した結果です。

カウンタータイプ：増分。

LOCK-TIMEOUTS

ロックでのタイムアウト（エラー 40）の数。要求のタイムアウト値に達すると、ファイルシステム（Enscribe および SQL/MP）は、ディスクプロセスにキャンセル要求を送信します。まだロックが承諾さ

れるまで待機している要求がディスクプロセスで検出されると、その要求はロック待機キューから外され、このカウンターが加算されます。

カウンタータイプ：増分。

LOCKWAIT-TIME

ロックの待機に費やした時間。

REQUESTS-BLOCKED カウンターは、ブロックされた要求の数、つまり要求がロックのキューに入れられた回数をカウントします。REPORT RATE 値がオンの場合、このカウンターは平均待機時間（LOCKWAIT-TIME を REQUESTS-BLOCKED で割った値）を返します。

カウンタータイプ：ロック待機。

MAX-LOCKWAIT-TIME

それぞれのロックの最大待機時間。アプリケーションでタイムアウト値を設定するガイドとしてこの数値を使用してください。REPORT RATE ON は、このカウンター値には影響しません。

カウンタータイプ：最大値。

OPEN-QTIME

このファイルが開いていたすべての時間（一時的なオープンを含む）。

カウンタータイプ：キュー。

OSS-BLOCK-WRITE-BYTES

OSS-BLOCK-WRITES の結果として DP2 によって書き込まれたバイトの数。

カウンタータイプ：累積。

OSS-BLOCK-WRITES

OSS ファイルシステムの DP2 によって発行されたブロック書き込み操作の数。

カウンタータイプ：増分。

OSS-CACHE-CALLBACKS

DP2 によって OSS ファイルマネージャーに送信されたキャッシュコールバック要求の数。

カウンタータイプ：増分。

OSS-CALLBACK-WRITES

OSS-CACHE-CALLBACKS の結果として DP2 によって受信された OSS-BLOCK-WRITES の数。

カウンタータイプ：増分。

REQUESTS

このファイルに対する要求の数。

カウンタータイプ：増分。

REQUESTS-BLOCKED

要求されたファイルまたはレコードがロックされていたために、I/O 操作が待機しなければならなかった回数。LOCKWAITS は、このカウンターの別の名前です。

カウンタータイプ：増分。

SQL-DELETES

SQL テーブルで実行された行削除操作の回数。

カウンタータイプ：増分。

SQL-INSERTS

SQL テーブルで実行された行挿入操作の回数。

カウンタータイプ：増分。

SQL-UPDATES

SQL テーブルで実行された行更新操作の回数。

カウンタータイプ：増分。

STARTING-EOF

測定開始時のファイルの EOF 値（バイト単位）。この項目は構造化 DISKFILE レコードの一部ですが、MEASCOM はこれを表示しません。代わりに、MEASCOM は、STARTING-EOF に対する ENDING‑EOF の比率を計算するための基準としてこのカウンター値を使用します。

カウンタータイプ：スナップショット。

STARTING-ROWS

（ZMS 形式のみ）測定間隔の開始時の SQL テーブルの行数。このカウンターは、ファイルのオープン以降のある時点でテーブルの行がゼロ行だった場合のみ有効です。

カウンタータイプ：スナップショット。

TRANSIENT-OPENS

測定期間中に発生したオープンの数。

カウンタータイプ：増分。

DISKFILE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

このカウンターフィールドは、Legacy 形式の DDL レコードのみで使用されます。

OPEN-QLEN-MAX

（Legacy 形式のみ）このファイルの最大オープン数。

カウンタータイプ：最大キュー。

すべての DISKFILE エンティティの使用上の注意

- DISCOPEN を使用する場合と DISKFILE を使用する場合の留意事項は次のとおりです。
  - DISCOPEN は、ファイルオープンごとに別々のカウンターレコードを作成します。オープナープロセスは個別に識別されます。同じプロセスが 1 つのファイルを複数回開くと、複数の DISCOPEN レコードが作成されます。
  - DISKFILE は、1 つのファイルのディスクファイルアクセスを全体として測定します。DISKFILE は、測定対象のファイルごとに 1 つのカウンターレコードを作成し、個々のオープナープロセスは追跡しません。
  - 代替キーファイルのオープンなどの間接オープンの場合、DISCOPEN は別々のレコードを作成します。DISKFILE は、間接オープンの場合は別々のレコードを作成しません。

- 測定中にファイルを閉じてから再度開いた場合、DISCOPEN と DISKFILE は両方とも新しいレコードを作成します。
- DISCOPEN は、ディスクファイルアクティビティの詳細なビューを提供します。DISCOPEN カウンターは、キーシーケンスデータベースファイルの構造に関する潜在的な問題を検出する場合に特に便利です。
- 高いレベルの詳細が不要である場合、DISKFILE では分析する必要があるデータ量を削減する一方で、ファイル構造と競合に関する潜在的な問題を検出することができます。

ファイルへのユーザーアクセスを測定するには、FILE エンティティタイプを使用します。説明については、**FILE**(284 ページ)を参照してください。

- ミラーディスクへの書き込みでは、物理的な書き込みが 2 回行われた場合でも、要求のカウントは 1 回増分されます。

G シリーズの DISKFILE エンティティの使用上の注意

- カウンター SQL-INSERTS、SQL-UPDATES、および SQL-DELETES は、DP2（T9053G08）および G06.11 以降の RVU で使用できるようになりました。カウンター SQL-ENDING-ROWS は、DP2（T9053G09）および G06.12 の RVU で使用できます。
- SQL-ENDING ROWS カウンターは、キューを表すために使用する SQL テーブルに関連があります。ファイルを最初に開いてから、テーブル内の行数がある時点でゼロになるまで、この値は表示されず、外部レコードには-1 として報告されます。行数がゼロに等しくなる（たとえば、キューが空になる）と、このカウンターは、各間隔の終了時にファイル内の行数を追跡します。
- Measure G08 以降の PVU では、DISKFILE エンティティのレポートは、ディスクファイルのファイルフォーマットが 1 と 2 のどちらであるかを識別します。フォーマット 2 のファイルは、以下の場合に使用されます。
  - パーティション分割されていないファイル（またはパーティション分割されたファイルのいずれかのパーティション）の最大サイズが 2 GB - 1 MB を超えた場合。
  - キーシーケンスファイル以外のパーティション分割されたファイルの最大サイズが 4 GB 以上の場合。
  - ファイルのエクステントサイズが 65535 ページを超えた場合。
  - 作成時にフォーマット 2 のファイルを明示的に要求された場合。
- Measure G09 以降の PVU では、DISKFILE エンティティは、次のように OSS ファイルのパス名の使用をサポートします。
  - DISKFILE で OSS ファイルパス名を使用すると、OSS PATHID のディレクトリ情報が無視されます。
  - 多くのアクセスが行われても、1 つのファイルに対して存在する DISKFILE レコードは 1 つだけです。
  - ファイルまで導く有効なパス名を使用することができます。
  - ディレクトリ名を使用して DISKFILE レコードのセットを示す方法は、サポートされていません。
  - OSS ファイルの一時的な DISKFILE レコードを測定する必要がある場合は、たとえば次のように、Guardian ファイル名のワイルドカード規則を使用して指定します。
    - $*.*.*
    - $*device*

      .*.*
    - *.ZYQ*nnnnn*.
  - MEASREADACTIVE によって返されるレコードには、OSS PATHID および CRVSN 情報は含まれません。特定の DISKFILE レコードに対応する OSS ファイルパス名を取得するには、MEASLISTPNAME を使用します。この変換によって、CRVSN が指定されていないという警告が生成されます。ただし、この変換はアクティブなレコードから値の変換であるため、この警告は無視できます。
- Measure G11 以降の PVU では、ANSI SQL 名が表示されます。

例

DISKFILE エンティティのレポート出力には、ANSI SQL 名を持つファイルについての ANSI SQL 名が含まれています。

```
Diskfile $DATA.ZSD12345.Z1234567               File Format x
ANSI SQL Name 'TABLE Catalog_12.Schema_45.Table_56 PARTITION Partition_78'
Device Name $DATA                                    Pool
Local CPU 0           File Type  Key Sequenced       File Code 550
Format Version:  H01  Data Version:  H01  Subsystem Version:  1
Local System \HURTS   From  12 Jan 2003, 19:35:28   For  46.9 Seconds
------------ File ------------------------------------------------------------
Open-Qtime                   72.11 ms   Transient-Opens
Starting-EOF                            Ending-EOF
Extent-Allocations                      File-Growth-Ratio
------------ Requests --------------------------------------------------------
Lockwait-Time                           Requests                          3 #
Max-Lockwait-Time                       Requests-Blocked
Lock-Timeouts                           Lock-Bounces
------------ Logical I/O -----------------------------------------------------
Cache-Read-Hits                2 #      Cache-Write-Hits
Block-Splits                            Cache-Write-Cleans
Driver-Input-Calls             2 #      Driver-Output-Calls
DBIO-Input-Calls                        DBIO-Output-Calls
------------ SQL -------------------------------------------------------------
SQL-Inserts
SQL-Updates                             SQL-Deletes
SQL-Starting-Rows                       SQL-Ending-Rows
------------ Open System Services --------------------------------------------
OSS-Cache-Callbacks                     OSS-Callback-Writes
OSS-Block-Writes                        OSS-Block-Write-Bytes
```

ENDPOINT

SERVERNET エンティティは ENDPOINT に置換されました。SERVERNET 名は引き続きコマンド行で受け付けられますが、将来の RVU での使用は非推奨になります。

H シリーズおよび J シリーズの RVU では、ENDPOINT エンティティは、ServerNet のクライアント、つまりメッセージシステム、ストレージサブシステム、および通信サブシステムに関係した物理 I/O 操作に関する情報を提供します。このカウンターにより、ユーザーはシステム全体のすべての ServerNet のアクティビティを監視できます。

トピック

ENDPOINT エンティティの DDL レコード（ZMS 形式）

ENDPOINT エンティティのエンティティ指定構文

ENDPOINT エンティティの DDL レコード（Legacy 形式）

ENDPOINT エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

ENDPOINT エンティティの使用上の注意

ENDPOINT エンティティのエンティティ指定構文

ENDPOINT エンティティを記述するには、次の構文を使用します。

```
[ ENDPOINT ] <endpt spec> [, <endpt spec> ] ...
[ SERVERNET ]
```

ENDPOINT

物理 I/O 操作に関する情報を提供します。

endpt spec

次のいずれかのように指定します。

- IPC ノードクラスの構文：

```
{ cpu [ ( IPC [, group [, module [, slot [, other-cpu ] ] ] ] ) ]
      [ ( type ) ] }
```

- RIPC ノードクラスの構文：

```
{ cpu [ ( RIPC [, group [, remote-node [, slot
      [, remote-cpu ] ] ] ] ) ] [ ( type ) ] }
```

- L シリーズの RVU の CLIM、CLMI、CLMO、CLMS ノードクラスの構文：

```
{ cpu [ ( node-class ) ] [ ( type ) ] }
{ cpu [, clim-name ] [ ( type ) ] }
```

- H および J シリーズシステムの CLIM、CLMI、CLMO、CLMS ノードクラスの構文：

```
{ cpu [ ( node-class [, group [, module [, slot
      [, port [, fiber ] ] ] ] ] ) ] }
{ cpu [, clim-name ] [ ( type ) ] }
```

指定する値は次の通りです。

*****

すべての CPU のすべてのデバイスを測定します。

cpu

CPU 番号です。0～15 の範囲の整数にする必要があります。デフォルトは、すべての CPU です。

node-class

測定対象の SAC のクラスです。CLIM、CLMI、CLMO、CLMS、COLO、ENET、IPC、MONT、NIOC、RIPC、SCSI、および SWAN。

リモートプロセッサー間通信（RIPC）は、リモートノード上の CPU との通信を記述します。

すべてのノードクラスを指定するには、アスタリスク（*）を使用します。*node-class* を指定しないと、すべてのノードクラスが測定されます。

L シリーズの RVU では、IPC、RIPC、および CLIM（CLMI、CLMS、および CLMO を含む）ノードクラスのみがサポートされます。

group

測定対象の SAC のグループ番号です。（グループは物理的なエンクロージャーに相当します）。すべてのグループを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのグループです。IPC または RIPC ノードクラスの場合は無視されます。L シリーズの RVU で作成される測定には使用されません。

module

測定対象の SAC のモジュール番号です。すべてのモジュールを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのモジュールです。IPC または RIPC ノードクラスの場合は無視されます。L シリーズの RVU で作成される測定には使用されません。

remote-node

リモートノード番号です。リモートプロセッサー間通信（RIPC）に対してのみ有効です。すべてのノードを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのノードです。

slot

測定対象の SAC のスロット番号です。すべてのスロットを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのスロットです。IPC または RIPC ノードクラスの場合は無視されます。L シリーズの RVU で作成される測定には使用されません。

port

CLIM が接続されている ServerNet PIC 内（または HPE NonStop BladeSystem 上の ServerNet ブレードスイッチ内）の特定の ServerNet コネクタを識別します。NonStop BladeSystem の場合のみ、有効なポートの値は 3～8、または"*"です。他のすべてのシステムでは、ポートの有効な値は 1～4、または"*"です。デフォルトは、すべてのポートです。L シリーズの RVU で作成される測定には使用されません。

fiber

CLIM が接続されている ServerNet ケーブル内の特定のコネクタを識別します。有効なファイバーの値は、1～4 です。ファイバーは NonStop BladeSystem にのみ関連します。デフォルトは、すべてのファイバー接続です。L シリーズの RVU で作成される測定には使用されません。

other-cpu

エンティティと通信中のローカル CPU の番号です。IPC に対してのみ有効です。デフォルトは、すべての CPU です。

remote-cpu

エンティティと通信中のリモートノードクラスター上の CPU の番号です。RIPC に対してのみ有効です。デフォルトは、すべての CPU です。

type

ServerNet アドレス指定可能コントローラーのクラスの部品番号、または CLMS のタイプです。

注記:

type 指定は、ストレージ CLIM（CLMS）の場合を除き、CLIM ノードクラスには使用されません。

- ストレージサブシステムから報告される CLMS アクティビティの場合、タイプは 2 です。
- CIP サブシステムから報告される CLMS アクティビティの場合、タイプは 3 です。

clim-name

エンティティと通信中の CLIM の名前です。

CLIM ノードクラスに対してのみ有効です。デフォルトは、すべての CLIM です。

ENDPOINT エンティティの DDL レコード（ZMS 形式）

ENDPOINT エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmssvnet-id.
 02 node-class                  type binary 32 unsigned.
 02 node-class-s                type character 4
                                redefines node-class.
 02 ctrl-type                   type binary 16 unsigned.
 02 flags                       type binary 16 unsigned.
 02 reserved-1                  type character 4
 02 remote-cluster              type binary 32 unsigned
 02 reserved-2                  type character 4
 02 remote-cpu                  type binary signed 32
 02 adapter-name                type character 64.
 02 SAC-name                    type character 64.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsendpt-ctrs.
 02 requests                    type binary 64.
 02 total-io-bytes              type binary 64.
 02 io-qtime                    type binary 64.
 02 io-qbusy-time               type binary 64.
 02 read-requests               type binary 64.
 02 write-requests              type binary 64.
 02 read-bytes                  type binary 64.
 02 read-cbytes                 type binary 64.
 02 read-qtime                  type binary 64.
 02 read-qbusy-time             type binary 64.
 02 write-bytes                 type binary 64.
 02 write-cbytes                type binary 64.
 02 write-qtime                 type binary 64.
 02 write-qbusy-time            type binary 64.
 02 server-qtime                type binary 64.
 02 retries                     type binary 64.
 02 acks                        type binary 64.
 02 x-defrd-busy-time           type binary 64.
 02 y-defrd-busy-time           type binary 64.
 02 read-crequests              type binary 64.
 02 write-crequests             type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsendpt.  FILE is "zmsendpt" ENTRY-SEQUENCED.
 02 hdr                         type zmsheader.
 02 ctr                         type zmsendpt-ctrs.
 02 id                          type zmsendpt-id.
end
```

ENDPOINT エンティティの DDL レコード（Legacy 形式）

ENDPOINT エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD endpoint.  FILE is "endpoint" ENTRY-SEQUENCED.
 02 error                      type binary 16 unsigned
 ...
```

```
 end
```
(error、time items、および measurement identification items については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください)

```
                        .
                        .
                        .
* entity identification items:
 02 channel                        type binary 16 unsigned.
 02 ctrl                           type binary 16 unsigned.
 02 ctrl-type                      type binary 16 unsigned.
* counter value items:
 02 requests                       type binary 32 unsigned.
 02 total-io-bytes                 type binary 64.
 02 io-qtime                       type binary 64.
 02 io-qlen-max                    type binary 16 unsigned.
* F40 new entity identification items:
 02 adapter-name                   type character 64.
 02 SAC-name                       type character 64.
 02 node-class                     type binary 32 unsigned.
 02 node-class-s                   type character 4
                                   redefines node-class.
 02 GMS.
    03 group                       type binary 32 unsigned.
    03 module                      type binary 32 unsigned.
    03 remote-cluster              type binary 32 unsigned
                                   redefines module.
    03 slot                        type binary 32 unsigned.
 02 subdevice                      type binary 32 unsigned.
 02 remote-CPU                     type binary 32 unsigned
                                   redefines subdevice.
* F40 new counter value items:
 02 io-qbusy-time                  type binary 64.
* G01 counter value items:
 02 read-requests                  type binary 32 unsigned.
 02 read-bytes                     type binary 64.
 02 read-qtime                     type binary 64.
 02 read-qlen-max                  type binary 16 unsigned.
 02 read-qbusy-time                type binary 64.
 02 write-requests                 type binary 32 unsigned.
 02 write-bytes                    type binary 64.
 02 write-qtime                    type binary 64.
 02 write-qlen-max                 type binary 16 unsigned.
 02 write-qbusy-time               type binary 64.

* G08 new counters:
 02 read-cbytes                    type binary 64.
 02 write-cbytes                   type binary 64.
 02 server-qtime                   type binary 64.
 02 server-qlen-max                type binary 16 unsigned.
 02 retries                        type binary 32.
 02 acks                           type binary 32.
 02 x-defrd-busy-time              type binary 64.
 02 y-defrd-busy-time              type binary 64.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

ENDPOINT エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

ADAPTER-NAME

SAC が存在するアダプターに関連付けられた論理名です。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

CTRL-TYPE

3128 などの製品番号。

IPC および RIPC レコードでの CTRL-TYPE は、リンカーの場合は 0、リスナーの場合は 1 です。IPC レコードについて詳しくは、**ENDPOINT エンティティの使用上の注意**(277 ページ)を参照してください。

FLAGS

（ZMS 形式のみ、L01、H07/J05 以降の PVU）測定が L シリーズ以降の RVU で行われたかどうかを示すために使用される 16 ビットフィールド。

flags.0:13	今後使用するために予約されています。
flags.14	1 = 測定は L シリーズ以降の RVU で行われました
flags.15	今後使用するために予約されています。

NODE-CLASS

システム構成データベースで定義されている SAC タイプ。

NODE-CLASS-S

NODE-CLASS を 4 文字の英数字フィールドとして再定義します。指定可能な値（空白を埋める 4 文字）は、次のとおりです。

- “CLIM”
- “CLMI”
- “CLMO”
- “CLMS”
- "COLO"
- "ENET"
- "IPC "
- "MONT"
- "NIOC"
- "RIPC"
- "SCSI"
- "SWAN"
- "* "

L シリーズの RVU でサポートされるノードクラスは、CLIM、CLMI、CLMO、CLMS、IPC、および RIPC です。

注記:

CLIM は、すべての CLIM ノードクラスのワイルドカードです。

REMOTE-CLUSTER

エンティティと通信中のリモートクラスターの数。RIPC に対してのみ有効です。

REMOTE-CPU

RIPC では、エンティティと通信中のリモートクラスター上の CPU の番号。

IPC では、エンティティと通信中のローカルクラスター上のもう一方の CPU の番号。

SAC-NAME

物理ディスクへのこのパスで使用される SAC の論理名。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

IPC レコードの場合、Measure は SAC 名の代わりにこの識別子を生成します。

`IPC-CPU`*`xx`*`-TO-CPU`*`yy`*

指定する値は次の通りです。*xx* はリンカー CPU 番号で、*yy* はリスナー CPU 番号です。

ENDPOINT エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。

SERVER-QTIME

このプロセッサーでのメッセージの受信から返信の発行までの時間（ミリ秒単位）。この値には、CPU にデータをプルする Measure システム読み取り操作の READ-QTIME は含まれません。このカウンターは、受信プロセスが要求に応答するまでに必要な時間から、Measure システムのレイテンシ時間を切り離すときに役立ちます。これは、IPC および RIPC レコードにのみ適用されます。

カウンタータイプ：累積。

X-DEFRD-BUSY-TIME

メッセージシステムが、Y ファブリックの使用から X ファブリックの使用に切り替えられなかった時間（マイクロ秒単位）。このカウンターの値が大きい場合、Y ファブリックに未処理のメッセージが生じていたり、X ファブリックに問題が起きていたりする可能性があります。このカウンターは、IPC および RIPC レコードにのみ適用されます。

カウンタータイプ：ビジー。

Y-DEFRD-BUSY-TIME

Measure システムが、X ファブリックの使用から Y ファブリックの使用に切り替えられなかった時間（マイクロ秒単位）。このカウンターの値が大きい場合、X ファブリックに未処理のメッセージが生じていたり、Y ファブリックに問題が起きていたりする可能性があります。このカウンターは、IPC および RIPC レコードにのみ適用されます。

カウンタータイプ：ビジー。

ENDPOINT エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードのみで使用されます。

フィールドはアルファベット順に一覧表示されます。

WRITE-QLEN-MAX

（Legacy 形式のみ）カウンターが割り当てられた後に書き込み要求キューにある要求の最大数。

カウンタータイプ：最大キュー。

ENDPOINT エンティティの使用上の注意

- ENDPOINT エンティティは、システム全体のすべての ENDPOINT 操作を表示します。たとえば、複数のデバイスが CLIM を共有している場合、ENDPOINT エンティティは、CLIM を使用するすべてのデバイスの複合的なアクティビティを測定します。
- ZMSENDPT レコードの SUBSYSTEM-VERSION は特定のサブシステムによって提供され、レコードの node^class で表されます。
  - CLIM ノードクラスの場合、CIP（クラスター I/O プロトコル）サブシステム
  - IPC および RIPC の場合、メッセージシステム
  - SCSI および COLO の場合、SCSI モジュールドライバー
  - NIOC の場合、SLSA モジュールドライバー
- 単一のデバイスが 1 つのコントローラーに接続されている場合、ENDPOINT エンティティとデバイスエンティティタイプ（DEVICE、DISK など）との間に重複が一部存在しています。
- MFIOB デバイスなどのストレージ SAC の場合、各要求のビジー時間とサービス時間を最も正確に示すインジケーターは IO-QBUSY-TIME カウンターです。READ-QBUSY-TIME と WRITE-QBUSY-TIME のカウンターは同様のデータを提供しますが、これら 2 つのカウンターは、要求がキューに入れられるため、ENDPOINT 環境では多少正確さが低下します。

ENDPOINT IPC および RIPC の使用上の注意

- プロセッサー間通信（IPC）アクティビティは、一部が開始側プロセッサー（リンカーと呼ばれます）で測定され、一部が受信側または応答側プロセッサー（リスナーと呼ばれます）で測定されます。各プロセッサーでは、1 つの ENDPOINT レコードが別のプロセッサーとのリンカーアクティビティを追跡し、2 番目の ENDPOINT レコードがリスナーアクティビティを追跡します。Measure パフォーマンスモニターは、IPC ENDPOINT レコードの SAC 名フィールドに次の識別子を生成します。

  ```
  IPC-CPUxx-TO-CPUyy
  ```

  指定する値は次の通りです。*xx* はリンカー CPU 番号で、*yy* はリスナー CPU 番号です。
- クラスター接続をサポートするために、リンカーおよびリスナーのレコードデータは、リンカーレコード（開始側プロセッサー）がメッセージトラフィックを完全にカウントできるように変更されました。この機能は、リンカーとリスナーの両方のレコードが、同じ数のメッセージ交換と対応するバイト数をカウントするように保証することで実現されました。このように変更した結果、クラスター接続のリスナーレコードが別のノードに置かれるようになり、Measure データファイルを簡単に利用できるのが 1 つのノードだけではなくなったため、リンカーレコードからのカウントがより適切になりました。
- リスナーの READ-BYTES は、対応するリンカーレコード WRITE-BYTES を超える場合があります。この状態は、メッセージシステムの MQC 割り当てが頻繁に失敗し、これによりメッセージデータの再転送が発生した場合に生じます。
- 事前プッシュ（非クラスター）メッセージの場合、READ-REQUESTS の数はリスナーレコードでは加算されません。リスナーレコードの READ-REQUESTS の数は、実行されたメッセージシステムプル操作の数を反映し、WRITE-REQUESTS カウンターのサブセットになります。READS と

READ-BUSY-QTIME を比較すると、プル操作によるメッセージシステムの正確なレイテンシが得られます。

- リモートノードメッセージ（メッセージを事前プッシュしない）では、リモートプロセッサーまたはリスナーが WRITE-REQUEST に応じようとし、そのメッセージが大規模であった場合に、READ-REQUESTS の数が REQUESTS の数の 2 倍になることがあります。readlinkcache サイズよりも大きなメッセージの場合（たとえば 2048 バイトより大きいメッセージ）、リスナープロセスは 2 回のプル操作（制御用のプルとデータ用のプル）を実行します。
- メッセージシステムのリスナーレコードの SERVER-QTIME カウンターは、リスナープロセッサーによるメッセージの受信からその後の応答までの時間を測定します。関連付けられているリンカープロセッサーレコードの WRITE-QTIME カウンターと組み合わせて使用した場合、このカウンターから、メッセージシステムのメッセージ転送に関するレイテンシが得られます。平均メッセージレイテンシは、リンカーレコードの WRITE-QTIME からリスナーレコードの SERVER-QTIME を引いた後、リンカーレコード WRITE-REQUESTS で割って算出できます。

  平均メッセージレイテンシの計算：

  ```
  WRITE-QTIME(リンカー) - SERVER-QTIME(リスナー)/WRITE-REQEUSTS(リンカー)
  ```

- Measure は、システム間で測定間隔を調整しません。別々のクラスターにあるプロセッサー間の平均メッセージレイテンシを計算するには、同様の間隔で SERVER-QTIME および WRITE-QTIME の平均値を計算してから、結果の値を減算します。
- NonStop S70000 サーバー（NSR-G プロセッサー）への接続では、RETRIES カウンターは高い値になると予想されます。これらのプロセッサーの ServerNet インターフェイス（MITE）の不具合は、ソフトウェアのリカバリによって処理されます。これらの問題は NonStop S72000 サーバー（NSR-T プロセッサー）では修正されていますが、NonStop S70000 サーバー（NSR-G プロセッサー）と通信するときには、S72000 サーバーは依然として高い RETRIES の数を報告することがあります。
- 場合によっては、リモートプロセッサーまたはリスナーが WRITE-REQUEST に応じようとし、メッセージが非常に大規模である場合に、READ-REQUESTS の数が REQUESTS の数の 2 倍になることがあります。readlinkcache サイズよりも大きなメッセージの場合（たとえば 2048 バイトより大きいメッセージ）、リスナープロセスは 2 回のプル操作（制御用のプルとデータ用のプル）を実行します。

CLIM の使用上の注意

- Measure H03/J01 以降の PVU で Measure の CLIM *node-class* を指定する場合、IP CLIM には CLMI を、ストレージ CLIM には CLMS がサポートされます。これらのノードクラスの場合、*node-class* の後の 4 番目に指定するオプションは *port* です。CLIM 以外のノードクラスの場合、これは *subdevice* です。

  Measure H04/J02 以降の PVU では、Telco CLIM の CLIM *node-class* を指定する場合に CLMO がサポートされます。

  CLIM は、すべての CLIM ノードクラスのワイルドカードです。

  NonStop BladeSystem および L シリーズの RVU を除き、各 CLIM はグループ-モジュール-スロット-ポートで一意に識別されます。つまり、CLIM が接続されている P-Switch の物理的な場所とポートの観点から CLIM を識別します。グループ（システムエンクロージャー）とモジュール（グループのサブセット）によって P-Switch が識別されます。スロット（モジュール内のラベル付きの物理的な領域）によって、その P-Switch 内の特定の ServerNet PIC とポートが識別されます。ポートによって、CLIM が接続されている ServerNet PIC 内で特定の ServerNet コネクタが識別されます。

  NonStop BladeSystem では、CLIM を識別するために追加のファイバー番号が必要です。

  L シリーズの RVU では、CLIM は CLIM 名でのみ識別されます。
- 各ストレージ CLIM のアクティビティは、2 つの ENDPOINT レコードで報告されます。一方のレコードは、ストレージサブシステムからのアクティビティを報告し、もう一方のレコードは、CIP サブシステムからのアクティビティを報告します。

  ストレージサブシステムレコードは、CLIM でサポートされているすべてのデバイスの I/O アクティビティ（READ-BYTES、WRITE-BYTES、READ-REQUESTS、WRITE-REQUESTS、およびさまざまなキューカウンター）を報告します。IO 制御アクティビティ（READ-CBYTES、WRITE-CBYTES、READ-CREQUESTS、および WRITE-CREQUESTS カウンター）は一切報告しません。

CIP サブシステムは、CLIM で実行される IO 制御アクティビティを報告しますが、IO アクティビティは報告しません。通常、IO 制御アクティビティは、CIP サブシステムと CLIM の間の低頻度の ping に過ぎません。

以下に、ストレージ CLIM の 2 つのレポートレコードの例を示します。最初の例は CIP サブシステムの例で、2 番目の例はストレージサブシステムの例です。

CIP サブシステムの例：

```
SAC   C1002531
Adpt  C1002531
CPU   6    Node Class  CLMS   GMS  100,2,5     Port  3   Fiber  1
Format Version:  H04  Data Version:  H04  Subsystem Version:  1
Local System \BLNSK2  From  26 Mar 2009, 17:13:10   For  33.6 Seconds
------------------------------------------------------------------------
-
Requests                         12 #    Total-IO-Bytes                672 #
IO-Qtime                                 IO-Qbusy-Time
--------- Data into this CPU --------  --------- Data out from this CPU
----
Read-Requests                            Write-Requests
Read-Bytes                               Write-Bytes
Read-Cbytes                     336 #    Write-Cbytes                  336 #
Read-Qtime                               Write-Qtime
Read-Qbusy-Time                          Write-Qbusy-Time
Read-Crequests                    6 #    Write-Crequests                 6 #
------------------------------------------------------------------------
-
Server-Qtime
Retries                                  Acks
X-Defrd-Busy-Time                        Y-Defrd-Busy-Time
```

ストレージサブシステムの例：

```
SAC   C1002531
Adpt  C1002531
CPU   6    Node Class  CLMS   GMS  100,2,5     Port  3   Fiber  1
Format Version:  H04  Data Version:  H04  Subsystem Version:  1
Local System \BLNSK2  From  26 Mar 2009, 17:13:10   For  33.6 Seconds
------------------------------------------------------------------------
-
Requests                     2,520 #    Total-IO-Bytes         82,281,472 #
IO-Qtime                      1.18 sec  IO-Qbusy-Time                1.17
sec
--------- Data into this CPU --------  --------- Data out from this CPU
----
Read-Requests                2,514 #    Write-Requests                  6 #
Read-Bytes              82,231,808 #    Write-Bytes                49,664 #
Read-Cbytes                              Write-Cbytes
Read-Qtime                    1.14 sec  Write-Qtime                 36.71 ms
Read-Qbusy-Time               1.14 sec  Write-Qbusy-Time            36.71 ms
Read-Crequests                           Write-Crequests
------------------------------------------------------------------------
-
Server-Qtime
Retries                                  Acks
X-Defrd-Busy-Time                        Y-Defrd-Busy-Time
```

ENDPOINT クラスターの測定の構成の例

ADD、DELETE、LIST、LISTACTIVE、および LISTALL コマンドは、1 つまたは複数の ENDPOINT エンティティを参照できます。この例では ADD コマンドが使用されていますが、DELETE、LIST、LISTACTIVE、および LISTALL コマンドはすべて、リモートクラスターアクティビティについての測定を構成または表示するために同じコマンド構文を共有します。

コマンド	以下の範囲のすべてのリモート IPC アクティビティを測定...
ADD ENDPOINT * (RIPC)	ローカルシステム上のすべての CPU とすべてのリモートクラスター上のすべての CPU
ADD ENDPOINT * (RIPC,*,4)	ローカルシステム上のすべての CPU とリモートノード番号 4 上のすべての CPU
ADD ENDPOINT * (RIPC,*,4,*,5)	ローカルシステム上のすべての CPU とリモートノード番号 4 上の CPU 5
ADD ENDPOINT * (RIPC, *, 4)	ローカルシステム上の CPU 1 とリモートノード番号 4 上のすべての CPU
ADD ENDPOINT 1 (RIPC, *, 4, *, 0)	ローカルシステム上の CPU 1 とリモートノード番号 4 上の CPU 0

CLIM の測定の構成の例

ADD、DELETE、LIST、LISTACTIVE、および LISTALL コマンドは、1 つまたは複数の ENDPOINT エンティティを参照できます。この例では ADD コマンドが使用されていますが、DELETE、LIST、LISTACTIVE、および LISTALL コマンドはすべて、CLIM の測定を構成または表示するために同じコマンド構文を共有します。

コマンド	測定対象...
ADD ENDPOINT *(CLIM)	すべての CPU 上のすべての CLIM
ADD ENDPOINT 1 (CLIM)	CPU 1 のすべての CLIM
ADD ENDPOINT * (CLMS, 1,2,3)	すべての CPU で GMS が 1,2,3 に等しいすべてのストレージ CLIM 上のすべてのポート（H または J シリーズの RVU）
ADD ENDPOINT * (CLMI, 1,2,3,1)	すべての CPU で GMS が 1,2,3 に等しいすべての IP CLIM 上のポート 1（H または J シリーズの RVU）
ADD ENDPOINT * (CLMI, 1,2,3,8,1)	すべての CPU で GMS が 1,2,3 に等しいすべての IP CLIM 上のポート 8、ファイバー 1（H または J シリーズの RVU）
ADD ENDPOINT 0, SCLIM000	L シリーズの RVU の CPU 0 で SCLIM000 という名前の CLIM の CLIM エンティティ
ADD ENDPOINT *, NCLIM001	L シリーズの RVU のすべての CPU で NCLIM001 という名前の CLIM のすべての CLIM エンティティ
ADD ENDPOINT * (CLMO)	すべての CPU 上のすべての Telco CLIM

FABRIC

FABRIC エンティティにより、ユーザーは CPU とクラスター全体の他のエンドポイントとの間のすべてのシステム相互接続アクティビティを監視できます。

FABRIC エンティティは、ENDPOINT エンティティよりも低いレベルで細分性を高めてシステム相互接続 I/O アクティビティを計測します。 たとえば、IPC ノードクラスの ENDPOINT エンティティは現在の CPU とターゲット CPU の間のアクティビティを計測します。2 つの CPU 間または CPU と CLIM の間の I/O アクティビティを計測する多くの FABRIC エンティティレコード（それぞれが仮想チャネルを表す）が存在する可能性があります。

トピック

FABRIC エンティティのエンティティ指定構文

FABRIC エンティティの DDL レコード

FABRIC エンティティの DDL フィールドの定義

FABRIC エンティティのカウンターフィールドの定義

FABRIC エンティティの使用上の注意

FABRIC エンティティのエンティティ指定構文

FABRIC エンティティを記述するには、次の構文を使用します。

```
FABRIC entity-spec [ , entity-spec ] ...
```

FABRIC

特定の仮想チャネル上のエンドポイント間の物理 I/O 操作の詳細情報を提供します。

entity-spec

次のいずれかになります。

```
{ cpu ( * )                                              }
{ cpu ( CPU     [, system-spec [, virt-chan ] ] ) }
{ cpu ( STORAGE [, clim-name   [, virt-chan ] ] ) }
{ cpu ( IP      [, clim-name   [, virt-chan ] ] ) }
{ cpu ( OPEN    [, clim-name   [, virt-chan ] ] ) }
```

system-spec

次のいずれかになります。

```
{ * }
{ { \sys-name | \sys-number } [ ( other-cpu ) ] }
```

sys-name

リモートシステムのシステム名です。

sys-number

リモートシステムのシステム番号です（0～254 の範囲）。

cpu

測定対象のデバイスが構成されている CPU の番号です（0～15 の範囲）。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

other-cpu

他の CPU の CPU 番号です（0～15 の範囲）。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

clim-name

CLIM の名前です。すべての CLIM を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CLIM です。

virt-chan

仮想チャネルの名前です。すべての仮想チャネルを指定するには、アスタリスク（*）を使用します。デフォルトは、すべての仮想チャネルです。

FABRIC エンティティの DDL レコード

（MEASDDLS で定義されている）ZMSFABR レコード形式を以下に示します。

BRIEF レポートに含まれるフィールドは、太字で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsfabr-id.
 02 endpoint-type                type binary 16 unsigned.
 02 target-cpu                   type binary 16 unsigned.
 02 target-node                  type binary 16 unsigned.
 02 reserved—1                   type binary 16 unsigned.
 02 target-node-name             type character 8.
 02 virtual-chan-name            type character 8.
 02 endpoint-name                type character 8.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsfabr-ctrs.
 02 x-recvs                      type binary 64.
 02 x-recv-bytes                 type binary 64.
 02 x-sends                      type binary 64.
 02 x-send-bytes                 type binary 64.
 02 x-send-qlen-start            type binary 64.
 02 x-send-qlen-end              type binary 64.
 02 x-reserved-1                 type character 48.
 02 y-recvs                      type binary 64.
 02 y-recv-bytes                 type binary 64.
 02 y-sends                      type binary 64.
 02 y-send-bytes                 type binary 64.
 02 y-send-qlen-start            type binary 64.
 02 y-send-qlen-end              type binary 64.
 02 y-reserved-2                 type character 48.
end
```

DDL レコード記述フィールド

```
RECORD zmsfabr. FILE is "zmsfabr" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 ctr                          type zmsfabr-ctrs.
 02 id                           type zmsfabr-id.
end
```

FABRIC エンティティの ID フィールドの DDL 定義

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

ENDPOINT-NAME

エンドポイント名（例：SCLIM000）を含む 8 バイトフィールドです。この名前は null で埋められ、このフィールドには CPU エンドポイントの null 値が含まれます。

ENDPOINT-TYPE

エンドポイントタイプの列挙値です。1 は CPU（ローカルまたはリモート）、2 はストレージ CLIM、3 は IP CLIM、4 は Telco CLIM です。これらの値のリテラルは、ファイル MEASDECS にあります。

TARGET-CPU

これは、エンドポイントタイプが CPU の場合はターゲット CPU の CPU 番号です。他のすべてのエンドポイントタイプでは、このフィールドは 0 です。

TARGET-NODE

CPU エンドポイントの場合、G、H、J シリーズの RVU では ServerNet ノード番号、L シリーズの RVU では Expand ノード番号です。他のエンドポイントタイプでは、このフィールドは 0 です。

TARGET-NODE-NAME

これは、ターゲットノード名（例：\BLPROD）が含まれている 8 バイトのフィールドです。このフィールドは、リモート CPU の CPU エンドポイントの場合のみ入力されます。この名前はスペースで埋められます。

VIRTUAL-CHAN-NAME

これは、仮想チャネル名を含む 8 バイトのフィールドです。この名前は null で埋められます。

FABRIC エンティティのカウンターフィールドの DDL 定義

フィールドはアルファベット順に一覧表示されます。

X-RECVS

X ファブリックでエンドポイントから受信した転送の数。

カウンタータイプ：増分。

X-RECV-BYTES

X ファブリックでエンドポイントから受信したバイト数。

カウンタータイプ：累積。

X-SENDS

X ファブリックでエンドポイントに送信した転送の数。

カウンタータイプ：増分。

X-SEND-BYTES

X ファブリックでエンドポイントに送信したバイト数。

カウンタータイプ：累積。

X-SEND-QLEN-END

測定間隔の終了時に X ファブリックの送信キューにあったデータ転送操作の数。

カウンタータイプ：スナップショット。

X-SEND-QLEN-START

測定間隔の開始時に X ファブリックの送信キューにあったデータ転送操作の数。

カウンタータイプ：スナップショット。

Y-RECVS

Y ファブリックでエンドポイントから受信した転送の数。

カウンタータイプ：増分。

Y-RECV-BYTES

Y ファブリックでエンドポイントから受信したバイト数。

カウンタータイプ：累積。

Y-SENDS

Y ファブリックでエンドポイントに送信した転送の数。

カウンタータイプ：増分。

Y-SEND-BYTES

Y ファブリックでエンドポイントに送信したバイト数。

カウンタータイプ：累積。

Y-SEND-QLEN-END

測定間隔の終了時に Y ファブリックの送信キューにあったデータ転送操作の数。

カウンタータイプ：スナップショット。

Y-SEND-QLEN-START

測定間隔の開始時に Y ファブリックの送信キューにあったデータ転送操作の数。

カウンタータイプ：スナップショット。

FABRIC エンティティの使用上の注意

現時点ではありません。

FILE

FILE エンティティは、明示的に開かれたファイルにユーザープロセスによって実行される I/O 操作を測定します（論理的なファイルアクセス）。

Measure G09 以降の PVU では、FILE エンティティは、Guardian ファイル名に代わる OSS ファイルパス名の使用をサポートします。また、これは FILE レポートで OSS ファイルパス名を表示し、外部の構造化レコードと OSSNAMES 構造化レコード出力との直接マッピングを提供します。OSS ファイルパス名

は、開かれるファイルの名前、またはファイルを開くプロセスのプログラムファイル名を指定するために使用できます。

Measure G11 以降の PVU では、FILE エンティティは FILE レポートで ANSI SQL 名を表示します。

Measure H01 以降の PVU では、MEASCOM は FILE エンティティ指定の ANSI SQL 名を受け入れます。

トピック
Measure と OSS ファイルのオープン
OSS の命名規則
ANSI SQL の命名規則
FILE エンティティのエンティティ指定構文
FILE エンティティの DDL レコード
FILE エンティティの DDL フィールドの定義
FILE エンティティの使用上の注意
コマンドの例：OSS ファイルのオープン

Measure と OSS ファイルのオープン

FILE エンティティには、OSS が使用できる多数の FILE リレーションシップが含まれています。OSS FILE オープンのタイプは次のとおりです。

OSS ファイルタイプ	説明
OSS の通常のファイル	OSS 階層的名前空間に名前がある OSS ディスクファイル
OSS パイプ	プロセス間の通信のための IPC メカニズム
OSS FIFO	名前付きパイプ（OSS 名前空間で使用可能な名前付き）
OSS ソケット	OSS AF_INET または OSS AF_UNIX ソケットを使用する IPC メカニズム

FILE エンティティは、OSS ディレクトリ、TTY デバイス、および/dev/null の OSS のオープンは記録しません。これらのオブジェクトに使用される集計カウンターについて詳しくは、**PROCESS**(358 ページ)を参照してください。OSS ファイルインストゥルメンテーションは、通常はライブラリ呼び出しのレベルではなく、システムコールのレベルです。既存の Guardian TCP/IP ソケットのインストゥルメンテーションは変更されません。プロセスファイルのオープンは記録されますが、使用できるソケットレベルの情報はありません。

FILE エンティティの OSS ファイルオープンのサポートと Guardian ファイルオープンのサポートの 2 つの重要な違いは、次のとおりです。

- カウンターレコードが作成されるタイミング
- OSS ファイルオープンオブジェクトを識別するための命名規則

レコードの作成

OSS ファイルは、多くの場合は親プロセスの環境内で開かれますが、オープンを継承する子プロセスのコンテキストでのみアクセスされます。同様に、子プロセスは、実際には親プロセスから継承されたすべてのファイルオープンを使用しない可能性があります。Measure での OSS ファイルオープンの追跡によるシステムへの影響を最小限に抑えるため、OSS ファイルオープンのカウンターレコードは、ファイルのオープン時ではなく、ファイルが初めて大幅に使用された時点で開始されます。以下の OSS API は、大幅な使用と見なされます。

accept()	open(O_TRUNC)	send()
fcntl(F_GETLK,F_SETLK,F_SETLKW)	read()	sendmsg()
fsync()	recv()	sendto()
ftruncate()	recvfrom()	write()
lseek()	recvmsg()	

OSS パイプまたは OSS FIFO ファイルを初めて作成または開くと、OSS ファイルシステムが特定の CPU に OSS パイプまたは OSS FIFO FILE オープンのデータバッファーを作成し、FILE エンティティレコードも作成します。この FILE エンティティは、すべてのプロセスでパイプまたは FIFO が閉じられているときに削除されます。パイプまたは FIFO を使用するすべてのプロセスに、FILE エンティティカウンターレコードが含まれることになります。

Guardian ファイルのオープンでは、通常、測定中にプロセスによって使用されないファイルレコードが表示されます。OSS ファイルオープンでは、このようなオープンではデータレコードは生成されません。

OSS の命名規則

Measure の FILE レコードは、その Guardian 形式の名前（$*vol.subvol.fname*）によってファイルオープンオブジェクトを識別します。OSS ファイルオープンオブジェクトにも、/work/files/fname などの OSS ファイルパス名が定義されている場合があります。OSS の AF_INET および AF_INET6 ソケットの追加の識別子である IP-ADDR は、ポートとアドレスがスペースで区切られた、引用符で囲まれた文字列で指定された IP アドレスとポートの組み合わせです（たとえば、"123.45.67.8 90"または"3FFE:1200:215:164"）。

OSS の通常のファイルを除き、OSS ファイルオープンのタイプには使用できる有効な Guardian ファイル名はありません。Measure で使用する場合に限って、OSS ファイルシステムが、構成、識別、および選択のためにこれらのファイルオープンの Guardian ファイル名を作成します。これらの名前は、Measure レコードの選択および表示専用です。FILE_OPEN_呼び出しのターゲットとして使用することはできず、それ以外の場合でも、ファイル名をパラメーターとして使用するコマンドでは適用できません。名前を形成するため、OSS ファイルシステムは、プロセスのデバイス名を使用してリソースとその後に一意の識別子を指定します（*qual1*、*qual2*）。

- AF_INET および AF_INET6 ソケットの場合は TCP/IP プロセス名（ユーザー定義）
- パイプの場合はパイプサーバー名（$ZPP

  nn

  ）、および FIFO の場合は（$ZPPNN）
- AF_UNIX ソケットの場合はローカルサーバー（$ZPLS）（AF_UNIX R2 より前）
- バインドされていない AF_UNIX ソケットの場合はローカルサーバー（$ZLS

  nn

  ）、およびバインドされた AF_UNIX ソケット（AF_UNIX R2）の場合は（$ZLSNN）

どのケースでも、一意のファイル名修飾子を形成するために、以下のリストで *cccccc* と示されている CRVSN/タイムスタンプがファイル名に付加されます。

OSS ファイルオープンのタイプに応じて、Measure ファイルレコード内にファイル識別子があります。OSS ジャーナリングが測定に対して有効になっている場合は、通常のファイル、FIFO、およびバインドされた AF_UNIX ソケットの OSS ファイルパス名のみがジャーナリングされます。

パイプサーバーとローカルサーバー（AF_UNIX R2）は、実際には、CPU ごとに 1 つのプロセスがあるプロセスグループです。プロセス名の最後の 2 バイト（*nn*）は、個々のプロセスが存在する CPU 番号です。たとえば、CPU が 4 つのシステムのパイプサーバーは、プロセス$ZPP00、$ZPP01、$ZPP02、および$ZPP03 で構成されるプロセスグループです。FIFO とバインドされたソケットに対して単一のサーバープロセスと錯覚させるために、LIST FILE 表示に表示されるファイル識別子は、パイプサーバーにはプロセス名$ZPPNN を使用し、バインドされたソケットのローカルサーバーには$ZLSNN を使用します。実際のプロセス名（*nn* が指定されたもの）は、引き続き LIST FILE 表示にデバイス名として表示されます。以下は、バインドされたソケットを示す LIST FILE 表示の例です。

```
> meascom;add ossfidfb;list file * (osssocket)
File Open $ZLSNN.Z00000.Z0001SQG:106899423347         Open Type OSSUNIXSTREAM
OSSPath: "/tmp/kenm/mysock"
Device Name $ZLS01            Device Type  0  (Process)      Subdevice Type
0
Opener 1,576    ($Z01H)       File Num 2       OSSPID:  64356356
Program  $OSS2.ZYQ00000.Z000161Y:929432249
OSSPath: "/tmp/kenm/server.exe"
Opener Device Name $OSS2
Format Version:  H04  Data Version:  H04  Subsystem Version:  0
Local System \MEASYOS From  14 Nov 2008, 14:50:33   For  19.7 Seconds
```

OSS ファイルオープンのタイプ	**ファイル識別子**	
通常のファイル	$*VOL*.ZYQ*nnnnn*.Ziiiiiii:*cccccc*	OSSPath
FIFO	$ZPPNN.Z*nnnnn*.Ziiiiiii:*cccccc*	OSSPath
パイプ	$ZPP*nn*.*qual1*.*qual2*:*cccccc*	OSSPath または IP-Addr 以外
AF_INET データグラムソケット（未バインド、未接続、または接続済み）	$ZTC*nn*.*qual1*.*qual2*:*cccccc*	OSSPath または IP-Addr 以外
AF_INET データグラムソケット（バインド）	$ZTC*nn*.*qual1*.*qual2*:*cccccc*	IP-Addr
AF_UNIX データグラムソケット（未バインド、未接続、または接続済み）（AF_UNIX R2 より前）	$ZPLS.*qual1*.*qual2*:*cccccc*	OSSPath または IP-Addr 以外
AF_UNIX データグラムソケット（バインド）（AF_UNIX R2 より前）	$ZPLS.Z*nnnnn*.Z*iiiiiii*:*cccccc*	OSSPath
AF_UNIX データグラムソケット（未バインド、未接続、または接続済み）（AF_UNIX R2）	$ZLS*nn*.*qual1*.*qual2*:*cccccc*	OSSPath または IP-Addr 以外
AF_UNIX データグラムソケット（バインド）（AF_UNIX R2）	$ZLSNN.Z*nnnnn*.*Ziiiiiii*:*cccccc*	OSSPath

表は続く

OSS ファイルオープンのタイプ	ファイル識別子	
AF_INET ストリームソケット（未バインドまたは未接続）	$ZTC*nn.qual1.qual2*:*cccccc*	OSSPath または IP-Addr 以外
AF_INET ストリームソケット（バインドまたは接続済み）	$ZTC*nn.qual1.qual2*:*cccccc*	IP-Addr
AF_UNIX ストリームソケット（未バインドまたは未接続）（AF_UNIX R2 より前）	$ZPLS.*qual1.qual2*:*cccccc*	OSSPath または IP-Addr 以外
AF_UNIX ストリームソケット（バインドまたは接続済み）（AF_UNIX R2 より前）	$ZPLS.*Znnnnn.Ziiiiiii*:*cccccc*	名前付きのソケット接続の OSSPath
AF_UNIX ストリームソケット（未バインドまたは未接続）（AF_UNIX R2）	$ZLS*nn.qual1.qual2*:*cccccc*	OSSPath または IP-Addr 以外
AF_UNIX ストリームソケット（バインドまたは接続済み）（AF_UNIX R2）	$ZLSNN.*Znnnnn.Ziiiiiii*:*cccccc*	名前付きのソケット接続の OSSPath

IP-ADDR

IP-ADDR の値の重要性の決定はアプリケーションによって異なります。一般に、サーバー内の連結または接続されているストリームソケットは、その独自のアドレスが IP-ADDR にあります。クライアントプロセス内の連結または接続されているソケットには、サーバーのアドレスが与えられています。

測定 ADD FILE コマンドの基準として IP-ADDR を使用することはできませんが、LIST FILE *で IF IP-ADDR = "address port"のように選択用にアドレスを使用して、表示するレポートを選択できます。たとえば次のような限定的なワイルドカードの使用がサポートされています。

```
{ "address port"     }
{ "address *"        }
{ "* port"            }
```

IP フィールド値について詳しくは、**IP**(297 ページ)を参照してください。使用例は**コマンドの例：OSS ファイルのオープン**(305 ページ)にあります。

ANSI SQL の命名規則

Measure G11 以降の PVU では、FILE エンティティはその出力に Guardian ファイル名の代わりに ANSI SQL 名を指定できます。SQL の命名規則についての詳細は、*SQL/MX Reference Manual* を参照してください。

FILE エンティティのエンティティ指定構文

FILE エンティティを記述するには、次の構文を使用します。

```
FILE entity-spec [,entity-spec] ...
```

FILE

D シリーズの RVU では、1 つまたは複数の論理ファイルでの操作に関する情報を収集します。ファイルはローカルまたはリモートとすることができます。ただし、ファイルにアクセスするローカルプロセスのみが測定されます。

G シリーズ以降の RVU では、明示的に開いているファイル上のプロセスによって実行された I/O 操作を測定します（論理的ファイルアクセス）。ファイルはローカルまたはリモートとすることができます。ただし、ファイルにアクセスするローカルプロセスのみが測定されます。

entity-spec

次のように指定します。

```
file [ (opener) ] [(file^open^type [, file^open^type, ...] ) ]
```

指定する値は次の通りです。

file

測定対象のファイルです。*file* は、以下のいずれかのように指定します。

```
{ *                                                    }
{ [ [ [\system.]$device.]subvolume.]filename[:crvsn]   }
{ [ [\system.]$device.]#tempfile                       }
{ [ [\system.]$device.]#subdevice                      }
{ "pathname"                                              }
{ '{TABLE|INDEX} [ [catalog.]schema.]object'              }
{ 'SCHEMA [ [catalog.]schema'                             }
{ 'CATALOG {catalog|* }'                                  }
```

指定する値は次の通りです。

*****

ローカルプロセスによってアクセスされる、ローカルとリモートの両方のすべてのファイル、一時ファイル、およびサブデバイスを測定します。

system

測定対象のファイルが含まれるシステムの名前です。すべてのシステムを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのシステムです。

$*device*

測定対象のファイルが含まれるボリューム（デバイス）の名前です。すべてのデバイスを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのデバイスです。

subvolume

測定対象のファイルが含まれるサブボリュームの名前です。すべてのサブボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのサブボリュームです。

filename

測定対象のファイルの名前です。すべてのファイル（別途指定される一時ファイルを除く）を指定するには、アスタリスク（*）を使用します。

:*crvsn*

（Measure G09 以降の PVU）一意のファイル名を形成するために必要な、タイムスタンプ、作成バージョンのシリアル番号（CRVSN）、またはファイル名拡張子です。このオプションを使用して、ファイル名の一意性を保証します。CRVSN は Measure レポートおよび LISTGNAME コマンドから使用できます。

#tempfile

測定対象の一時ファイルのファイル識別番号です。すべての一時ファイルを指定するには、アスタリスク（*）を使用します。

#subdevice

測定対象のサブデバイスの名前です。すべてサブデバイスを指定するには、アスタリスク（*）を使用します。

"*pathname*"

（Measure G09 以降の PVU）完全修飾または部分修飾のいずれかの OSS ファイルパス名です。スラッシュ（/）で始まらない OSS ファイルパス名は、部分的なパス名と見なされ、OSSPATH の現在の設定を前に指定することで拡張されます。

> **注記:**
>
> OSS ファイルパス名は大文字と小文字が区別され、二重引用符（""）で囲んで指定する必要があります。有効な OSS ファイルパス名は、特定のファイル、または特定のディレクトリ内のファイルのセットを参照できます。ディレクトリが指定されている場合は、そのディレクトリ内のファイルのみが含まれます。指定したディレクトリの下位のディレクトリ内のファイルは含まれません。

catalog*、*schema*、*object

（Measure H01 以降の PVU）ANSI SQL 識別子です。省略すると、*catalog* および *schema* フィールドは、MEASCOM の *catalog* および *schema* の環境値によって解決されます。名前空間キーワード TABLE または INDEX が省略された場合、デフォルトとして TABLE が推定されます。

ワイルドカードの構文は次のとおりです。

```
SCHEMA [catalogC.]schemaS – schemaS 内のすべてのオブジェクト
CATALOG catalogC – catalogC 内のすべてのオブジェクト
CATALOG * – ノード上で表示されるすべてのオブジェクト
```

opener

指定されたファイルでの I/O 操作が測定されるプロセスです。*opener* は、以下のいずれかのように指定します。

```
{ cpu, pin [ , filenum ]                    }
{ pname [ , filenum   ]                     }
{ "pname" [ , filenum ]                     }
```

cpu

プロセスが実行されている CPU の番号です。

pin

プロセスのプロセス識別番号です。

filenum

プロセスのファイル番号です。

pname

（D シリーズの RVU）プロセス名またはプロセスのプログラムファイル名です。

"pname"

（Measure G09 以降の PVU）プロセスのプログラムファイル名の OSS ファイルパス名です。/G で始まる OSS ファイルパス名はサポートされていないため、代わりに Guardian ファイル名を使用してください。

file^open^type

（Measure G10 以降の PVU）1 つ、すべて、またはいくつかのファイルオープンタイプを測定用に選択します。デフォルトでは、Guardian ファイルオープンのみが測定されます。有効なオプションには以下のリテラルがあります。

ALLFILES	測定についてのすべてのファイルオープンタイプを選択します（ADD FILE および LIST コマンドのデフォルト）。Guardian、OSS、および ANSI SQL ファイルを含みます
GUARDIAN	すべての Enscribe、プロセス、および SQL ファイルオープンを選択します（ADD コマンドの現在のデフォルト）
ENSCRIBE	非構造化ファイル、相対ファイル、エントリーシーケンスファイル、およびキーシーケンスファイルの Enscribe オープンを選択します
UNSTRUCT	非構造化ファイルまたは通常の OSS ファイルの Enscribe オープンを選択します
RELFILE	相対ファイルの Enscribe オープンを選択します
ENTRYFILE	エントリーシーケンスファイルの Enscribe オープンを選択します
KEYFILE	キーシーケンスファイルの Enscribe オープンを選択します
SQLFILE	SQL ファイルオープンを選択します
PROCFILE	プロセスおよびその他の非ディスクファイルの Enscribe オープンを選択します
OSS	すべての OSS オープンが含まれていることを指定します
OSSDISK	通常のファイルの OSS オープンを選択します
OSSFIFO	名前付きパイプの OSS オープンを選択します
OSSPIPE	名前なしパイプの OSS オープンを選択します
OSSSOCKET	すべてのソケットの OSS オープンを選択します
OSSINETSOCKET	AF_INET ソケットの OSS オープンを選択します
OSSINETDGRAM	AF_INET データグラムソケットの OSS オープンを選択します
OSSINETSTREAM	AF_INET ストリームソケットの OSS オープンを選択します

表は続く

OSSUNIXSOCKET　AF_UNIX ソケットの OSS オープンを選択します

OSSUNIXDGRAM　AF_UNIX データグラムソケットの OSS オープンを選択します

OSSUNIXSTREAM　AF_UNIX ストリームソケットの OSS オープンを選択します

FILE エンティティの DDL レコード（ZMS 形式）

FILE エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsfile-id.
 02 opener-pin                    type binary 16 unsigned.
 02 file-number                   type binary 16 unsigned.
 02 file-type                     type binary 16 unsigned.
 02 device-type                   type binary 16 unsigned.
 02 file-name.
    03 volume                     type character 8.
    03 subvol                     type character 8.
    03 filename                   type character 8.
 02 file-system-name              type character 8.
 02 opener-processname            type character 8.
 02 opener-program-filename.
    03 volume                     type character 8.
    03 subvol                     type character 8.
    03 filename                   type character 8.
 02 device-name                   type character 8.
 02 opener-device-name            type character 8.
 02 file-name-MID.
    03 PATHID                     type character 24.
    03 CRVSN                      type character 6.
 02 reserved-1                    type character 6.
 02 opener-osspid                 type binary 32 unsigned.
 02 opener-program-fname-MID.
    03 PATHID                     type character 24.
    03 CRVSN                      type character 6.
 02 file-open-type                type binary 16 unsigned.
 02 IP.
    03 family                     type binary 16 unsigned.
    03 port                       type binary 16 unsigned.
    03 IP-addr                    type character 16.
 02 reserved-2                    type character 4.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsfile-ctrs.
 02 file-busy-time                type binary 64.
 02 reads                         type binary 64.
 02 writes                        type binary 64.
```

```
 02 updates-or-replies           type binary 64.
 02 deletes-or-writereads        type binary 64.
 02 info-calls                   type binary 64.
 02 records-used                 type binary 64.
 02 records-accessed             type binary 64.
 02 disc-reads                   type binary 64.
 02 messages                     type binary 64.
 02 message-bytes                type binary 64.
 02 lock-waits                   type binary 64.
 02 timeouts-or-cancels          type binary 64.
 02 escalations                  type binary 64.
 02 DBIO-reads                   type binary 64.
 02 DBIO-writes                  type binary 64.
 02 DBIO-read-bytes              type binary 64.
 02 DBIO-write-bytes             type binary 64.
 02 read-bytes                   type binary 64.
 02 write-bytes                  type binary 64.
 02 OSS-cache-reads              type binary 64.
 02 OSS-cache-writes             type binary 64.
 02 OSS-cache-read-bytes         type binary 64.
 02 OSS-cache-write-bytes        type binary 64.
 02 OSS-block-reads              type binary 64.
 02 OSS-block-read-bytes         type binary 64.
 02 OSS-flow-controls            type binary 64.
 02 misc-calls                   type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsfile. FILE is "zmsfile" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 ctr                          type zmsfile-ctrs.
 02 id                           type zmsfile-id.
end
```

FILE エンティティの DDL レコード（Legacy 形式）

FILE エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G シリーズ FILE エンティティの DDL レコードは、次の点を除いて D シリーズ FILE エンティティのレコードと同じです。

- ダイレクトバルク I/O 操作を測定するためのカウンターが追加されます。
- フィールドのオーバーフローの可能性を減らすために、長い byte-count フィールドが提供されます。32 ビットの各 byte-count フィールドには、対応する 64 ビットがあります。64 ビットのフィールドの使用については、**SYSTEM**(163 ページ)を参照してください。
- ERROR フィールドが、32 ビットの byte-count フィールドのフィールドオーバーフローを通知することがあります。
- OSS ファイルパス名のサポートのための識別子が追加されます。

```
RECORD file. FILE is "file" ENTRY-SEQUENCED.
                      .
                      .
                      .
```

(error、time items、および measurement identification items については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください)

```
                    .
                    .
                    .
 02 opener-cpu                    type binary 16 unsigned.

* entity identification items:
 02 opener-pin                    type binary 16 unsigned.
 02 file-number                   type binary 16 unsigned.
 02 file-name.
    03 volume                      type character 8.
    03 subvol                      type character 8.
    03 filename                    type character 8.
 02 file-system-name              type character 8.
 02 file-type                     type binary 16 unsigned.
 02 device-type                   type binary 16 unsigned.

* counter value items:
 02 file-busy-time                type binary 64.
 02 reads                          type binary 32 unsigned.
 02 writes                         type binary 32 unsigned.
 02 updates-or-replies            type binary 32 unsigned.
 02 deletes-or-writereads         type binary 32 unsigned.
 02 info-calls                    type binary 32 unsigned.
 02 records-used                  type binary 32 unsigned.
 02 records-accessed              type binary 32 unsigned.
 02 disc-reads                    type binary 32 unsigned.
 02 messages                       type binary 32 unsigned.
 02 message-bytes                 type binary 32 unsigned.
 02 lock-waits                    type binary 32 unsigned.
 02 timeouts-or-cancels           type binary 32 unsigned.
 02 escalations                    type binary 32 unsigned.

* new entity identification items for D10:
 02 opener-processname            type character 8.
 02 opener-program-filename.
    03 volume                      type character 8.
    03 subvol                      type character 8.
    03 filename                    type character 8.

* SMS changes:
 02 device-name                    type character 8.
 02 opener-device-name             type character 8.
* New F40 counter value items:
 02 message-bytes-f               type binary 64.

* New counters in G05:
 02 dbio-reads                     type binary 32 unsigned.
 02 dbio-writes                    type binary 32 unsigned.
 02 dbio-read-bytes                type binary 64.
 02 dbio-write-bytes               type binary 64.

* New identifiers for OSS file pathname support:
 02 file-name-mid.
    03 pathid                      type character 24.
    03 crvsn                       type character  6.
 02 opener-osspid                  type binary 32 unsigned.
 02 opener-program-fname-mid.
```

```
    03 pathid                    type character 24.
    03 crvsn                     type character  6.

* New identifier for G10:
 02 File-open-type               type binary 16 unsigned.
 02 IP.
    03 family                    type binary 16 unsigned.
    03 port                      type binary 16 unsigned.
    03 IP-addr                   type character 16.

* New counters for G10:
 02 read-bytes                   type binary 64.
 02 write-bytes                  type binary 64.
 02 OSS-cache-reads              type binary 64.
 02 OSS-cache-writes             type binary 64.
 02 OSS-cache-read-bytes         type binary 64.
 02 OSS-cache-write-bytes        type binary 64.
 02 OSS-block-reads              type binary 32 unsigned.
 02 OSS-block-read-bytes         type binary 64.
 02 OSS-flow-controls            type binary 32 unsigned.
 02 misc-calls                   type binary 32 unsigned.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

FILE エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

DEVICE-NAME

測定ファイルが配置されているディスクデバイス。SMF ファイルの場合、このフィールドには、論理ファイル名に対応する物理的な場所が表示されます。NonStop ファイルの場合、このフィールドは、FILE-NAME VOLUME サブフィールドと同じです。

DEVICE-TYPE

次のいずれかの値で、ファイルに関連付けられているデバイスのタイプを示します。

0	プロセス
1	オペレーター
2	$RECEIVE
3	ディスク
4	テープドライブ
5	プリンター
6	端末

表は続く

7	通信回線
8	SCSI

FILE-NAME

測定対象のファイルの名前。この名前は、NonStop ファイルか SMF ファイルのいずれかに適用できます。NonStop ファイルでは、プライマリディスクファイルがパーティションまたは代替キーファイルに関連付けられている場合、すべてのアクティビティがプライマリファイル名の下で記録されます。FILE-NAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、ディスクファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、ファイルを開いたときに指定された物理ファイル名を表します。 SMF ファイルでは、ファイルを開いたときに使用された、位置に依存しない論理ファイル名を表します。

FILE-NAME-MID

OSS ファイルまたは SQL/MX ファイルの内部形式表現。FILE-NAME-MID には次の 2 つのサブフィールドがあります。

PATHID	OSS ファイルまたは SQL/MX ファイルの内部形式表現。その他のファイルでは、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルまたは SQL/MX ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。その他のファイルでは、このフィールドにはゼロが含まれます。

FILE-NUMBER

ファイルの数。

FILE-OPEN-TYPE

測定のために追加されるかリスト表示レポートのために選択されるファイルオープンタイプの識別子。利用可能な値は次のとおりです。

%H000	UNSTRUCT
%H001	RELFILE
%H002	ENTRYFILE
%H003	KEYFILE
%H008	PROCFILE
%H081	SQLFILE
%H101	OSSFIFO

表は続く

%H102	OSSPIPE
%H201	OSSDISK
%H411	OSSUNIXSTREAM
%H412	OSSUNIXDGRAM
%H421	OSSINETSTREAM
%H422	OSSINETDGRAM

FILE-SYSTEM-NAME

完全なファイル指定からのシステムの名前です。

FILE-TYPE

アクセス制御ブロックからのファイルタイプ。

1	相対ファイル
2	エントリーシーケンスファイル
3	キーシーケンスファイル
0	他のタイプのファイル

IP

IP アドレスの値。IP には、次の 3 つのサブフィールドがあります。

FAMILY	IPv4 アドレス指定を使用するか、IPv6 アドレス指定を使用するかを指定します。このフィールドの指定可能な値は、2（IPv4）または 26（IPv6）です。
PORT	ポート番号が含まれています。
IP-ADDR	IPv4 と IPv6 では異なって解釈されます。FAMILY フィールドで IPv4 アドレス指定を指定している場合（FAMILY の値が 2 の場合）、IP-ADDR フィールドの最初の 4 バイトには、ドットで区切られた 10 進数形式の IP アドレスの 4 つの部分を表す数字の 16 進値が含まれます。FAMILY フィールドで IPv6 アドレス指定を指定している場合（FAMILY の値が 26 の場合）、IP-ADDR フィールドのすべての 16 バイトは、IP アドレスの数字に対応する 16 進数値を収めるために使用されます。

IP-ADDR の値の重要性の決定はアプリケーションによって異なります。一般に、サーバー内の連結または接続されているストリームソケットは、その独自のアドレスが IP-ADDR にあります。クライアントプロセス内の連結または接続されているソケットには、サーバーのアドレスが与えられています。詳しくは、**IP-ADDR**(288 ページ)、**OSS の OSS AF_INET および AF_INET6 ストリームソケットのオープン**(307 ページ)、および **OSS の OSS AF_INET および AF_INET6 データグラムソケットのオープン**(308 ページ)を参照してください。

IP-ADDR 値は、ADD FILE コマンドで測定の基準として使用することはできません。レポートを表示する選択オプションとして、LIST FILE コマンドの IF 句で、ポートをアドレスからスペースで区切って引用符で囲んだ IP アドレス（"123.45.67.8 90"や"3FFE:1200:215:1 64"など）を指定できます。

OPENER-DEVICE-NAME

オープナープログラムファイルが配置されているディスクデバイス。SMF ファイルの場合、このフィールドには、論理ファイル名に対応する物理的な場所が表示されます。NonStop ファイルの場合、このフィールドは、OPENER-PROGRAM-FILENAME VOLUME サブフィールドと同じです。

OPENER-OSSPID

オープナープロセスが OSS 環境で実行されている場合、このフィールドには OSS プロセス ID が含まれます。このプロセスが OSS プロセスではない場合、このフィールドにはゼロが含まれます。

OPENER-PIN

ファイルを開いたプロセスのプロセス識別番号。

OPENER-PROCESSNAME

オープナープロセスの名前。

OPENER-PROGRAM-FILENAME

オープナープログラムファイルの名前。この名前は、NonStop ファイルか SMF ファイルのいずれかに適用できます。OPENER-PROGRAM-FILENAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、プログラムファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、ファイルオープナーの物理プログラムファイル名を表します。 SMF ファイルでは、ファイルオープナーの位置に依存しない論理ファイル名を表します。

OPENER-PROGRAM-FNAME-MID

OSS ファイル名の内部表現。OPENER-PROGRAM-FNAME-MID には次の 2 つのサブフィールドがあります。

PATHID	OSS ファイル名の内部表現です。OSS 以外のファイルでは、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。その他のファイルでは、このフィールドにはゼロが含まれます。

FILE エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。

DBIO-READ-BYTES

ダイレクトバルク I/O 読み取り操作によって転送されたバイト数。

カウンタータイプ：増分。

DBIO-READS

測定対象のファイルへのダイレクトバルク I/O 読み取り操作の数。このカウンターは READS のサブセットです。

カウンタータイプ：増分。

DBIO-WRITE-BYTES

ダイレクトバルク I/O 書き込み操作によって転送されたバイト数。

カウンタータイプ：増分。

DBIO-WRITES

測定対象のファイルへのダイレクトバルク I/O 書き込み操作の数。このカウンターは WRITES のサブセットです。

カウンタータイプ：増分。

DELETES-OR-WRITEREADS

バッファー長が 0 の WRITEUPDATE または WRITEUPDATEUNLOCK プロシージャーへの呼び出しの数（削除操作）、または WRITEREAD プロシージャーへの呼び出しの数（書き込み読み取り操作）。このカウンターには、これらのプロシージャーへのユーザー呼び出しとシステム呼び出しの両方が含まれます。

このカウンターの解釈は、測定対象のファイルのタイプによって異なります。削除操作は、ディスクファイルとテープファイルに対して実行されます。書き込み読み取り操作は、端末またはプロセスに対して実行されます。

カウンタータイプ：増分。

DISC-READS

実行される物理ディスク読み取りの回数。

カウンタータイプ：増分。

ESCALATIONS

ロックがファイルレベルロックにエスカレーションされた回数。

カウンタータイプ：増分。

FILE-BUSY-TIME

待機 I/O の場合、FILE-BUSY-TIME カウンターは待機 I/O 要求の実行に費やされた経過時間です。OSS ファイルオープンは、ファイルへの最初の重要なファイル操作後に、待機要求についてのファイルビジー時間をカウントします。このカウンターは、I/O 要求のメッセージ（たとえば READ または WRITE）がサーバーに送信されたときにファイルシステムによって開始されます。カウンターは、応答が受信されて応答を処理するプロセスをもう一度実行した後に停止します。このため、カウンターにはリクエスターの実行時間、サーバーの実行時間（たとえば、DP2 ディスクプロセス）、すべての物理 I/O 時間（たとえば、ディスク読み取り）が含まれるほか、両方の CPU での優先順位の高いプロセスの実行時間などが含まれる可能性があります。ファイルのオープンおよびクローズ中に送信される要求と、OSS ネームサーバーに送信される要求は、FILE-BUSY-TIME カウンターにカウントされませんが、PROCESS エンティティにはカウントされます。

J06.20/L16.05 以降の RVU の待ちなしの I/O 処理では、FILE-BUSY-TIME カウンターには、サーバーへの待ちなしの I/O 処理要求を開始するための経過時間、および各ファイルオープンに対する待ちなしの I/O 処理を待機するプロセスが AWAITIO[X]で停止する経過時間が含まれます。FILE-BUSY-TIME はファイルオープンごとに個別に累積されます。あるプロセスが AWAITIO[X]で中断されたときに、複数のファイルオープンについて未処理の待ちなし I/O 処理がある場合、これらの各ファイルオープンに対して同じプロセス待機経過時間が FILE-BUSY-TIME に追加されます。

注記:

FILE-BUSY-TIME カウンターに以下は含まれません。

- 通常のファイルでない OSS ファイルの準備が完了するまでの select()および FILE_COMPLETE_()でのプロセス待機時間。
- 通常のファイルでない OSS ファイルに関してスレッド認識 I/O 操作が完了するのを OSS スレッドアプリケーションプロセスが待機する時間。

J06.20/L16.05 より前の RVU の待ちなしの I/O 処理では、FILE-BUSY-TIME カウンターは、サーバーへの待ちなしの I/O 処理要求を開始するための実行時間と、待ちなし要求が AWAITIO[X]で完了するのを待機するためにプロセスが実際に費やした経過時間です。

OSSPIPE FILE オープンタイプでは、OSSPIPE の一般的なアプリケーション使用によって、read()または write()がメモリ間の移動で完了することができ、非常に低い FILE-BUSY-TIME が生成されます。ただし、アプリケーションが OSSFM エラー EWOULDBLOCK（4101）を無視した場合、FILE-BUSY-TIME がインクリメントします。

カウンタータイプ：ビジー。

INFO-CALLS

要求された情報を取得するために、メッセージをプロセス（IOP など）にディスパッチする、これらのプロシージャーに対する呼び出しの数。

FILEINFO	FILE_GETINFOBYNAME_	DEVICEINFO
FILERECINFO	FILE_GETINFOLISTBYNAME_	DEVICEINFO2
FILE_GETINFO_	FILEGETINFO_LIST	

OSS の場合、このファイルに対する fstat()、fstatvfs()、fpathconf()、getsockname()、getpeername()、getsockopt()、および sockatmark()の API 呼び出しの数。

このカウンターには、これらのプロシージャーへのユーザー呼び出しとシステム呼び出しの両方が含まれます。

カウンタータイプ：増分。

LOCK-WAITS

コールがロック要求を待機した回数。OSS の場合、fcntl (,F_SETLKW)ロック要求が別のロックを待機していたときにカウントします。

カウンタータイプ：増分。

MESSAGE-BYTES

このファイルオープンのために送受信されたメッセージのバイト数です。

OSS では、サーバープロセスとこのファイルの OSS ネームサーバーからの MESSAGES で送受信されたメッセージデータの合計バイト数をカウントします。

通常、MESSAGE-BYTES は、READ-BYTES および WRITE-BYTES の合計数と同じです。ただし、OSSPIPE FILE オープンタイプでは、OSSPIPE オープナーが OSSPIPE バッファーと同じ CPU にある場合、MESSAGES は read()または write()ごとには使用されません。MESSAGE-BYTES は MESSAGES が使用される場合のみ加算されるため、MESSAGE-BYTES は READ-BYTES と WRITE-BYTES の合計数とは異なることがあります。

このカウンターには、ダイレクトバルク I/O によって転送されたバイトは含まれません。DBIO-READ-BYTES および DBIO-WRITE-BYTES カウンターが、ダイレクトバルク I/O の転送を測定します。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、MESSAGE-BYTES-F フィールドが MESSAGE-BYTES の 64 ビットバージョンです。

OSS の PIPE および FIFO の場合、所有者は、PIPE が作成されたか FIFO が初めて開かれた CPU の PIPE サーバーです。IO が所有者に対してローカルの CPU の PIPE または FIFO で実行されても、MESSAGE-BYTES カウンターは更新されません。IO が所有者に対してリモートの CPU の PIPE または FIFO で実行されると、MESSAGE-BYTES カウンターが更新されます。

カウンタータイプ：累積。

MESSAGES

このオープンのために送信されたメッセージ（OPEN メッセージを含む）の数。ファイルラベルは 30KB のファイルメッセージ制限を超える場合があるため、複数の OPEN メッセージが送信されることがあります。

OSS ファイルのオープンでは、MESSAGES は、サーバープロセスとこのファイルの OSS ネームサーバーから送受信されたメッセージの合計数です。これには、PROCESS エンティティで追跡される OPEN メッセージは含まれません。OSS ソケットでは、MESSAGES および MESSAGE-BYTES は OSS ネームサーバーのアクティビティのみを反映します。

通常、MESSAGES は、READS および WRITES の合計数と同じです。ただし、OSSPIPE FILE オープンタイプでは、OSSPIPE オープナーが OSSPIPE バッファーと同じ CPU にある場合、MESSAGES は read()または write()ごとには使用されません。

カウンタータイプ：増分。

MISC-CALLS

OSS では、このカウンターは、READS、WRITES、または INFO-CALLS ではなく、FILE-BUSY-TIME が増加する動作を合計します。このカテゴリに分類される API は、ioctl()、fsync()、ftruncate()、lseek()、および fcntl()と、ソケット API の connect()、setsockopt()、accept()、および shutdown()です。

カウンタータイプ：増分。

OSS-BLOCK-READ-BYTES

OSS の通常のファイル専用で、OSS-BLOCK-READS の結果としてキャッシュに読み出されたバイトの数。

カウンタータイプ：累積。

OSS-BLOCK-READS

OSS の通常のファイル専用で、OSS キャッシュをいっぱいにするために発行された DP2 へのブロック読み取り要求の数。ブロック読み取り呼び出しは、複数のブロックを返すことがあります。

カウンタータイプ：増分。

OSS-CACHE-READ-BYTES

OSS の通常のファイル専用で、OSS-CACHE-READS によって読み取られたバイトの数。

カウンタータイプ：累積。

OSS-CACHE-READS

OSS の通常のファイル専用で、OSS キャッシュを使用して実行された READS の数。

カウンタータイプ：増分。

OSS-CACHE-WRITE-BYTES

OSS の通常のファイル専用で、OSS-CACHE-WRITES によって書き込まれたバイトの数。

カウンタータイプ：累積。

OSS-CACHE-WRITES

OSS の通常のファイル専用で、OSS キャッシュを使用して実行された WRITES の数。

カウンタータイプ：増分。

OSS-FLOW-CONTROLS

OSS のソケットとパイプの場合、このカウンターは、WRITE 操作がブロックされた回数を報告します。

カウンタータイプ：増分。

READ-BYTES

ユーザーアプリケーションのデータバッファーに返されたバイト数。すべてのファイルオープンタイプで使用できます。

カウンタータイプ：累積。

READS

D シリーズの RVU では、READX や OSS ファイルシステムの読み取り API（read()や recv()など）といったファイルシステムの読み取り操作への呼び出しの数。OSS では、合計には OSS-CACHE-READS が含まれます。Enscribe ファイルシステムの場合、合計には非 DBIO および DBIO の操作が含まれます。このカウンターには、これらのプロシージャーへのユーザー呼び出しとシステム呼び出しの両方が含まれます。

G シリーズの RVU では、READ、READLOCK、READUPDATE、および READUPDATELOCK プロシージャーへの呼び出しの数。このカウンターには、これらのプロシージャーへのユーザー呼び出しとシステム呼び出しの両方が含まれます。G05 以降の RVU では、このカウンターにはダイレクトバルク I/O 読み取り操作が含まれます。

カウンタータイプ：増分。

RECORDS-ACCESSED

操作を実行するためにディスクプロセスまたはファイルシステムによって読み取られたレコードの数。アクセスされたレコードの数は、使用されたレコードの数以上になります。RECORDS-ACCESSED に対する RECORDS-USED の比率は、文の選択性を示します。このカウンターの値は、Enscribe ファイルではゼロです。

カウンタータイプ：増分。

RECORDS-USED

読み取り、挿入、書き込み、更新、および削除のときに SQL エグゼキューターに返されるレコードの数。このカウンターの値は、Enscribe ファイルではゼロです。

カウンタータイプ：増分。

TIMEOUTS-OR-CANCELS

この OPEN に対して発行されたタイムアウトまたはキャンセルの数。

カウンタータイプ：増分。

UPDATES-OR-REPLIES

バッファー長がゼロより大きい WRITEUPDATE または WRITEUPDATEUNLOCK プロシージャーへの呼び出し回数（更新操作）、または REPLY プロシージャーへの呼び出し回数（応答操作）。このカウンターには、これらのプロシージャーへのユーザー呼び出しとシステム呼び出しの両方が含まれます。

このカウンターの解釈は、測定対象のファイルのタイプによって異なります。更新操作は、ディスクファイルとテープファイルに対して実行されます。応答操作は、端末またはプロセスに対して実行されます。

カウンタータイプ：増分。

WRITE-BYTES

ユーザーアプリケーションバッファーから書き込まれたバイト数。すべてのファイルオープンタイプで使用できます。

カウンタータイプ：累積。

WRITES

D シリーズの RVU では、WRITEX などのファイルシステムの書き込み操作、および write()や send()などの OSS ファイルシステムの書き込み API への呼び出し回数を表します。OSS の場合、合計には OSS-CACHE-WRITES が含まれます。Enscribe ファイルシステムの場合、合計には非 DBIO および DBIO の操作が含まれます。このカウンターには、ユーザーとシステムの両方からの WRITE 呼び出しが含まれます。

G シリーズの RVU では、WRITE プロシージャーへの呼び出し回数を表します。このカウンターには、ユーザーとシステムの両方からの WRITE 呼び出しが含まれます。G05 以降の RVU では、このカウンターにはダイレクトバルク I/O 書き込み操作が含まれます。

カウンタータイプ：増分。

FILE エンティティの Legacy 形式のみの ID フィールドの DDL 定義

この ID フィールドは、Legacy 形式の DDL レコードのみで使用されます。

OPENER-CPU

（Legacy 形式のみ）ファイルを開いたプロセスが実行されている CPU の番号。

FILE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

このカウンターフィールドは、Legacy 形式の DDL レコードのみで使用されます。

MESSAGE-BYTES-F

（G シリーズ、Legacy 形式のみ）MESSAGE-BYTES と同じですが、より大きい値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積

すべての FILE エンティティの使用上の注意

- FILE カウンターは、データベースアクセスに関する潜在的な問題を検出する場合に役立ちます。
- Measure サブシステムは、ローカルエンティティだけを追跡します。そのため、FILE エンティティには、ローカルユーザープロセスによって実行された I/O 操作のみが含まれます。アクセスされるファイルは、測定の設定によっては、ローカルであることも、リモートであることも、その両方であることもあります。
- FILE エンティティは、パーティション分割されたファイルのすべてのパーティションを一緒にして、単一のエンティティとして測定します。パーティションを個別に測定するには、DISCOPEN エンティティを使用します。

- FILE エンティティは、ユーザーのアプリケーションによって明示的に開かれていないファイル（代替キーファイルなど）を測定できません。
- Remote Data Facility および TMF の I/O 操作はディスクプロセスで直接インターフェイスによって実行されるため、これらの操作は FILE エンティティでは報告されません。

H シリーズ以降の FILE エンティティの使用上の注意

- H シリーズ以降の RVU では、すべての byte-count フィールドが 64 ビットに対応しています。-F で終わるフィールド名は、ZMS 形式のレコードでは使用されなくなりましたが、Legacy 形式でデータを要求するアプリケーションでは引き続き使用できます。
- H シリーズ以降の RVU では、ZMSFILE レコードの SUBSYSTEM-VERSION は、ZMSFILE レコードが表すオープンのタイプに応じて、Guardian または OSS ファイルシステムによって提供されます。

G シリーズの FILE エンティティの使用上の注意

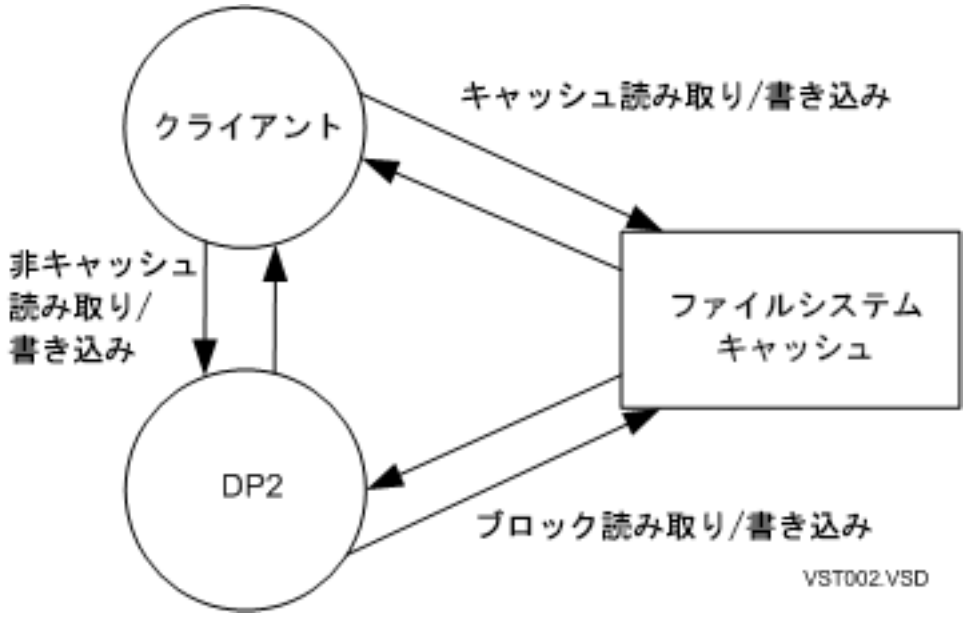
- Measure G11 以降の PVU では、FILE はすべての ANSI SQL ファイルのオープンを記録します。
- G シリーズの RVU では、SUBSYSTEM-VERSION の値はゼロです。
- 64 ビットのバイトカウントフィールド（-F で終わるフィールド）は、従来の 32 ビットのバイトカウントフィールドと同じデータを収集します。たとえば、64 ビットフィールドの MESSAGE-BYTES-F は、32 ビットフィールドの MESSAGE-BYTES と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

  32 ビットフィールドは現在アクティブであり、引き続き値を返します フィールドのオーバーフローが存在しない場合、32 ビットフィールドと 64 ビットのフィールドは同じ値を返します。32 ビットフィールドがオーバーフローした場合、対応する 64 ビットフィールドは正しい値を返し、32 ビットフィールドは-1 の値を返します。測定対象のエンティティの ERROR フィールドも、オーバーフロー状態を示す-1 を返します。

  64 ビットフィールドを使用するには、アプリケーションを変換してください。32 ビットのフィールドは、今後のリリースでは使用できなくなる可能性があります。

  MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットフィールドの名前を使用してください。たとえば、LIST DEVICE BY INPUT-BYTES-F ではなく LIST DEVICE BY INPUT-BYTES のコマンドを発行してください。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。
- Measure G09 の PVU での OSS ファイルパス名に対するサポートでは、Guardian ファイルシステムを使用した場合にのみ OSS ファイルのオープンのファイルレコードを使用できます。Measure G10 以降の PVU では、FILE エンティティは OSS ファイルシステムによるオープンを記録します。
- OSS の通常のファイルでの読み取りまたは書き込み操作は、キャッシュされることもキャッシュされないこともあります。
  - キャッシュされない操作は、結果としてメッセージを DP2 に直接送信し、クライアントが指定した量の情報しか転送しません。
  - キャッシュされる操作は、クライアントの CPU でファイルシステムのキャッシュを利用し、結果としてキャッシュと DP2 との間のブロックの読み取り/書き込みになる可能性があります。

    ブロックの読み取りまたは書き込みは、キャッシュブロックが 4 KB であるキャッシュブロック境界（ファイルの最後のブロックを除く）で発生します。ブロックの読み取り操作は、直近の情報の必要性を見込んで先読みを試みます。



- READS および WRITES に対する FILE エンティティのカウンターは、OSS キャッシュを使用した読み取りと書き込みを含む、すべての論理ファイルの I/O 操作をカウントします。これらには、キャッシュを補充するブロック操作は含まれません。そのため、キャッシュミスでは次の 6 つのカウンターが加算されます。

READS	OSS-CACHE-READS	OSS-BLOCK-READS
READ-BYTES	OSS-CACHE-READ-BYTES	OSS-BLOCK-READ-BYTES

- キャッシュ書き込みは必ずしも所有するファイルによってトリガーされる必要はないため、FILE エンティティには、OSS ブロック書き込みまたは書き込みバイトのカウンターは含まれません。これらのカウンターは、DISKFILE エンティティに加算されます。
- OSS ファイルシステムは、内部アルゴリズムを使用してファイルのキャッシュ操作の有効性を評価します。キャッシュは、状況によってはファイルオープンのためにファイルシステムによってオフになるため、キャッシュカウンターは増分されません。

コマンドの例：OSS ファイルのオープン

これらのコマンドの使用例では、Measure ファイルレポートを選択するファイル識別子の使用について説明します。

OSS のディスクファイルのオープン

同じディスクファイルのファイルオープンは、同じ Guardian 形式のファイル名と OSS ファイルパス名を共有します。

同じディスクファイルのファイルオープンを選択するには、その Guardian 形式のファイル名または OSS ファイルパス名のいずれかを使用します。

```
LIST FILE $VOL.SUBVOL.FILE:CRVSN

LIST FILE "/a/b/c"
```

Guardian 形式の名前をマスクすると、同じデバイス名または同じデバイス名とサブボリューム名を持つディスクファイルのオープンを選択できます。

```
LIST FILE $VOL.*.*

LIST FILE $VOL.SUBVOL.*
```

Measure ファイルオープンタイプは、OSS の通常のファイルと/G:で開いた Guardian ファイルの OSS ファイルオープンを選択します。

```
LIST FILE * (OSSDISK)

LIST FILE $VOL.*.* (OSSDISK)
```

OSS の FIFO のオープン

同じ OSS FIFO の OSS ファイルオープンは、Guardian 形式のファイル名または OSS ファイルパス名によって選択します。FIFO ファイルパス名の LISTGNAME コマンドは、$ZPPNN という形式の Guardian デバイス名を返します。Measure のレコードには、すべての FIFO オープンに対して同じ形式（$ZPPNN.*Znnnnn*.*Ziiiiiii*:CRVSN）のファイルオープン名が表示されます。ただし、レコードのデバイス名フィールドでは、NN は常に FIFO オープナーの CPU 番号で置き換えられます。異なる CPU の複数のオープナーがアクティブである場合、$ZPP デバイス名は異なります。FIFO パス名による LIST FILE は、すべてのインスタンスを返します。

```
LIST FILE $ZPPNN.*.*:cccccccccc      (cccccccccc is the CRSVN)
LIST FILE "/a/b/fifoz"
```

Measure ファイルオープンタイプを使用して、OSS FIFO の OSS ファイルオープンを選択できます。

```
LIST FILE * (OSSFIFO)
LIST FILE $ZPPNN.*.* (OSSFIFO)
```

Expand ネットワーク環境では、OSS ファイルオープンがリモートシステムに配置されている FIFO に接続してもリモートが G06.17 以降の PVU で実行されていない場合、Measure ファイルレコードは作成されません。次に例を示します。

```
+ LIST FILE $ZPPNN.*.* (OSSFIFO)
File Open $ZPPNN.Z00001.Z0007HRX:106589078765          Open Type OSSFIFO
OSSPath: "/home/software/markr/x"
Device Name $ZPP04            Device Type  0  (Process)      Subdevice Type
0
Opener 4,397                  File Num 6       OSSPID:  269223152
Program  $ROOT.ZYQ00000.Z0000HJM:874302705
```

OSS のパイプのオープン

同じパイプの OSS オープンは、同じ Guardian 形式のファイル名を持ちますが、OSS ファイルパス名は持ちません。パイプ（読み取り/書き込み）を形成するために使用される両方のファイルが、同じ Guardian 形式のファイル名を共有します。Measure レコードは、その作成によってではなく、パイプのファイルオープンによって作成されます。

同じパイプの OSS ファイルオープンを選択するには、その Guardian 形式のファイル名を使用します。

```
LIST FILE $ZPPnn.*.*:cccccccccc (nn =CPU number, cccccccccc = CRSVN))
```

同じデバイス名を持つ OSS パイプと FIFO のオープンを選択するには、Guardian 形式の名前をマスクします。

```
LIST FILE $ZPPnn.*.*                    (nn =CPU number)
```

Measure ファイルオープンタイプは、OSS パイプの OSS ファイルオープンを選択します。

```
LIST FILE * (OSSPIPE)
```

OSS の AF_UNIX ソケットのオープン

OSS AF_UNIX ソケットの OSS ソケットオープンを選択するには、Guardian 形式の名前をマスクするか、Measure ファイルオープンタイプを使用します。

```
LIST FILE $ZPLS.*.*             (pre-AF_UNIX R2)
LIST FILE $ZLSNN.*.*            (AF_UNIX R2 - bound sockets)
```

```
LIST FILE $ZLSnn.*.*            (AF_UNIX R2 - unbound sockets)
LIST FILE * (OSSUNIXSOCKET)
LIST FILE * (OSSUNIXSTREAM)
LIST FILE * (OSSUNIXDGRAM)
```

OSS の socketpair()を使用した AF_UNIX ソケットのオープン

socketpair()による AF_UNIX ソケットの OSS オープンは同じ Guardian 形式のファイル名を共有しますが、OSS ファイルパス名は持ちません。socketpair()によって作成された両方のソケットは同じ Guardian 形式のファイル名を共有するため、それによって共有の OSS ソケットのオープンを選択できます。

```
LIST FILE $ZPLS.*.*     (pre-AF_UNIX R2)
LIST FILE $ZLSnn.*.*    (AF_UNIX R2 - unbound sockets, nn =CPU number)
```

OSS の socket()または accept()を使用した AF_UNIX ストリームソケットのオープン

サーバーのソケットアドレスによって示されるソケット接続に参加している OSS AF_UNIX ストリームソケットの OSS オープンはすべて、リスニングソケットの同じ Guardian 形式のファイル名と OSS ファイルパス名を共有します。これには、bind()、connect()、および accept()呼び出しに関連するソケットが含まれます。バインドされていない未接続の AF_UNIX ストリームソケットの OSS オープンは、一意の Guardian 形式のファイル名を持ち、OSS ファイルパス名は持ちません。

同じ AF_UNIX ソケットストリーム接続の OSS ソケットのオープンを選択するには、リスニングソケットの Guardian 形式のファイル名または OSS ファイルパス名を使用します。

```
LIST FILE $ZPLS.xxxxxx.xxxxxxxx:CRVSN
LIST FILE $ZLSNN.*.*          (AF_UNIX R2 - bound sockets)
LIST FILE "/a/b/socketf"
```

OSS の AF_UNIX データグラムソケットのオープン

バインドされている AF_UNIX データグラムソケットのオープンは、一意の Guardian 形式の名前と OSS ファイルパス名を持ちます。バインドされていない AF_UNIX データグラムソケットのオープンは、一意の Guardian 形式の名前を持ち、OSS ファイルパス名は持ちません。

バインドされているデータグラムソケットの OSS オープンを選択するには、そのバインドされている OSS ファイルパス名を使用します。

```
LIST FILE "/a/b/socketf"
```

OSS の OSS AF_INET および AF_INET6 ソケットのオープン

AF_INET および IF_INET6 ソケットのオープンを選択するには、Guardian 形式の名前をマスクするか、Measure ファイルオープンタイプを使用します。

```
LIST FILE $ZTC0.*.*

LIST FILE * (OSSINETSOCKET)

LIST FILE * (OSSINETSTREAM)

LIST FILE * (OSSINETDGRAM)
```

OSS の OSS AF_INET および AF_INET6 ストリームソケットのオープン

（バインドされている同じソケットアドレスを使用する）ソケット接続に関係する AF_INET および AF_INET6 ストリームソケットの OSS オープンはすべて、リスニングソケットの IP アドレスを共有します。これには、bind()、connect()、および accept()呼び出しに関連するソケットが含まれます。リスニングソケットの IP アドレスが 0.0.0.0 である（複数の IP アドレスにバインドされている）場合、許容されるソケットの IP アドレスはそれ自体のソケットアドレスです。

ソケット接続に関係する AF_INET および AF_INET6 ストリームソケットの OSS オープンは、一意の Guardian 形式の名前を持ちます。ただし、許容される AF_INET または AF_INET6 ストリームソケットは、その Guardian 形式の名前をリスニング AF_INET または AF_INET6 ストリームソケットと共有します。

同じ AF_INET または AF_INET6 ソケットストリーム接続の OSS ソケットオープンを選択するには、リスニングソケットの IP アドレスを使用します。リスニングソケットの IP アドレスが 0.0.0.0 である（複数の IP アドレスにバインドされている）場合、許容されるソケットの IP アドレスはそれ自体のソケットアドレスです。

```
LIST FILE $ZTC0.*.*, if IP-ADDR = "1.2.3.4 555"

LIST FILE *, if IP-ADDR = "1.2.3.4 *"   IP address = 1.2.3.4, any port number

LIST FILE *, if IP-ADDR = "* 37"        IP address = any IP address, port 37

LIST FILE *, if IP-ADDR = "1.2.3.4 37"  IP address = 1.2.3.4, port 37

LIST FILE *, if IP-ADDR = "1070::800:2006:4178 37"  IPv6 address, port 37
```

OSS の OSS AF_INET および AF_INET6 データグラムソケットのオープン

バインドされている AF_INET または AF_INET6 データグラムソケットのオープンは、一意の Guardian 形式の名前を持ち、ソケットアドレスを IP アドレスとして使用します。バインドされていない AF_INET または AF_INET6 データグラムソケットのオープンは、一意の Guardian 形式の名前を持ちますが、IP アドレスはありません。

OSS AF_INET または AF_INET6 データグラムソケットの OSS オープンは、そのバインドされているソケットアドレスを使用して選択できます。

```
LIST FILE *, if IP-ADDR = "1.2.3.4 *"   IP address = 1.2.3.4, any port number

LIST FILE *, if IP-ADDR = "* 37"        IP address = any IP address, port 37

LIST FILE *, if IP-ADDR = "1.2.3.4 37"  IP address = 1.2.3.4, port 37

LIST FILE *, if IP-ADDR = "1070::800:2006:4178 37"  IPv6 address, port 37
```

IPv6 アドレスについて詳しくは、*TCP/IPv6 Configuration and Management Manual* を参照してください。

例

FILE エンティティのレポート出力には、ANSI SQL 名を持つファイルについての ANSI SQL 名が含まれています。例を示します。

```
File Open $DATA.ZSD12345.Z1234567                    Open Type
ANSI SQL Name 'TABLE Catalog_12.Schema_45.Table_56'
Device Name                  Device Type  3  (Disk)
Opener 0,285                 File Num
Program
Opener Device Name $SYSTEM
Format Version:  H01  Data Version:  H01  Subsystem Version:  1
Local System \HURTS   From  12 Jan 2003, 19:35:28   For  46.8 Seconds
------------ Requests ------------------------------------------------------
File-Busy-Time                       Disc-Reads
Reads                                Writes
Deletes-or-Writereads                Updates-or-Replies
Timeouts-Or-Cancels                  Info-Calls
Misc-Calls
```

```
------------ Logical I/O ----------------------------------------------------
Messages                          15 #    Message-Bytes               47,558 #
Read-Bytes                                Write-Bytes
DBIO-Reads                                DBIO-Writes
DBIO-Read-Bytes                           DBIO-Write-Bytes
Lock-Waits                                Escalations
------------ SQL ------------------------------------------------------------
Records-Accessed                          Records-Used
------------ Open System Services -------------------------------------------
OSS-Cache-Reads                           OSS-Cache-Writes
OSS-Cache-Read-Bytes                      OSS-Cache-Write-Bytes
OSS-Block-Reads                           OSS-Flow-Controls
OSS-Block-Read-Bytes
```

LINE

LINE エンティティは、1 つまたは複数のネットワーク通信回線に関する情報を提供します。

トピック

LINE エンティティのエンティティ指定構文

LINE エンティティの DDL レコード

LINE エンティティの DDL フィールドの定義

LINE エンティティの使用上の注意

LINE エンティティのエンティティ指定構文

LINE エンティティを記述するには、次の構文を使用します。

注記:

アプリケーションがすべての LINE エンティティを指定するか（ADD LINE *）、*$device* または *$device (cpu)*のみを指定する場合、D シリーズの測定アプリケーションを使用して G シリーズの LINE エンティティを測定できます。アプリケーションが *controller*、*channel*、または *unit* を指定している場合は、G シリーズの LINE エンティティを測定するようにエンティティ識別子を変更する必要があります。

```
LINE entity-spec [,entity-spec] ...
```

LINE

1 つまたは複数の通信回線に関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                                                                    }
{ $device                                                              }
{ cpu [ ( %Htrackid [, clip [, line ] ] ) ] [ ( type [, subtype ] ) ] }
```

また、D シリーズのデータのみの場合は、次のように指定します。

```
{ $device [ ( cpu [ , channel [ , ctrl [ , unit ] ] ] ) ]              }
```

指定する値は次の通りです。

*****

すべての CPU のすべての通信回線を測定します。

$*device*

測定対象の回線のデバイス名です。すべての回線を指定するには、アスタリスク（*）を使用します。

cpu

測定対象の回線が構成されている CPU の番号です。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

%Htrackid

（G シリーズ）3 つの CLIP の特定の SWAN コンセントレーターのエンクロージャーの識別子です。（各 CLIP は 2 回線を制御します。）すべての TRACKID を指定するには、アスタリスク（*）を使用します。

TRACKID は、コントローラーの SEEPROM に記録されます。TRACKID は通常はエンクロージャーの外部のラベルにも印字されています。

clip

（G シリーズ）SWAN コンセントレーターのエンクロージャー内の特定の CLIP を示す場合は 1、2、または 3、SWAN 2 コンセントレーターのエンクロージャー内の特定の CLIP を示す場合は 1、2、3、4、5、または 6 です。すべての CLIP を指定するには、アスタリスク（*）を使用します。

line

（G シリーズ）SWAN コンセントレーターの CLIP によって管理される特定の回線を示す場合は、0 または 1 です。すべての回線を指定するには、アスタリスク（*）を使用します。

type

（G シリーズ）特定の回線タイプです。（*type* の有効な値を一覧表示するには、SCF LISTDEV を使用します。）*type* を指定すると、タイプ 61（X25AM）などの特定のプロトコルの回線のみを選択します。すべてのタイプを指定するには、アスタリスク（*）を使用します。デフォルトはすべてのタイプです。

subtype

（G シリーズ）回線タイプについて詳しく説明します。（*subtype* の有効な値を表示するには、SCF LISTDEV を使用します。）*subtype* はサブシステムに固有の値です。デフォルトは、*type* に適用されるすべてのサブタイプです。

channel

（D シリーズ）測定対象の回線のチャネル番号です。デフォルトは、すべてのチャネルです。

ctrl

（D シリーズ）測定対象の回線のコントローラー番号です。デフォルトは、すべてのコントローラーです。

unit

（D シリーズ）測定対象の回線のユニット番号です。デフォルトは、すべてのユニットです。

LINE エンティティの DDL レコード（ZMS 形式）

LINE エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsline-id.
 02 device-name                 type character 8.
 02 logical-device              type binary 32 unsigned.
 02 device-type                 type binary 16 unsigned.
 02 device-subtype              type binary 16 unsigned.
 02 pin                         type binary 16 unsigned.
 02 trackid                     type character 6.
 02 clip                        type binary 16 unsigned.
 02 line                        type binary 16 unsigned.
 02 reserved-1                  type character 4.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsline-ctrs.
 02 read-busy-time              type binary 64.
 02 write-busy-time             type binary 64.
 02 requests                    type binary 64.
 02 input-bytes                 type binary 64.
 02 output-bytes                type binary 64.
 02 input-data-bytes            type binary 64.
 02 output-data-bytes           type binary 64.
 02 reads                       type binary 64.
 02 writes                      type binary 64.
 02 retries                     type binary 64.
 02 transactions                type binary 64.
 02 response-time               type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsline. FILE is "zmsline" ENTRY-SEQUENCED.
 02 hdr                         type zmsheader.
 02 ctr                         type zmsline-ctrs.
 02 id                          type zmsline-id.
end
```

LINE エンティティの DDL レコード（Legacy 形式）

LINE エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G シリーズ LINE エンティティの DDL レコードは、次の点を除いて D シリーズ LINE エンティティのレコードと同じです。

- 別のエンティティ識別フィールドが使用されます。CTRL フィールド、UNIT フィールド、および CHANNEL フィールドは使用されなくなります。
- フィールドのオーバーフローの可能性を減らすために、長い byte-count フィールドが提供されます。G シリーズの RVU では、32 ビットの各 byte-count フィールドに対応する 64 ビットがあります。64

ビットのフィールドの使用については、**G シリーズの LINE エンティティの使用上の注意**(317 ページ)を参照してください

。H シリーズ以降の RVU では、すべての byte-count フィールドが 64 ビットに対応しています。

- ERROR フィールドが、32 ビットの byte-count フィールドのフィールドオーバーフローを通知することがあります。

```
RECORD line. FILE is "line" ENTRY-SEQUENCED.
                      .
                      .
                      .
(error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)
                      .
                      .
                      .
* entity identification items:
 02 pin                          type binary 16 unsigned.
 02 device-name                  type character 8.
 02 logical-device               type binary 16 unsigned.
 02 ctrl                         type binary 16 unsigned.
 02 unit                         type binary 16 unsigned.
 02 device-type                  type binary 16 unsigned.
 02 device-subtype               type binary 16 unsigned.
* counter value items:
 02 read-busy-time               type binary 64.
 02 write-busy-time              type binary 64.
 02 requests                     type binary 32 unsigned.
 02 input-bytes                  type binary 32 unsigned.
 02 output-bytes                 type binary 32 unsigned.
 02 input-data-bytes             type binary 32 unsigned.
 02 output-data-bytes            type binary 32 unsigned.
 02 reads                        type binary 32 unsigned.
 02 writes                       type binary 32 unsigned.
 02 retries                      type binary 32 unsigned.
 02 transactions                 type binary 32 unsigned.
 02 response-time                type binary 64.
* New entity identification item for D10:
 02 channel                      type binary 16 unsigned.
* F40 new entity identification items:
 02 trackid                      type character 6.
 02 clip                         type binary 16 unsigned.
 02 line                         type binary 16 unsigned.
* F40 new counter value items:
 02 input-bytes-f                type binary 64.
 02 output-bytes-f               type binary 64.
 02 input-data-bytes-f           type binary 64.
 02 output-data-bytes-f          type binary 64.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

LINE エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。

CLIP

報告された測定間隔の最後にこのプロセッサーの新しいカウンターレコード用に使用可能なカウンター ID 制御ブロックの数。

DEVICE-NAME

測定対象の回線のデバイス名。

DEVICE-SUBTYPE

DEVICE-TYPE の追加の識別子。通信回線のサブタイプ値のリストについては、*System Generation Manual* を参照してください。

DEVICE-TYPE

デバイスタイプ（たとえば ATP6100 ラインの場合は 54）。通信回線のデバイスタイプ値のリストについては、*System Generation Manual* を参照してください。

LINE

3880 SWAN コンセントレーター上の CLIP 内の特定の回線。0 または 1 です。

LOGICAL-DEVICE

測定された回線の論理デバイス番号。

PIN

プライマリプロセスのプロセス識別番号。

TRACKID

3880 コントローラーの SEEPROM によって返される、ユニットで外部に表示される 6 バイトの識別子。

LINE エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。

INPUT-BYTES

通信回線から読み取られたバイト数。

SNAXLINK 回線の場合、INPUT-BYTES は SNAXLINK 読み取りユニットから受信したバイトの合計です。INPUT-BYTES には、要求/応答ヘッダー（RH）、要求/応答ユニット（RU）、送信ヘッダー（TH）、および HDLC フレームのオーバーヘッドが含まれます（3 + RU + 6 +フレームに到達した PIU が 1 つだけだと仮定した場合は約 5 バイト）。

その他のすべての回線については、INPUT-BYTES は読み取りユニットから受信したバイトの合計です。INPUT-BYTES には、RH + RU + TH + SDLC フレームのオーバーヘッド（3 + RU + 6 + 6 バイト）が含まれます。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、INPUT-BYTES-F フィールドが INPUT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

INPUT-DATA-BYTES

通信回線から読み取られたデータバイト数。このカウンターには、ユーザーデータだけが含まれます。回線プロトコルで要求されたデータは無視されます。

すべての INPUT-DATA-BYTES イベントは回線の書き込みユニットに記録されます。
INPUT-DATA-BYTES は常に INPUT-BYTES 数より少なくなります。

SNAX 回線については、このカウンターには、要求/応答ヘッダー（RH）と、パススルーセッションで OPEN 論理ユニットまたは論理ユニットに宛てられた RU の RU データ数（3 + RU バイト）だけが含まれます。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、INPUT-DATA-BYTES-F フィールドが INPUT-DATA-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

OUTPUT-BYTES

通信回線に書き込まれたバイト数。

SNAXLINK 回線の場合、OUTPUT-BYTES は SNAXLINK 書き込みユニットで送信されたバイトの合計です。OUTPUT-BYTES には、要求/応答ヘッダー（RH）、要求/応答ユニット（RU）、送信ヘッダー（TH）、および HDLC フレームのオーバーヘッドが含まれます（3 + RU + 6 +フレームに PIU を 1 つだけ送信されたと仮定した場合は約 6 バイト）。

PIU は、ブロック化された形式で SNAXLINK ユニットに送信できます。このため、オーバーヘッドを正確に計算することはできません。

他のすべての回線の場合、OUTPUT-BYTES は書き込みユニットで送信されたバイトの合計です。
OUTPUT-BYTES には、RH + RU + TH + SDLC フレームのオーバーヘッド（3 + RU + 6 + 6 バイト）が含まれます。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、OUTPUT-BYTES-F フィールドが OUTPUT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

OUTPUT-DATA-BYTES

通信回線に書き込まれたデータバイト数。このカウンターには、ユーザーデータだけが含まれます。回線プロトコルで要求されたデータは無視されます。

OUTPUT-DATA-BYTES カウントは常に OUTPUT-BYTES カウントより少なくなります。

SNAX 回線の場合、このカウンターには、回線に送信された要求/応答ヘッダー（RH）と RU データカウント（3 + RU バイト）のみが含まれます。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、OUTPUT-DATA-BYTES-F フィールドが OUTPUT-DATA-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

READ-BUSY-TIME

通信回線からの読み取りに費やした時間。このカウンターは、読み取りを開始する命令から I/O 操作の終了を示す割り込みまでの時間を測定します。この時間の間には、どのプロセスでも実行できます。

SNAX 回線と AM6520 回線では、ポーリングプロトコルには常に処理待ちの読み取りがあるため、このカウンターは常にほぼ 100%ビジーです。

カウンタータイプ：ビジー。

READS

I/O プロセスによって実行された（通信回線からメモリへの）読み取り操作の数。I/O プロセスは I/O 操作の前にこのカウンターを変更するため、このカウンターには成功と失敗の両方の操作が含まれます。RETRIES カウンターは、障害による I/O 操作の再試行回数をカウントします。

カウンタータイプ：増分。

REQUESTS

通信回線に対する I/O プロセスが受信した要求の数。I/O プロセスが読み取るまで要求がキューに入れられていた時間を確認するには、I/O プロセスを測定し、PROCESS エンティティの RECV-QTIME カウンターを調べます。要求を読み取った後、I/O プロセスはその要求を内部キューに入れます。

カウンタータイプ：増分。

RESPONSE-TIME

この回線上のすべての測定対象のサブデバイスに対して、I/O プロセスが端末応答で費やした時間。トランザクションとは、端末からの読み取りと、その直後の端末への書き込みを表します。端末応答とは、読み取りが終わってから書き込みが始まるまでの間隔を表します。TRANSACTIONS はトランザクションの数をカウントします。

SNAX 回線では、このカウンターは常にゼロです。

カウンタータイプ：応答時間。

RETRIES

通信回線のエラーのために I/O プロセスが I/O 操作を再試行した回数。

カウンタータイプ：増分。

TRANSACTIONS

この回線上にあるすべての測定対象サブデバイスの I/O プロセスによって実行された端末トランザクションの数。トランザクションとは、端末からの読み取りと、その直後の端末への書き込みを表します。端末応答とは、読み取りが終わってから書き込みが始まるまでの間隔を表します。RESPONSE-TIME は、トランザクションで費やした時間を測定します。

SNAX 回線では、このカウンターは常にゼロです。

カウンタータイプ：増分。

WRITE-BUSY-TIME

通信回線への書き込みに費やした時間。このカウンターは、書き込み操作を開始する命令から I/O 操作の終了を示す割り込みまでの時間を測定します。この時間の間には、どのプロセスでも実行できます。

カウンタータイプ：ビジー。

WRITES

I/O プロセスによって実行された（メモリから通信回線への）書き込み操作の回数。I/O プロセスは I/O 操作の前にこのカウンターを変更するため、このカウンターには成功と失敗の両方の操作が含まれます。RETRIES カウンターは、障害による I/O 操作の再試行回数をカウントします。

カウンタータイプ：増分。

LINE エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードのみで使用されます。

フィールドはアルファベット順に一覧表示されます。

CHANNEL

（Legacy 形式のみ）D シリーズの RVU で、回線のチャネル番号。G シリーズの RVU では使用されず、ゼロを返します。

CTRL

（Legacy 形式のみ）D シリーズの RVU では、測定された回線のコントローラー番号。G シリーズの RVU では使用されず、ゼロを返します。

UNIT

（Legacy 形式のみ）D シリーズの RVU では、測定対象の回線のユニット番号を表します。G シリーズの RVU では使用されず、ゼロを返します。

LINE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードだけで使用されます。

フィールドはアルファベット順に一覧表示されます。

INPUT-BYTES-F

（G シリーズ、Legacy 形式のみ）INPUT-BYTES と同じですが、より大きな値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

INPUT-DATA-BYTES-F

（G シリーズ、Legacy 形式のみ）INPUT-DATA-BYTES と同じですが、より大きな値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

OUTPUT-BYTES-F

（G シリーズ、Legacy 形式のみ）OUTPUT-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

OUTPUT-DATA-BYTES-F

（G シリーズ、Legacy 形式のみ）OUTPUT-DATA-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

H シリーズ以降の LINE エンティティの使用上の注意

- H シリーズ以降の RVU では、すべての byte-count フィールドが 64 ビットに対応しています。-F で終わるフィールド名は、ZMS 形式のレコードでは使用されなくなりましたが、Legacy 形式でデータを要求するアプリケーションでは引き続き使用できます。
- SUBSYSTEM-VERSION は、最初の H シリーズ以降のバージョンのサブシステムでは 1 に設定されます。

G シリーズの LINE エンティティの使用上の注意

- 64 ビットのバイトカウントフィールド（-F で終わるフィールド）は、従来の 32 ビットのバイトカウントフィールドと同じデータを収集します。たとえば、64 ビットフィールドの INPUT-BYTES-F は、32 ビットフィールドの INPUT-BYTES と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

  32 ビットフィールドは現在アクティブであり、引き続き値を返します フィールドのオーバーフローが存在しない場合、32 ビットフィールドと 64 ビットのフィールドは同じ値を返します。32 ビットフィールドがオーバーフローした場合、対応する 64 ビットフィールドは正しい値を返し、32 ビットフィールドは-1 の値を返します。測定対象のエンティティの ERROR フィールドも、オーバーフロー状態を示す-1 を返します。

  64 ビットフィールドを使用するには、アプリケーションを変換してください。32 ビットのフィールドは、今後の RVU では使用できなくなる可能性があります。

  MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットフィールドの名前を使用してください。たとえば、LIST LINE BY INPUT-BYTES-F ではなく LIST LINE BY INPUT-BYTES のコマンドを発行してください。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。
- ZMSLINE レコードの SUBSYSTEM-VERSION は、各 ZMSLINE レコードによって表される回線（SNAX/XF、X25AM、ENVOY など）の特定の製品サブシステムによって与えられます。G シリーズの RVU では、SUBSYSTEM-VERSION の値はゼロです。
- Measure G09 の PVU では、1 つの SWAN コンセントレーターで最高 6 つの CLIP をサポートします。

NETLINE

NETLINE エンティティは、ネットワーク通信回線に関する情報を提供します。

トピック
NETLINE エンティティのエンティティ指定構文
NETLINE エンティティの DDL レコード
NETLINE エンティティの DDL フィールドの定義
NETLINE エンティティの使用上の注意

NETLINE エンティティのエンティティ指定構文

NETLINE エンティティを記述するには、次の構文を使用します。

```
NETLINE entity-spec [,entity-spec] ...
```

注記:

すべての NETLINE エンティティを指定しているか（ADD NETLINE *）、*$device* または*$device (cpu)* のみを指定している場合、D シリーズの測定アプリケーションで G シリーズのネットワーク通信回線を測定できます。アプリケーションが *controller*、*channel*、または *unit* を指定している場合は、G シリーズ NETLINE エンティティを測定するようにエンティティ識別子を変更する必要があります。

NETLINE

1 つまたは複数のネットワーク通信回線に関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                                                                          }
{ $device                                                                    }
{ cpu [ ( %Htrackid [, clip [, line ] ] ) ] [ ( type [, subtype ] ) ] }
```

また、D シリーズのデータのみの場合は、次のように指定します。

```
{ $device [ ( cpu [ , channel [ , ctrl [ , unit ] ] ] ) ]              }
```

指定する値は次の通りです。

*****

すべての CPU のすべてのネットワーク通信回線を測定します。

$*device*

測定対象の回線のデバイス名です。すべてのネットワーク回線を指定するには、アスタリスク（*）を使用します。

cpu

測定する回線が構成されている CPU の番号です。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

%Htrackid

3 つの CLIP の特定の 3880 SWAN コンセントレーターのエンクロージャーの識別子です。（各 CLIP は 2 回線を制御します。）すべての TRACKID を指定するには、アスタリスク（*）を使用します。

TRACKID は、コントローラーの SEEPROM に記録されます。通常はエンクロージャーの外部のラベルにも印字されています。

clip

SWAN コンセントレーターのエンクロージャー内の特定の CLIP を示す場合は 1、2、または 3、SWAN 2 コンセントレーターのエンクロージャー内の特定の CLIP を示す場合は 1、2、3、4、5、または 6 です。すべての CLIP を指定するには、アスタリスク（*）を使用します。

line

3880 SWAN コンセントレーターの CLIP によって管理される特定の回線を示す場合は、0 または 1 です。すべての回線を指定するには、アスタリスク（*）を使用します。

type

特定のプロトコルを示す 63 などのタイプ番号です（Expand 回線ハンドラー）。回線タイプの有効な値の一覧を表示するには、SCF LISTDEV を使用します。すべての回線タイプを指定するには、アスタリスク（*）を使用します。

subtype

3 などのサブタイプ番号です(FOX 回線ハンドラー)。有効なサブタイプ番号を確認するには、SCF LISTDEV を使用します。デフォルトは、*type* に適用されるすべてのサブタイプです。

channel

（D シリーズ）測定対象の回線のチャネル番号です。デフォルトは、すべてのチャネルです。

ctrl

（D シリーズ）測定対象の回線のコントローラー番号です。デフォルトは、すべてのコントローラーです。

unit

（D シリーズ）測定対象の回線のユニット番号です。デフォルトは、すべてのユニットです。

NETLINE エンティティの DDL レコード（ZMS 形式）

NETLINE エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsnline-id.
 02 device-name                 type character 8.
 02 logical-device              type binary 32 unsigned.
 02 device-type                 type binary 16 unsigned.
 02 device-subtype              type binary 16 unsigned.
 02 pin                         type binary 16 unsigned.
 02 trackid                     type character 6.
 02 clip                        type binary 16 unsigned.
 02 line                        type binary 16 unsigned.
 02 reserved-1                  type character 4.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsnline-ctrs.
 02 write-busy-time             type binary 64.
 02 read-busy-time              type binary 64.
 02 requests                    type binary 64.
 02 reads                       type binary 64.
 02 writes                      type binary 64.
 02 l2in-bytes                  type binary 64.
 02 l2out-bytes                 type binary 64.
 02 din4-bytes                  type binary 64.
 02 dout4-bytes                 type binary 64.
 02 cin4-bytes                  type binary 64.
 02 cout4-bytes                 type binary 64.
 02 u64-bytes                   type binary 64.
 02 u128-bytes                  type binary 64.
 02 u256-bytes                  type binary 64.
 02 u512-bytes                  type binary 64.
 02 u1024-bytes                 type binary 64.
 02 u2048-bytes                 type binary 64.
 02 u4096-bytes                 type binary 64.
```

```
 02 o4095-bytes                  type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsnline. FILE is "zmsnline" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 ctr                          type zmsnline-ctrs.
 02 id                           type zmsnline-id.
end
```

NETLINE エンティティの DDL レコード（Legacy 形式）

NETLINE エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G シリーズ NETLINE エンティティの DDL レコードは、次の点を除いて D シリーズ NETLINE エンティティのレコードと同じです。

- 別のエンティティ識別フィールドが使用されます。CHANNEL フィールド、CTRL フィールド、および UNIT フィールドは使用されなくなります。
- フィールドのオーバーフローを防ぐために、長い byte-count フィールドが提供されます。32 ビットの各 byte-count フィールドには、対応する 64 ビットがあります。64 ビットのフィールドの使用については、**G シリーズの NETLINE エンティティの使用上の注意**(326 ページ)を参照してください。
- ERROR フィールドが、32 ビットの byte-count フィールドのフィールドオーバーフローを通知することがあります。

```
RECORD netline. FILE is "netline" ENTRY-SEQUENCED.
                        .
                        .
                        .
```

(error、time items、および measurement identification items については、**一般的なエンティティヘッダーフィールド**(177 ページ) を参照してください)

```
                        .
                        .
                        .
* entity identification items:
 02 pin                          type binary 16 unsigned.
 02 device-name                  type character 8.
 02 logical-device               type binary 16 unsigned.
 02 ctrl                         type binary 16 unsigned.
 02 unit                         type binary 16 unsigned.
 02 device-type                  type binary 16 unsigned.
 02 device-subtype               type binary 16 unsigned.
* counter value items:
 02 write-busy-time              type binary 64.
 02 read-busy-time               type binary 64.
 02 requests                     type binary 32 unsigned.
 02 reads                        type binary 32 unsigned.
 02 writes                       type binary 32 unsigned.
 02 l2in-bytes                   type binary 32 unsigned.
 02 l2out-bytes                  type binary 32 unsigned.
 02 din4-bytes                   type binary 32 unsigned.
 02 dout4-bytes                  type binary 32 unsigned.
 02 cin4-bytes                   type binary 32 unsigned.
 02 cout4-bytes                  type binary 32 unsigned.
 02 u64-bytes                    type binary 32 unsigned.
```

```
 02 u128-bytes                 type binary 32 unsigned.
 02 u256-bytes                 type binary 32 unsigned.
 02 u512-bytes                 type binary 32 unsigned.
 02 u1024-bytes                type binary 32 unsigned.
 02 u2048-bytes                type binary 32 unsigned.
 02 u4096-bytes                type binary 32 unsigned.
 02 o4095-bytes                type binary 32 unsigned.
* new entity identification item for D10:
 02 channel                     type binary 16 unsigned.
* F40 new entity identification items:
 02 TrackID                     type character 6.
 02 Clip                        type binary 16 unsigned.
 02 Line                        type binary 16 unsigned.
* F40 new counter value items:
 02 l2in-bytes-f               type binary 64.
 02 l2out-bytes-f              type binary 64.
 02 din4-bytes-f               type binary 64.
 02 dout4-bytes-f              type binary 64.
 02 cin4-bytes-f               type binary 64.
 02 cout4-bytes-f              type binary 64.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

NETLINE エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。

CLIP

報告された測定間隔の最後にこのプロセッサーの新しいカウンターレコード用に使用可能なカウンター ID 制御ブロックの数。

DEVICE-NAME

測定対象の回線のデバイス名。

DEVICE-SUBTYPE

デバイスタイプに関する追加情報：

0	コントローラーに接続されている単一の回線ハンドラー
1	マルチラインパスハンドラー
2	コントローラーに接続されているマルチラインのライン
3	FOX 回線ハンドラー
5	6100 サブシステムに接続されている単一の回線ハンドラー
6	6100 サブシステムに接続されているマルチラインハンドラー

DEVICE-TYPE

値は 63 で、Expand 回線を示しています。

LINE

SWAN コントローラー上の CLIP 内の特定の回線。0 または 1 です。

LOGICAL-DEVICE

測定された回線の論理デバイス番号。

PIN

プライマリプロセスのプロセス識別番号。

TRACKID

3880 コントローラーの SEEPROM によって返される、エンクロージャーの外部のラベルに示される 6 バイトの識別子。

NETLINE エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧表示されます。

CIN4-BYTES

Expand 回線のネットワーク回線ハンドラーによって読み取られたレベル 4 のプロトコルバイト数および NCP 要求バイト数（ACK、NAK、ENQ、PCHG、NCPM、CONN、および CAN）。

X.25 ではこのカウンターは更新されません。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、CIN4-BYTES の 64 ビットバージョンは CIN4-BYTES-F フィールドです。

カウンタータイプ：累積。

COUT4-BYTES

Expand 回線のネットワーク回線ハンドラーによって書き込まれるレベル 4 プロトコルバイト数および NCP 要求バイト数（ACK、NAK、ENQ、PCHG、NCPM、CONN、および CAN）

X.25 ではこのカウンターは更新されません。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、COUT4-BYTES-F フィールドが COUT4-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

DIN4-BYTES

ネットワーク回線ハンドラーで読み取られたレベル 4 データバイトの数。このカウンターはユーザーデータだけを含み、D シリーズではメッセージシステムプロトコルバイトを含みます。回線プロトコルに関連付けられているデータは無視されます。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、DIN4-BYTES-F フィールドが DIN4-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

DOUT4-BYTES

ネットワーク回線ハンドラーで書き込まれたレベル 4 データバイトの数。このカウンターはユーザーデータだけを含み、D シリーズではメッセージシステムプロトコルバイトを含みます。回線プロトコルに関連付けられているデータは無視されます。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、DOUT4-BYTES-F フィールドが DOUT4-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

L2IN-BYTES

ネットワーク回線ハンドラーで読み取られたレベル 2 バイトの数。このカウンターには、ユーザーデータと、回線プロトコルに関連付けられたデータの両方が含まれます。

非 CSS タイプのコントローラーの場合、このカウンターには、DIN4-BYTES と CIN4-BYTES が含まれます。CSS タイプのコントローラーの場合、このカウンターには CLB（CAP から CLIP）プロトコルと、CLIP TRACE ブロックも含まれます。

ネットワーク回線上のレベル 2 バイトの合計数は、このカウンターに L2OUT-BYTES を加えた数になります。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、L2IN-BYTES-F フィールドが L2IN-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

L2OUT-BYTES

ネットワーク回線ハンドラーで書き込まれたレベル 2 バイトの数。このカウンターには、ユーザーデータと、回線プロトコルに関連付けられたデータの両方が含まれます。

非 CSS タイプのコントローラーの場合、このカウンターには、DOUT4-BYTES と COUT4-BYTES が含まれます。CSS タイプのコントローラーの場合、このカウンターには CLB プロトコルバイトも含まれます。

ネットワーク回線上のレベル 2 バイトの合計数は、このカウンターに L2IN-BYTES を加えた数になります。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RVU では、L2OUT-BYTES-F フィールドが L2OUT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

O4095-BYTES

圧縮後に 4095 バイトを超えたメッセージの数です。サイズにはプロトコルバイトが含まれます。

カウンタータイプ：増分。

READ-BUSY-TIME

ネットワーク通信回線からのデータの読み取りに費やした時間。このカウンターには、回線への入力を待機するのに費やした時間が含まれます。このカウンターは、読み取りを開始する命令から I/O 操作の終了を示す割り込みまでの時間を測定します。

アクティブな全二重回線には、通常は続行の処理待ちの読み取りがあるため、このカウンターには高い値が予想されます。この時間の間には、どのプロセスでも実行できます。

このカウンターは、SWAN 経由の Expand 回線が使用中の場合のみ測定されます。

カウンタータイプ：ビジー。

READS

回線ハンドラープロセスによって実行された（ネットワーク通信回線からメモリへの）読み取り操作の数。

カウンタータイプ：増分。

REQUESTS

このパスが処理したすべての SYSTEM LINKS の合計と、このパスが処理したすべての SYSTEM SENT FORWARD（PASSTHRU REQUESTS）の合計。

LINKS はメッセージレベル番号です。SENT、RECEIVED、SENT FORWARD、および RECEIVE FORWARD はパケットカウンターです。1 つのメッセージが複数のパケットである場合があります。

REQUESTS は、メッセージレベルのイベント（LINKS）とパッケージレベルのイベント（SENT FORWARD）の両方を同じカウンターでカウントします。要求は、この回線ハンドラーが受け取り、リモートシステムに送信されるメッセージクイックセル（MQC）の数です。

カウンタータイプ：増分。

U64-BYTES

圧縮後に 64 バイトより短くなったメッセージの数。サイズにはプロトコルバイトが含まれます。

カウンタータイプ：増分。

U128-BYTES

圧縮後に 64～127 バイトになったメッセージの数。サイズにはプロトコルバイトが含まれます。

カウンタータイプ：増分。

U256-BYTES

圧縮後に 128～255 バイトになったメッセージの数。サイズにはプロトコルバイトが含まれます。

カウンタータイプ：増分。

U512-BYTES

圧縮後に 256～511 バイトになったメッセージの数。サイズにはプロトコルバイトが含まれます。

カウンタータイプ：増分。

U1024-BYTES

圧縮後に 512～1023 バイトになったメッセージの数。サイズにはプロトコルバイトが含まれます。

カウンタータイプ：増分。

U2048-BYTES

圧縮後に 1024～2047 バイトになったメッセージの数。サイズにはプロトコルバイトが含まれます。

カウンタータイプ：増分。

U4096-BYTES

圧縮後に 2048～4095 バイトになったメッセージの数。サイズにはプロトコルバイトが含まれます。

カウンタータイプ：増分。

WRITE-BUSY-TIME

ネットワーク通信回線への書き込みに費やした時間。このカウンターは、書き込み操作を開始する命令から I/O 操作の終了を示す割り込みまでの時間を測定します。この時間の間には、どのプロセスでも実行できます。

このカウンターは、SWAN 経由の Expand 回線が使用中の場合のみ測定されます。

カウンタータイプ：ビジー。

WRITES

回線ハンドラープロセスによって実行された（メモリからネットワーク通信回線への）書き込み操作の回数。

カウンタータイプ：増分。

NETLINE エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードのみで使用されます。

フィールドはアルファベット順に一覧表示されます。

CHANNEL

（Legacy 形式のみ）D シリーズの RVU で、測定された回線のチャネル番号。G シリーズの RVU では使用されず、ゼロを返します。

CTRL

（Legacy 形式のみ）D シリーズの RVU では、測定された回線のコントローラー番号。G シリーズの RVU では使用されず、ゼロを返します。

UNIT

（Legacy 形式のみ）D シリーズの RVU では、測定対象の回線のユニット番号を表します。G シリーズの RVU では使用されず、ゼロを返します。

NETLINE エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードだけで使用されます。

フィールドはアルファベット順に一覧表示されます。

CIN4-BYTES-F

（G シリーズ、Legacy 形式のみ）CIN4-BYTES と同じですが、より大きい値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

COUT4-BYTES-F

（G シリーズ、Legacy 形式のみ）COUT4-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

DIN4-BYTES-F

（G シリーズ、Legacy 形式のみ）DIN4-BYTES と同じですが、より大きい値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

DOUT4-BYTES-F

（G シリーズ、Legacy 形式のみ）DOUT4-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

L2IN-BYTES-F

（G シリーズ、Legacy 形式のみ）L2IN-BYTES と同じですが、より大きな値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

L2OUT-BYTES-F

（G シリーズ、Legacy 形式のみ）L2OUT-BYTES と同じですが、より大きな値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

すべての NETLINE エンティティの使用上の注意

- 一般に、Measure の L2 カウンターによって収集されるデータは、実際の回線のバイトカウントとほぼ一致します。ただし、回線の問題や CLIP のトレースによって Measure のバイトカウントが変動することがあります。

  CSS タイプのコントローラーでは、Measure によって収集される情報はコントローラーから回線ハンドラーへの情報です。回線の問題が原因でデータの再送が発生した場合、Measure のカウンターは加算されません。SCF 回線の統計は、再送を引き起こすイベント（タイムアウトと FCS エラー）をカウントします。回線の問題に関する詳細な情報を取得するには、データスコープを使用します。

  CLIP のトレースを行う場合、トレースデータは L2IN-BYTES カウンターに含まれます。同様に、CIN4-BYTES および COUT4-BYTES カウンターには、場合によっては NCP NETMAP パケットが含まれます。
- ネットワークおよびネットワークプロトコルについて詳しくは、*Expand Configuration and Management Manual*、*Expand Network Management and Troubleshooting Manual*、および *X25AM Configuration and Management Manual* を参照してください。

H シリーズ以降の NETLINE エンティティの使用上の注意

H シリーズ以降の RVU では、すべての byte-count フィールドが 64 ビットに対応しています。-F で終わるフィールド名は、ZMS 形式のレコードでは使用されなくなりましたが、Legacy 形式でデータを要求するアプリケーションでは引き続き使用できます。

G シリーズの NETLINE エンティティの使用上の注意

- ZMSNLINE レコードの SUBSYSTEM-VERSION は Expand によって提供されます。
- 64 ビットのバイトカウントフィールド（-F で終わるフィールド）は、従来の 32 ビットのバイトカウントフィールドと同じデータを収集します。たとえば、64 ビットフィールドの L2IN-BYTES-F は、32 ビットフィールドの L2IN-BYTES と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

32 ビットフィールドは現在アクティブであり、引き続き値を返します フィールドのオーバーフローが存在しない場合、32 ビットフィールドと 64 ビットのフィールドは同じ値を返します。32 ビットフィールドがオーバーフローした場合、対応する 64 ビットフィールドは正しい値を返し、32 ビットフィールドは-1 の値を返します。測定対象のエンティティの ERROR フィールドも、オーバーフロー状態を示す-1 を返します。

64 ビットフィールドを使用するには、アプリケーションを変換してください。32 ビットのフィールドは、今後の RVU では使用できなくなる可能性があります。

MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットフィールドの名前を使用してください。たとえば、LIST LINE BY INPUT-BYTES-F ではなく LIST LINE BY INPUT-BYTES のコマンドを発行してください。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。

Measure G09 以降の PVU では、1 つの SWAN コンセントレーターで最高 6 つの CLIP をサポートします。

OPDISK

OPDISK エンティティは、ローカルシステムの 1 つまたは複数のオプティカルディスクシステムに関する情報を提供します。

オプティカルディスクシステムは、エンクロージャー（よくジュークボックスと呼ばれます）であり、1 つまたは複数のディスクドライブと、セルに格納されている複数のディスクカートリッジが含まれます。各カートリッジには 2 つの面があり、それぞれの面が 1 つのボリュームです。

G シリーズ以降の RVU は、オプティカルディスクをサポートしていません。ただし、Measure 以降の PVU には、今後使用するための OPDISK DDL レコードが含まれています。エンティティをサポートしていない RVU で OPDISK エンティティを指定しても、エラーメッセージが表示されないだけでなく、測定の構成にも影響しません。

D シリーズのデータからのオプティカルディスクのレコードは、フォーマットして表示することができます。

トピック

OPDISK エンティティのエンティティ指定構文

OPDISK エンティティの DDL レコード

OPDISK エンティティの DDL フィールドの定義

OPDISK エンティティの使用上の注意

OPDISK エンティティのエンティティ指定構文

D シリーズの RVU の OPDISK エンティティを記述するには、次の構文を使用します。

```
OPDISK entity-spec [,entity-spec] ...
```

OPDISK

ローカルシステム上の 1 つまたは複数のオプティカルディスクに関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                                                              }
{ $device [ $vol ] [ ( cpu [ , chan [ , ctrl [ , unit] ] ] ) ] }
```

指定する値は次の通りです。

*****

システム上のすべてのオプティカルディスクを測定します。

$*device*

測定対象のオプティカルディスクのデバイス名です。すべてのデバイスを指定するには、アスタリスク（*）を使用します。

$*vol*

測定対象のオプティカルディスクボリュームの名前です。すべてのボリュームを指定するには、アスタリスク（*）を使用します。

cpu

オプティカルディスクが構成されている CPU の番号です。デフォルトは、すべての CPU です。

chan

測定対象のオプティカルディスクのチャネル番号です。デフォルト値は、すべてのチャネルです。

ctrl

測定対象のオプティカルディスクのコントローラー番号（0～31）です。デフォルトは、すべてのコントローラーです。

unit

測定対象のオプティカルディスクのユニット番号です。デフォルトは、すべてのユニットです。

OPDISK エンティティの DDL レコード（ZMS 形式）

OPDISK エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsodisk-id.
 02 pin                          type binary 16 unsigned.
 02 device-type                  type binary 16 unsigned.
 02 device-subtype               type binary 16 unsigned.
 02 servernet                    type binary 16 unsigned.
 02 device-name                  type character 8.
 02 logical-device               type binary 32 unsigned.
 02 GMS.
    03 group                     type binary 32 unsigned.
    03 module                    type binary 32 unsigned.
    03 slot                      type binary 32 unsigned.
 02 SCSI-id                      type binary 64.
 02 config-name                  type character 64.
 02 adapter-name                 type character 64.
 02 SAC-name                     type character 64.
 02 volume-name                  type character 8.
 02 cell                         type binary 16 unsigned.
 02 side                         type binary 16 unsigned.
 02 reserved-1                   type character 4.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsodisk-ctrs.
 02 request-qtime                type binary 64.
 02 requests                     type binary 64.
 02 reads                        type binary 64.
 02 writes                       type binary 64.
 02 input-bytes                  type binary 64.
 02 output-bytes                 type binary 64.
 02 read-qbusy-time              type binary 64.
 02 read-qtime                   type binary 64.
 02 write-qbusy-time             type binary 64.
 02 write-qtime                  type binary 64.
 02 device-qbusy-time            type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsodisk. FILE is "zmsodisk" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 ctr                          type zmsodisk-ctrs.
 02 id                           type zmsodisk-id.
end
```

OPDISK エンティティの DDL レコード（Legacy 形式）

OPDISK エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD opdisk.  FILE is "opdisk" ENTRY-SEQUENCED.
                      .
                      .
                      .
```

(error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)

```
                      .
                      .
                      .
* entity identification items:
 02 pin                          type binary 16 unsigned.
 02 device-name                 type character 8.
 02 logical-device              type binary 16 unsigned.
 02 ctrl                         type binary 16 unsigned.
 02 unit                         type binary 16 unsigned.
 02 device-type                 type binary 16 unsigned.
 02 device-subtype              type binary 16 unsigned.
 02 volume-name                 type character 8.
 02 cell                         type binary 16 unsigned.
 02 side                         type binary 16 unsigned.
* counter value items:
 02 request-qtime               type binary 64.
 02 request-qlen-max            type binary 16 unsigned.
 02 requests                     type binary 32 unsigned.
 02 read-busy-time              type binary 64.
 02 write-busy-time             type binary 64.
 02 seek-busy-time              type binary 64.
 02 reads                        type binary 32 unsigned.
```

```
 02 writes                        type binary 32 unsigned.
 02 seeks                         type binary 32 unsigned.
02 input-bytes                   type binary 32 unsigned.
 02 output-bytes                  type binary 32 unsigned.
* new entity identification item for D10:
 02 channel                       type binary 16 unsigned.
 02 servernet                     type binary 16 unsigned
                                  redefines channel.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

OPDISK エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

ADAPTER-NAME

(ZMS 形式のみ) SAC が存在するアダプターに関連付けられた論理名です。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

CELL

オプティカルディスクエンクロージャー内のストレージセルの数。

CONFIG-NAME

(ZMS 形式のみ) この物理ディスクに関連付けられた論理名です。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

DEVICE-NAME

オプティカルディスクエンクロージャーの名前。

DEVICE-SUBTYPE

DEVICE-TYPE の追加の識別子。オプティカルディスクのサブタイプ値の完全なリストについては、*System Generation Manual* を参照してください。

DEVICE-TYPE

値は 30 (オプティカルディスクデバイス)。

GMS

(ZMS 形式のみ) 物理位置アドレス (グループ、モジュール、スロット)。GMS フィールドは、次の 3 つのサブフィールドに分かれています。

GROUP	グループ番号です。
MODULE	モジュール番号です。
SLOT	スロット番号です。

GROUP、MODULE、および SLOT のキーワードは、LIST コマンドおよび LISTALL コマンドの IF 句と BY 句で使用できます。

LOGICAL-DEVICE

光ディスクエンクロージャーの論理デバイス番号。

PIN

オプティカルディスクプロセスのプロセス識別番号。

SAC-NAME

（ZMS 形式のみ）物理ボリュームへのこのパスで使用される SAC に関連付けられている論理名。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

SCSI-ID

（ZMS 形式のみ）このオプティカルディスクの SCSI ポート識別子。

SERVERNET

（G シリーズ以降の RVU）オプティカルディスクへのこの特定のパスで使用される ServerNet ファブリック。Measure サブシステムは、指定されたファブリックに相対的なすべてのオプティカルディスクレコードを測定します。値は、X ファブリックでは 0、Y ファブリックでは 1 です。デフォルトは、両方のファブリックのレコードを測定することです。

ServerNet/DA で接続されたオプティカルディスクには、X または Y ファブリックはありません。これらのディスクを指定するには、アスタリスク（*）を使用します。この場合、エンティティ指定の残りの部分が 1 つのディスクを記述している場合に限り、LISTACTIVE コマンドでアスタリスク（*）を使用することができます。

FCSA または CLIM によって接続されているオプティカルディスクはデュアルファブリックのアダプターです。つまり、X と Y の両方のファブリックに接続されています。これらのディスクの場合、LIST コマンドでは「*」または「X」のみを指定し、LISTACTIVE コマンドでは「X」のみを指定する必要があります。

SIDE

オプティカルディスクカートリッジの面を識別する番号。

VOLUME-NAME

光ディスクボリュームの名前。光ディスクボリュームは、光ディスクカートリッジの片面です。

OPDISK エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧表示されます。サブフィールドは、フィールドに指定する順に一覧表示されます。

DEVICE-QBUSY-TIME

（ZMS 形式のみ）任意の数またはタイプの要求が、このプロセッサーのオプティカルディスクのキューに入れられた状態で経過した時間。

カウンタータイプ：キュービジー。

INPUT-BYTES

光ディスクから読み取られたバイト数。プログラムによる読み取り操作だけでなく、ファイルラベルの読み取りなどの内部操作もこのカウンターを変更します。

I/O 操作前に光ディスクプロセスがこのカウンターを変更するので、読み取りが失敗した場合、このカウンターが正確でなくなる場合があります。

カウンタータイプ：累積。

OUTPUT-BYTES

オプティカルディスクに書き込まれたバイト数。プログラムによる書き込み操作だけでなく、ボリュームラベルの書き込みなどの内部操作もこのカウンターを変更します。

オプティカルディスクプロセスは I/O 操作の前にこのカウンターを変更するため、書き込みが失敗するとこのカウンターは正確ではなくなることがあります。

カウンタータイプ：累積。

READ-QBUSY-TIME

（ZMS 形式のみ）任意の数の読み取り要求が、このプロセッサーのオプティカルディスクのキューに入れられた状態で経過した時間。

カウンタータイプ：キュービジー。

READ-QTIME

（ZMS 形式のみ）オプティカルディスクのキューに入れられた読み取り要求が費やした合計時間。

カウンタータイプ：キュー。

READS

オプティカルディスクプロセスによって実行されたデバイスからメモリへの読み取り操作の数。プログラムによる読み取り操作だけでなく、ファイルラベルの読み取りなどの内部操作もこのカウンターを変更します。

カウンタータイプ：増分。

REQUEST-QTIME

要求がオプティカルディスクプロセスの内部キューで費やした時間（現在アクティブな要求を含む）。

要求を読み取るとき、オプティカルディスクプロセスはその要求を内部キューに入れます。要求を処理した後、オプティカルディスクプロセスは内部キューから要求を削除します。オプティカルディスクプロセスの外部キュー（オプティカルディスクプロセスによって読み取られるのを待機している要求）の長さを確認するには、オプティカルディスクプロセスを測定し、PROCESS エンティティの RECV-QTIME カウンターを調べます。

カウンタータイプ：キュー。

REQUESTS

オプティカルディスクプロセスが受信した要求の数。

オプティカルディスクプロセスが読み取るまで要求がキューに入れられていた時間を確認するには、オプティカルディスクプロセスを測定し、PROCESS エンティティの RECV-QTIME カウンターを調べます。

カウンタータイプ：増分。

WRITE-QBUSY-TIME

（ZMS 形式のみ）任意の数の書き込み要求が、このプロセッサーの光ディスクのキューに入れられた状態で経過した時間。

カウンタータイプ：キュービジー。

WRITE-QTIME

（ZMS 形式のみ）この光ディスクのキューに入れられた書き込み要求が費やした合計時間。

カウンタータイプ：キュー。

WRITES

光ディスクプロセスによって実行されたメモリからデバイスへの書き込み操作の回数。プログラムによる書き込み操作だけでなく、ボリュームラベルの書き込みなどの内部操作もこのカウンターを変更します。

カウンタータイプ：増分。

OPDISK エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードのみで使用されます。

フィールドはアルファベット順に一覧表示されます。

CHANNEL

（Legacy 形式のみ）デバイスのチャネル番号。

CTRL

（Legacy 形式のみ）デバイスのコントローラー番号。

UNIT

（Legacy 形式のみ）デバイスのユニット番号。

OPDISK エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードだけで使用されます。

フィールドはアルファベット順に一覧表示されます。

READ-BUSY-TIME

（Legacy 形式のみ）ディスクからの読み取りに費やした時間。このカウンターは、読み取り操作を開始する命令から操作を終了する割り込みまでの時間を測定します。これには、データの読み取りと、データを読み取るためのディスクヘッドの位置決めに費やした時間が含まれます。

カウンタータイプ：ビジー。

REQUEST-QLEN-MAX

（Legacy 形式のみ）REQUEST-QTIME カウンターで記述されるキューの項目の最大数。

カウンタータイプ：最大キュー。

SEEK-BUSY-TIME

（Legacy 形式のみ）現在は使用されていません。ゼロを返します。読み取り操作と書き込み操作では、すべてのシーク操作は暗黙的です。

SEEKS

（Legacy 形式のみ）現在は使用されていません。ゼロを返します。読み取り操作と書き込み操作では、すべてのシーク操作は暗黙的です。

WRITE-BUSY-TIME

（Legacy 形式のみ）ディスクへの書き込みに費やした時間。このカウンターは、書き込み操作を開始する命令から操作を終了する割り込みまでの時間を測定します。このカウンターには、データの書き込みと、データを書き込むためのディスクヘッドの位置決めに費やした時間が含まれます。

カウンタータイプ：ビジー。

すべての OPDISK エンティティの使用上の注意

- OPDISK は、REPORT ZERO-REPORTS 属性が INCLUDE に設定されている場合でも、アクティビティのなかった光ディスクボリュームは測定しません。
- ZMSODISK レコードの SUBSYSTEM-VERSION はゼロとして報告されます。
- OPDISK レポートには、OPDISK カウンターから計算されるフィールドが含まれます。これらのフィールドは、OPDISK DDL レコードでは定義されていません。

  DEVICE-AST

  （ZMS 形式のみ）各要求が待機または DP2 による処理に費やした平均時間。DEVICE-QBUSY-TIME は READS と WRITES の合計で除算されます。

  カウンタータイプ：サービス時間。

OSSCPU

OSSCPU エンティティは、システムの各プロセッサーで動作する次のようなオープンシステムサービスの要素に関する情報を提供します。

- POSIX 拡張セグメント（PXS）
- OSS ファイルシステムキャッシュ
- File Manager
- パイププール
- パイプサーバー
- AF_UNIX ソケットのローカルサーバー 2（LS2）

OSSCPU エンティティは、Measure G10 以降の PVU を実行しているシステムでのみ使用できます。

トピック
OSSCPU エンティティのエンティティ指定構文
OSSCPU エンティティの DDL レコード
OSSCPU エンティティの DDL フィールドの定義
OSSCPU エンティティの使用上の注意

OSSCPU エンティティのエンティティ指定構文

OSSCPU エンティティを記述するには、次の構文を使用します。

```
OSSCPU entity-spec [,entity-spec] ...
```

entity-spec

次のように指定します。

```
{ * | cpu }
```

指定する値は次の通りです。

****

ローカルシステム上のすべてのプロセスの OSS 要素を測定します。

cpu

測定対象の OSS 要素が存在する CPU です。複数の CPU 番号を指定するには、カンマ区切りのリストを使用します。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

OSSCPU エンティティの DDL レコード（ZMS 形式）

OSSCPU エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsosscp-id.
 02 PXS-bytes-allowed              type binary 64.
 02 Cache-block-size               type binary 32 unsigned.
 02 Cache-blocks-allowed           type binary 32 unsigned.
 02 PP-block-size                  type binary 32 unsigned.
 02 PP-blocks-allowed              type binary 32 unsigned.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsosscp-ctrs.
 02 PXS-ending-bytes               type binary 64.
 02 PXS-failures                   type binary 64.
 02 FS-direct-reads                type binary 64.
 02 FS-direct-read-bytes           type binary 64.
 02 FS-direct-writes               type binary 64.
 02 FS-direct-write-bytes          type binary 64.
 02 FS-cache-reads                 type binary 64.
 02 FS-cache-read-bytes            type binary 64.
 02 FS-cache-writes                type binary 64.
 02 FS-cache-write-bytes           type binary 64.
 02 FS-cache-valid-qtime           type binary 64.
 02 FS-cache-active-qtime          type binary 64.
 02 FS-cache-dirty-qtime           type binary 64.
 02 FS-cache-RD-read-reqs          type binary 64.
 02 FS-cache-WT-read-reqs          type binary 64.
 02 FS-cache-read-blocks           type binary 64.
 02 FS-cache-write-reqs            type binary 64.
```

```
02 FS-cache-write-blocks          type binary 64.
02 FS-prefetch-blocks             type binary 64.
02 FS-prefetch-used               type binary 64.
02 FS-reread-stolen-blks          type binary 64.
02 FS-map-failures                type binary 64.
02 FM-cache-infos                 type binary 64.
02 FM-callbacks                   type binary 64.
02 FM-callback-WT-reqs            type binary 64.
02 FM-callback-WT-blocks          type binary 64.
02 FM-stolen-blk-WT-reqs          type binary 64.
02 PP-block-inuse-qtime           type binary 64.
02 PP-alloc-failures              type binary 64.
02 PS-gettime-reqs                type binary 64.
02 PS-settime-reqs                type binary 64.

02 OSS-opens                      type binary 64.
02 OSS-opens-limit                type binary 64.
02 PXS-buf-bytes                  type binary 64.
02 PXS-buf-bytes-limit            type binary 64.
02 PXS64-buf-bytes                type binary 64.
02 PXS64-buf-bytes-limit          type binary 64.
02 disk-opens                     type binary 64.
02 disk-opens-limit               type binary 64.
02 disk-cache-buf-bytes           type binary 64.
02 disk-cache-buf-bytes-limit     type binary 64.
02 AF-INET-socket-opens           type binary 64.
02 AF-INET-socket-opens-limit     type binary 64.
02 AF-INET-socket-buf-bytes       type binary 64.
02 AF-INET-socket-buf-bytes-limit type binary 64.
02 AF-UNIX-socket-opens           type binary 64.
02 AF-UNIX-socket-opens-limit     type binary 64.
02 AF-UNIX-socket-buf-bytes       type binary 64.
02 AF-UNIX-socket-buf-bytes-limit type binary 64.
02 pipe-FIFO-opens                type binary 64.
02 pipe-FIFO-opens-limit          type binary 64.
02 pipe-FIFO-buf-bytes            type binary 64.
02 pipe-FIFO-buf-bytes-limit      type binary 64.
02 TTY-opens                      type binary 64.
02 TTY-opens-limit                type binary 64.
02 NFS-opens                      type binary 64.
02 NFS-opens-limit                type binary 64.
02 dir-opens                      type binary 64.
02 dir-opens-limit                type binary 64.
02 dir-streams                    type binary 64.
02 dir-streams-limit              type binary 64.

02 LCL                            occurs 16 times.
   03 PS-proxy-reads              type binary 64.
   03 PS-proxy-writes             type binary 64.
   03 PS-proxy-read-bytes         type binary 64.
   03 PS-proxy-write-bytes        type binary 64.
   03 PS-select-rcvd              type binary 64.
   03 PS-select-sent              type binary 64.
   03 PS-select-ready             type binary 64.
   03 LS-sends                    type binary 64.
   03 LS-recvs                    type binary 64.
   03 LS-queues                   type binary 64.
   03 LS-send-bytes               type binary 64.
```

```
    03 LS-recv-bytes                type binary 64.
    03 LS-queue-bytes               type binary 64.
    03 LS-awakes                    type binary 64.
    03 LS-selects                   type binary 64.
 02 REM                             occurs 1 times.
    03 PS-proxy-reads               type binary 64.
    03 PS-proxy-writes              type binary 64.
    03 PS-proxy-read-bytes          type binary 64.
    03 PS-proxy-write-bytes         type binary 64.
    03 PS-select-rcvd               type binary 64.
    03 PS-select-sent               type binary 64.
    03 PS-select-ready              type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsosscp. FILE is "zmsosscp" ENTRY-SEQUENCED.
 02 hdr                             type zmsheader.
 02 ctr                             type zmsosscp-ctrs.
 02 id                              type zmsosscp-id.
end
```

OSSCPU エンティティの DDL レコード（Legacy 形式）

OSSCPU エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD osscpu.  FILE is "osscpu" ENTRY-SEQUENCED.
                      .
                      .
                      .
```

error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)

```
                      .
                      .
                      .
*Entity Identification items:
 02 PXS-bytes-allowed          type binary 64.
 02 Cache-block-size           type binary 32 unsigned.
 02 Cache-blocks-allowed       type binary 32 unsigned.
 02 PP-block-size              type binary 32 unsigned.
 02 PP-blocks-allowed          type binary 32 unsigned.

* Counter value items:
 02 PXS-ending-bytes           type binary 64.
 02 PXS-failures               type binary 32 unsigned.
 02 FS-direct-reads            type binary 64.
 02 FS-direct-read-bytes       type binary 64.
 02 FS-direct-writes           type binary 64.
 02 FS-direct-write-bytes      type binary 64.
 02 FS-cache-reads             type binary 64.
 02 FS-cache-read-bytes        type binary 64.
 02 FS-cache-writes            type binary 64.
 02 FS-cache-write-bytes       type binary 64.
 02 FS-cache-valid-qtime       type binary 64.
 02 FS-cache-valid-qmax        type binary 16 unsigned.
 02 FS-cache-active-qtime      type binary 64.
```

```
 02 FS-cache-active-qmax       type binary 16 unsigned.
 02 FS-cache-dirty-qtime       type binary 64.
 02 FS-cache-dirty-qmax        type binary 16 unsigned.
 02 FS-cache-RD-read-reqs      type binary 64.
 02 FS-cache-WT-read-reqs      type binary 64.
 02 FS-cache-read-blocks       type binary 64.
 02 FS-cache-write-reqs        type binary 64.
 02 FS-cache-write-blocks      type binary 64.
 02 FS-prefetch-blocks         type binary 64.
 02 FS-prefetch-used           type binary 64.
 02 FS-reread-stolen-blks      type binary 64.
 02 FS-map-failures            type binary 32 unsigned.
 02 FM-cache-infos             type binary 32 unsigned.
 02 FM-callbacks               type binary 32 unsigned.
 02 FM-callback-WT-reqs        type binary 32 unsigned.
 02 FM-callback-WT-blocks      type binary 64.
 02 FM-stolen-blk-WT-reqs      type binary 64.
 02 PP-block-inuse-qtime       type binary 64.
 02 PP-block-inuse-qmax        type binary 16 unsigned.
 02 PP-alloc-failures          type binary 32 unsigned.
 02 PS-gettime-reqs            type binary 32 unsigned.
 02 PS-settime-reqs            type binary 32 unsigned.
 02 LCL                        occurs 16 times.
    03 PS-proxy-reads          type binary 32 unsigned.
    03 PS-proxy-writes         type binary 32 unsigned.
    03 PS-proxy-read-bytes     type binary 64.
    03 PS-proxy-write-bytes    type binary 64.
    03 PS-select-rcvd          type binary 32 unsigned.
    03 PS-select-sent          type binary 32 unsigned.
    03 PS-select-ready         type binary 32 unsigned.
 02 REM                        occurs 1 times.
    03 PS-proxy-reads          type binary 32 unsigned.
    03 PS-proxy-writes         type binary 32 unsigned.
    03 PS-proxy-read-bytes     type binary 64.
    03 PS-proxy-write-bytes    type binary 64.
    03 PS-select-rcvd          type binary 32 unsigned.
    03 PS-select-sent          type binary 32 unsigned.
    03 PS-select-ready         type binary 32 unsigned.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

OSSCPU エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧しています。

CACHE-BLOCK-SIZE

OSS ファイルシステムのキャッシュのブロックのサイズ（バイト単位）。

CACHE-BLOCKS-ALLOWED

OSS ファイルシステムのキャッシュ用に構成されたブロックの最大数。

PP-BLOCK-SIZE

OSS パイププールブロックのサイズ（バイト単位）。

PP-BLOCKS-ALLOWED

OSS パイププールで許可されるブロックの最大数。

PXS-BYTES-ALLOWED

POSIX 拡張セグメント（PXS）で許可される最大サイズ（バイト単位）。

OSSCPU エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧にしています。サブフィールドは、フィールドに指定する順に一覧にしています。

AF-INET-SOCKET-OPENS

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の現在の AF_INET ソケットのオープンの数。

カウンタータイプ：スナップショット。

AF-INET-SOCKET-OPENS-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の AF_INET ソケットのオープンの制限。

カウンタータイプ：スナップショット。

AF-INET-SOCKET-BUF-BYTES

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の AF_INET ソケットデータバッファーに割り当てられた現在のバイト数。

カウンタータイプ：スナップショット。

AF-INET-SOCKET-BUF-BYTES-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の AF_INET ソケットデータバッファーの最大サイズ（バイト単位）。

カウンタータイプ：スナップショット。

AF-UNIX-SOCKET-OPENS

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の現在の AF_UNIX ソケットのオープンの数。

カウンタータイプ：スナップショット。

AF-UNIX-SOCKET-OPENS-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の AF_UNIX ソケットのオープンの制限。

カウンタータイプ：スナップショット。

AF-UNIX-SOCKET-BUF-BYTES

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の AF_UNIX ソケットデータバッファーに割り当てられた現在のバイト数。

カウンタータイプ：スナップショット。

AF-UNIX-SOCKET-BUF-BYTES-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の AF_UNIX ソケットデータバッファーの最大サイズ（バイト単位）。

カウンタータイプ：スナップショット。

DIR-OPENS

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU で `open()` を使用したディレクトリのオープンの現在の数。

カウンタータイプ：スナップショット。

DIR-OPENS-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU で `open()` を使用したディレクトリのオープンのソフト制限。**OSSCPU エンティティの使用上の注意**を参照してください。

カウンタータイプ：スナップショット。

DIR-STREAMS

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU で `opendir()` を使用して開かれた現在のディレクトリストリームの数。

カウンタータイプ：スナップショット。

DIR-STREAMS-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU で `opendir()` を使用して開かれたディレクトリストリームのソフト制限。**OSSCPU エンティティの使用上の注意**を参照してください。

カウンタータイプ：スナップショット。

DISK-CACHE-BUF-BYTES

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の OSS の通常のファイルの OSS ファイルシステムのキャッシュバッファーに割り当てられた現在のバイト数。

カウンタータイプ：スナップショット。

DISK-CACHE-BUF-BYTES-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の通常のファイルの OSS ファイルシステムのキャッシュバッファーに割り当てられた最大サイズ（バイト単位）。

カウンタータイプ：スナップショット。

DISK-OPENS

（ZMS 形式のみ）H06.25/J06.14 以降の RVU では、CPU 上の通常のファイルの現在の OSS のオープンの数。

カウンタータイプ：スナップショット。

DISK-OPENS-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の通常のファイルの OSS のオープンのソフト制限です。**OSSCPU エンティティの使用上の注意**を参照してください。

カウンタータイプ：スナップショット。

FM-CACHE-INFOS

DP2 から受信した OSS ファイルマネージャーキャッシュ情報要求の数。OSS ファイルマネージャーは PXS とディスクキャッシュセグメントを割り振り、DP2 およびキャッシュ管理要求のための NonStop メモリマネージャーのエージェントとして機能します。

カウンタータイプ：増分。

FM-CALLBACK-WT-BLOCKS

FM-CALLBACK-WT-REQS の結果として OSS ファイルマネージャーによって書き戻されるブロックの数。

カウンタータイプ：累積。

FM-CALLBACK-WT-REQS

DP2 キャッシュコールバックのため OSS ファイルマネージャーによって発行されたキャッシュ書き込み要求の数。

カウンタータイプ：増分。

FM-CALLBACKS

OSS ファイルマネージャーによって DP2 から受信されるキャッシュコールバック要求の数。

カウンタータイプ：増分。

FM-STOLEN-BLK-WT-REQS

NonStop メモリマネージャーがダーティメモリページをスチールできるようにするために、それらのメモリページをクリーンアップするための OSS キャッシュブロック書き込みが実行された回数。

カウンタータイプ：増分。

FS-CACHE-ACTIVE-QTIME

オープンしているファイルに対して OSS ファイルシステムキャッシュブロックが存在していた時間。

カウンタータイプ：キュー。

FS-CACHE-DIRTY-QTIME

OSS ファイルシステムキャッシュブロックがダーティだった時間。

カウンタータイプ：キュー。

FS-CACHE-READ-BLOCKS

DP2 から OSS ファイルシステムキャッシュへのブロック読み取りで読み取られるブロック数。

カウンタータイプ：累積。

FS-CACHE-READ-BYTES

FS-CACHE-READS から読み取られたバイト数。

カウンタータイプ：累積。

FS-CACHE-READS

CPU 上で発生している OSS ファイルシステムキャッシュからのアプリケーション読み取りの数。

カウンタータイプ：増分。

FS-CACHE-RD-READ-REQS

アプリケーション読み取り API 呼び出しに応じるための OSS ファイルシステムキャッシュから DP2 へのブロック読み取り要求の数。ブロック読み取り論理操作は複数のブロックを返すことができます。

カウンタータイプ：増分。

FS-CACHE-VALID-QTIME

OSS ファイルシステムキャッシュブロックが有効な時間。開いているファイルの有効なキャッシュには、アクティブなキャッシュブロックと最近閉じたファイルの非アクティブな（所有されていない）ファイルキャッシュブロックが含まれています。有効でアクティブなキャッシュカウンターには、NonStop メモリマネージャーがスチールするキャッシュブロックは反映されません。**FS-REREAD-STOLEN-BLKS**(343 ページ)を参照してください。

カウンタータイプ：キュー。

FS-CACHE-WRITE-BLOCKS

OSS ファイルシステムキャッシュから FS-CACHE-WRITE-REQS により DP2 に書き込まれるブロックの数。

カウンタータイプ：累積。

FS-CACHE-WRITE-BYTES

FS-CACHE-WRITES で書き込まれたバイト数。

カウンタータイプ：累積。

FS-CACHE-WRITE-REQS

OSS ファイルシステムキャッシュから DP2 へのブロック書き込みの数。

カウンタータイプ：増分。

FS-CACHE-WRITES

OSS ファイルシステムキャッシュに書き込まれたデータを使用して処理されるアプリケーション書き込みの数。

カウンタータイプ：増分。

FS-CACHE-WT-READ-REQS

アプリケーション書き込み API 呼び出しに応じるための OSS ファイルシステムキャッシュから DP2 へのブロック読み取り要求の数。ブロック読み取り論理操作は複数のブロックを返すことができます。

カウンタータイプ：増分。

FS-DIRECT-READ-BYTES

CPU 上で FS-DIRECT-READS によって読み取られる合計バイト数。

カウンタータイプ：累積。

FS-DIRECT-READS

DP2 に対する OSS ファイルシステムのキャッシュされないディスク読み取り要求の数。

カウンタータイプ：増分。

FS-DIRECT-WRITE-BYTES

OSS ファイルシステムキャッシュを使用せずに FS-DIRECT-WRITES によって書き込まれるバイト数。

カウンタータイプ：累積。

FS-DIRECT-WRITES

DP2 に対する OSS ファイルシステムのキャッシュされない書き込み要求の数。

カウンタータイプ：増分。

FS-MAP-FAILURES

OSS ファイルシステムキャッシュが、キャッシュブロックをプリフェッチするためのマップバッファーの取得に失敗した回数。マップバッファーは DP2 との I/O 転送のために使用されるキャッシュ I/O バッファーです。マップ障害はパフォーマンスの低下の兆候である可能性があります。障害を解決するには、アプリケーションを移動します。

カウンタータイプ：増分。

FS-PREFETCH-BLOCKS

OSS ファイルシステムキャッシュによってプリフェッチされたブロックの数。

カウンタータイプ：増分。

FS-PREFETCH-USED

アプリケーションによって実際に使用されたプリフェッチブロックの数。

カウンタータイプ：増分。

FS-REREAD-STOLEN-BLKS

キャッシュデータブロックが配置されていたメモリページを NonStop メモリマネージャーが取得したため、OSS ファイルシステムキャッシュに再読み取りしなければならなかったデータブロック数。

カウンタータイプ：増分。

LCL

CPU リモートパイプサーバープロセスからの OSS パイプサーバー要求と CPU リモートローカルサーバー 2 サーバープロセスからのローカルサーバー 2 の要求の数。測定は 15 のカウンターに分割されます。

サブフィールド	説明	カウンタータイプ
PS-PROXY-READS	OSS パイプサーバープロセスによって実行されるプロキシ読み取りの数。	増分
PS-PROXY-WRITES	OSS パイプサーバープロセスによって実行されるプロキシ書き込みの数。	増分
PS-PROXY-READ-BYTES	OSS パイプサーバープロセスによって読み取られるプロキシ読み取りバイト数。	累積
PS-PROXY-WRITE-BYTES	OSS パイプサーバープロセスによって書き込まれるプロキシ書き込みバイト数。	累積

表は続く

サブフィールド	説明	カウンタータイプ
PS-SELECT-RCVD	他の CPU から OSS パイプサーバープロセスが受信した選択要求の数。	増分
PS-SELECT-SENT	他の CPU から OSS パイプサーバープロセスに送信された選択可能要求の数。	増分
PS-SELECT-READY	他の CPU から OSS パイプサーバープロセスが受信した選択可能要求の数。	増分
LS-SENDS	CPU*n* 内のサテライトソケットの代わりに LS2 プロセスによって実行された SEND メッセージの数で、*n* は LCL 配列インデックスです。Socket API、`write()`、`send()`、`sendto()`、および `sendmsg()` は、クライアントがサテライトソケットを使用しているとき（ソケットは別の CPU の管理下に置かれている）このカウンターを更新します。 **注記:** プロセスは、子孫が親のファイル記述子（開かれているファイル）を継承した他の CPU で起動できます。マスター（または元の）ソケットは、socket()、socketpair()、または accept()への呼び出しによって作成されます。ソケットを作成したプロセスが、別の CPU 内でプロセスを起動すると、サテライトソケットが作成されます。	増分
LS-SEND-BYTES	CPU*n* のサテライトソケットからの SEND 要求を受信したバイト数で、*n* は LCL 配列インデックスです。Socket API、`write()`、`send()`、`sendto()`、および `sendmsg()` は、クライアントがサテライトソケットを使用しているとき（ソケットは別の CPU の管理下に置かれている）このカウンターを更新します。（LS-SENDS カウンターの下の注記を参照してください。）	累積
LS-RECVS	CPU*n* 内のサテライトソケットの代わりに LS2 プロセスによって実行された RECEIVE 操作の数で、*n* は LCL 配列インデックスです。Socket API、`read()`、`recv()`、`recvfrom()`、および `recvmsg()` は、クライアントがサテライトソケットを使用しているとき（ソケットは別の CPU で支配される）このカウンターを更新します。（LS-SENDS カウンターの下の注記を参照してください。）	増分
LS-RECV-BYTES	CPU*n* からの RECEIVE 要求に応答して返されるバイト数で、*n* は LCL 配列インデックスです。Socket API、`read()`、`recv()`、`recvfrom()`、および `recvmsg()` は、クライアントがサテライトソケットを使用しているとき（ソケットは別の CPU で支配される）このカウンターを更新します。（LS-SENDS カウンターの下の注記を参照してください。）	累積

表は続く

サブフィールド	説明	カウンタータイプ
LS-QUEUES	LS2 プロセスによって実行された QUEUE 操作の数。QUEUE 要求はソケットクライアントから受け取るか、CPU*n* の LS2 プロセスから受け取る可能性があり、*n* は LCL 配列インデックスです。Socket API、`write()`、`send()`、`sendto()`、および `sendmsg()` は、ソケットの送信および受信が異なる CPU の管理下に置かれているときに、このカウンターを更新します。（LS-SENDS カウンターの下の注記を参照してください。）	増分
LS-QUEUE-BYTES	CPU*n* からの QUEUE 要求に応答して受け取るバイト数で、*n* は LCL 配列インデックスです。Socket API、`write()`、`send()`、`sendto()`、および `sendmsg()` は、ソケットの送信および受信が異なる CPU の管理下に置かれているときに、このカウンターを更新します。（LS-SENDS カウンターの下の注記を参照してください。）	累積
LS-AWAKES	CPU*n* 内のプロセスの代わりに実行される AWAKE 操作の数で、*n* は LCL 配列インデックスです。Socket API、`connect()`、`accept()`、`write()`、`send()`、`sendto()`、`sendmsg()`、`read()`、`recv()`、`recvfrom()`、`recvmsg()`、`shutdown()`、および `close()` は、このカウンターを更新します。	増分
LS-SELECTS	CPU*n* に対して LS2 プロセスによって実行された SELECT 操作の数で、*n* は LCL 配列インデックスです。ソケット API の `select()` は、このカウンターを更新します。さらに、`FILE_COMPLETE_()` および `FILE_COMPLETE_SET_()` プロシージャーはこのカウンターを更新します。	増分

これらのカウンターは、システム内の各 CPU につき 1 回ずつ、16 回実行されます。

NFS-OPENS

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の現在の NFS オープン（NFS を使用してアクセスできるディスクファイル）の数。

カウンタータイプ：スナップショット。

NFS-OPENS-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の NFS オープン（NFS を使用してアクセスできるディスクファイル）のソフト制限。**OSSCPU エンティティの使用上の注意**を参照してください。

カウンタータイプ：スナップショット。

OSS-OPENS

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、現在 CPU 上にあるすべてのファイルタイプの OSS ファイルオープン（通常のファイルオープン、パイプ/FIFO オープン、ディレクトリオープン、ディレクトリストリーム、TTY オープン、NFS オープン、AF_INET ソケットオープン、および AF_UNIX ソケットオープン）の数。

カウンタータイプ：スナップショット。

OSS-OPENS-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上にあるすべてのファイルタイプの OSS ファイルオープン（通常ファイルオープン、パイプ/FIFO オープン、ディレクトリオープン、ディレクトリストリーム、TTY オープン、NFS オープン、AF_INET ソケットオープン、および AF_UNIX ソケットオープン）の限度。

カウンタータイプ：スナップショット。

注記:

通常のファイルオープン、ディレクトリオープン、ディレクトリストリーム、および NFS オープンは、OSS ファイルシステムで使用される無制限のオープンリソースです。DISK-OPENS-LIMIT 値、DIR-OPENS-LIMIT 値、DIR-STREAMS-LIMIT 値、および NFS-OPENS-LIMIT 値はソフト制限を表します。OSS-OPENS-LIMIT 値に達するまで、これらのソフト制限はすべて超過可能です （また、OSS SCF インターフェイスの OSS-OPENS-LIMIT カウンターおよび ALL_OPENS リソース制限は同じ値を表すことに注意してください）。

PIPE-FIFO-OPENS

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、現在 CPU 上にあるパイプおよび FIFO のオープンの数。

カウンタータイプ：スナップショット。

PIPE-FIFO-OPENS-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上にあるパイプおよび FIFO のオープンの制限。

カウンタータイプ：スナップショット。

PIPE-FIFO-BUF-BYTES

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上のパイプおよび FIFO データバッファーに割り当てられた現在のバイト数。

カウンタータイプ：スナップショット。

PIPE-FIFO-BUF-BYTES-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上のパイプおよび FIFO データバッファーの最大サイズ（バイト単位）。

カウンタータイプ：スナップショット。

PP-ALLOC-FAILURES

OSS パイプのサイズ変更や割り当てを行えなかった回数。

カウンタータイプ：増分。

PS-GETTIME-REQS

OSS ネームサーバーから OSS パイプサーバープロセスが受信した gettime 要求の数。

カウンタータイプ：増分。

PS-SETTIME-REQS

OSS ネームサーバーから OSS パイプサーバープロセスが受信した settime 要求の数。

カウンタータイプ：増分。

PXS-BUF-BYTES

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の 32 ビットのアドレス可能メモリから POSIX 拡張セグメント（PXS）内で現在割り当てられているバイト数。

カウンタータイプ：スナップショット。

PXS-BUF-BYTES-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の 32 ビットのアドレス可能メモリから割り当てられている POSIX 拡張セグメント（PXS）の最大サイズ（バイト単位）。

カウンタータイプ：スナップショット。

PXS-ENDING-BYTES

POSIX 拡張セグメント（PXS）内で現在割り当てられているバイト数。

カウンタータイプ：スナップショット。

PXS-FAILURES

POSIX 拡張セグメント（PXS）で発生した割り当て障害の数。割り当て障害は、空き領域の不足や、連続した空き領域の不足が原因で発生することがあります。問題が解決しない場合は、ユーザーアプリケーションを別の CPU に移す必要が生じることがあります。

カウンタータイプ：増分。

PXS64-BUF-BYTES

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の 64 ビットのアドレス可能メモリから POSIX 拡張セグメント（PXS64）内で現在割り当てられているバイト数。

カウンタータイプ：スナップショット。

PXS64-BUF-BYTES-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の 64 ビットのアドレス可能メモリから割り当てられている POSIX 拡張セグメント（PXS64）の最大サイズ（バイト単位）。

カウンタータイプ：スナップショット。

REM

システム-リモートパイプサーバープロセスからの OSS パイプサーバー要求の数。測定は以下の 7 つのカウンターに分割されます。

サブフィールド	説明	カウンタータイプ
PS-PROXY-READS	OSS パイプサーバープロセスによって実行されるプロキシ読み取りの数。	増分
PS-PROXY-WRITES	OSS パイプサーバープロセスによって実行されるプロキシ書き込みの数。	増分
PS-PROXY-READ-BYTES	OSS パイプサーバープロセスによって読み取られるプロキシ読み取りバイト数。	累積

表は続く

サブフィールド	説明	カウンタータイプ
PS-PROXY-WRITE-BYTES	OSS パイプサーバープロセスによって書き込まれるプロキシ書き込みバイト数。	累積
PS-SELECT-RCVD	他のシステムから OSS パイプサーバープロセスが受信した選択要求の数。	増分
PS-SELECT-SENT	他のシステムから OSS パイプサーバープロセスに送信された選択可能要求の数。	増分
PS-SELECT-READY	システムから OSS パイプサーバープロセスが受信した選択可能要求の数。	増分

これらのカウンターは、リモートシステムに対して一度だけ行われます。

TTY-OPENS

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の現在の TTY オープンの数（/G 名および/dev/tty 名を使用した tty デバイスのオープン）。

カウンタータイプ：スナップショット。

TTY-OPENS-LIMIT

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、CPU 上の TTY オープンの限度（/G 名および/dev/tty 名を使用した tty デバイスのオープン）。

カウンタータイプ：スナップショット。

OSSCPU エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

FS-CACHE-ACTIVE-QMAX

（Measure G10 PVU のみ）使用中のアクティブなキャッシュブロックの最大数。

カウンタータイプ：最大キュー。

FS-CACHE-DIRTY-QMAX

（Measure G10 PVU のみ）ダーティキャッシュブロックの最大値。

カウンタータイプ：最大キュー。

FS-CACHE-VALID-QMAX

（Measure G10 PVU のみ）有効なキャッシュブロックの最大値。

カウンタータイプ：最大キュー。

PP-BLOCK-INUSE-QMAX

（Measure G10 PVU のみ）OSS パイププールで使用されているブロックの最大数。

カウンタータイプ：最大キュー。

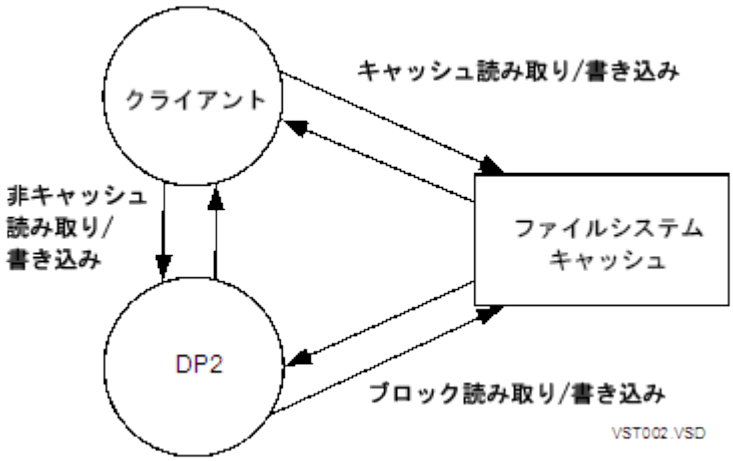
OSSCPU エンティティの使用上の注意

- 次の OSSCPU カウンター値は、OSS ファイルシステムが使用する無制限のオープンリソースのソフト制限を表します。
  - DIR-OPENS-LIMIT
  - DIR-STREAMS-LIMIT
  - DISK-OPENS-LIMIT
  - NFS-OPENS-LIMIT

  無制限のオープンリソースには、128,000 の OSS-OPENS-LIMIT に達しない限り超過できるソフト制限があります （Measure の OSS-OPENS-LIMIT カウンターおよび OSS SCF インターフェイスの ALL_OPENS リソース制限は同じ値を表すことに注意してください）。

  OSS 環境の制限について詳しくは、*Open System Services Management and Operations Guide* を参照してください。
- 通常のファイルでの読み取りまたは書き込み操作は、DP2 に直接送信することも、OSS ファイルシステムキャッシュ経由で実行することもできます。
  - キャッシュされない操作（FS-DIRECT-READS、FS-DIRECT-WRITES）は、結果として DP2 へのメッセージを生成し、クライアントが指定した量の情報しか転送しません。
  - キャッシュされた操作（FS-CACHE-READS、FS-CACHE-WRITES）は、クライアントの CPU のファイルシステムキャッシュを利用し、結果としてキャッシュと DP2 との間のブロック読み取り/書き込み（FS-CACHE-RD-READ-REQS、FS-CACHE-WT-READ-REQS、FS-CACHE-WRITE-REQS）を実行する可能性があります。

    ブロックの読み取りと書き込みは、キャッシュブロックが 4 KB であるキャッシュブロック境界（ファイルの最後のキャッシュブロックを除く）で発生します。ブロックの読み取り操作は、直近の情報の必要性を見込んで先読み（FS-PREFETCH-BLOCKS）を試みます。OSS ファイルシステムはキャッシュの使用を制御し、内部的な理由によって、特定のファイルオープンではその使用をバイパスするよう選択することもできます。



- パイプが送信側または受信側からリモートにある場合は、パイプが作成された CPU のパイプサーバーがリモートクライアント用のエージェントとして機能します。書き込み要求と読み取り要求は、結果としてパイプが作成された CPU にあるパイプサーバーへのメッセージを生成します。その後、パイプサーバーが、リモートクライアントの代わりにプロキシパイプの読み取りと書き込みをパイプバッファーに対して実行します。
- クライアントとパイプサーバー間、およびパイプサーバー間の select()メッセージは、クライアントが呼び出す select()が、パイプが作成された CPU 以外の CPU で実行されているときにのみ発生します。

  1. クライアントは、パイプを管理するパイプサーバーに選択要求を送信します。
  2. パイプサーバーは即座に応答します。

3. クライアントの読み取り、書き込み、またはその両方が正常に行われると、パイプサーバーはクライアントの CPU にあるパイプサーバーに選択可能要求を送信します。
4. そのパイプサーバーがクライアントを再びアクティブにして、select()呼び出しが完了します。

- OSS キャッシュミス率は次のように計算することができます。

```
OSS cache miss rate = (FS-Cache-RD-Read-Reqs +
                       FS-Cache-WT-Read-Reqs +
                       FS-Cache-Write-Reqs)

------------------------------------------------------------

(FS-Cache-Reads + FS-Cache-Writes)
```

OSSNS

OSSNS エンティティは、OSS ネームサーバープロセスの操作とパフォーマンスに関する情報を提供します。システム上で定義されている OSS ネームサーバーごとに、プライマリおよびバックアッププロセス用のレコードが 1 つ構成されています。OSS ファイルのパス名を含む OSS 操作はすべて、OSS ネームサーバーを使用します。

OSSNS エンティティは、Measure G10 以降の PVU を実行しているシステムでのみ使用できます。

トピック

OSSNS エンティティのエンティティ指定構文

OSSNS エンティティの DDL レコード

OSSNS エンティティの DDL フィールドの定義

OSSNS エンティティの使用上の注意

OSSNS エンティティのエンティティ指定構文

OSSNS エンティティを記述するには、次の構文を使用します。

```
OSSNS entity-spec [, entity-spec ] ...
```

entity-spec

次のように指定します。

```
{ *                                                  }
{ pid                                                }
{ $process-name [(pid)]                              }
{ [[$device.]subvolume.]filename [(pid)]             }
```

指定する値は次のとおりです。

*****

ローカルシステム上のすべての OSS ネームサーバープロセスの要素を測定します。

pid

測定対象の OSS ネームサーバープロセスのプロセス ID です。*pid* は次のいずれかのように指定します。

```
{ cpu,pin           }
{ SYSTEM-PROCESSES }
```

cpu

測定対象の OSS ネームサーバープロセスが実行されている CPU の番号です。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

pin

測定対象の OSS ネームサーバープロセスのプロセス識別番号です。すべてのプロセス識別番号を指定するには、アスタリスク（*）を使用します

SYSTEM-PROCESSES

SYSGEN プログラムを使用してインストールしたすべての OSS ネームサーバープロセスを測定します。

process-name

測定する特定の OSS ネームサーバープロセスの Guardian 名です。

$*device*

測定対象の OSS ネームサーバープロセスのオブジェクトファイルが置かれているデバイス（ボリューム）です。すべてのボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのボリュームです。

subvolume

測定対象の OSS ネームサーバープロセスのオブジェクトファイルが置かれているサブボリュームです。すべてのサブボリュームを指定するには、アスタリスク（*）を使用します。

filename

測定対象の OSS ネームサーバープロセスのオブジェクトファイルの名前です。すべてのファイル（一時ファイルを除く）を指定するには、アスタリスク（*）を使用します。

OSSNS エンティティの DDL レコード（ZMS 形式）

OSSNS エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsossns-id.
 02 pin                        type binary 16 unsigned.
 02 reserved-1                 type character 6.
 02 process-name               type character 8.
 02 program-file-name.
    03 volume                  type character 8.
    03 subvol                  type character 8.
    03 filename                type character 8.
 02 IC-entries                 type binary 32 unsigned.
 02 LC-entries                 type binary 32 unsigned.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsossns-ctrs.
 02 RR-processed                type binary 64.
 02 RR-redir-sent               type binary 64.
 02 RR-redir-processed          type binary 64.
 02 IC-lookups                  type binary 64.
 02 IC-hits                     type binary 64.
 02 IC-dirty-qtime              type binary 64.
 02 LC-lookups                  type binary 64.
 02 LC-hits                     type binary 64.
 02 LC-inuse-qtime              type binary 64.
 02 Checkpoint-reqs             type binary 64.
 02 Checkpoint-blks             type binary 64.
 02 sem-waits                   type binary 64.
 02 sem-wait-qtime              type binary 64.
 02 DP2-DD-reqs                 type binary 64.
 02 DP2-messages                type binary 64.
 02 DP2-msg-qtime               type binary 64.
 02 PS-messages                 type binary 64.
 02 PS-msg-qtime                type binary 64.
 02 LS-messages                 type binary 64.
 02 LS-msg-qtime                type binary 64.
 02 gettime-reqs                type binary 64.
 02 settime-reqs                type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsossns. FILE is "zmsossns" ENTRY-SEQUENCED.
 02 hdr                         type zmsheader.
 02 ctr                         type zmsossns-ctrs.
 02 id                          type zmsossns-id.
end
```

OSSNS エンティティの DDL レコード（Legacy 形式）

OSSNS エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD ossns.  FILE is "ossns" ENTRY-SEQUENCED.
                      .
                      .
                      .
error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)
                      .
                      .
                      .
* Entity identification items:
 02 pin                        type binary 16 unsigned.
 02 process-name               type character 8.
 02 program-file-name.
    03 volume                  type character 8.
    03 subvol                  type character 8.
    03 filename                type character 8.
 02 IC-entries                 type binary 32 unsigned.
```

```
 02 LC-entries                  type binary 32 unsigned.

*Counter value items:
 02 RR-processed                type binary 32 unsigned.
 02 RR-redir-sent               type binary 32 unsigned.
 02 RR-redir-processed          type binary 32 unsigned.
 02 IC-lookups                  type binary 32 unsigned.
 02 IC-hits                     type binary 32 unsigned.
 02 IC-dirty-qtime              type binary 64.
 02 LC-lookups                  type binary 32 unsigned.
 02 LC-hits                     type binary 32 unsigned.
 02 LC-inuse-qtime              type binary 64.
 02 Checkpoint-reqs             type binary 32 unsigned.
 02 Checkpoint-blks             type binary 32 unsigned.
 02 Sem-waits                   type binary 32 unsigned.
 02 Sem-wait-qtime              type binary 64.
 02 Sem-wait-qlen-max           type binary 16 unsigned.
 02 DP2-DD-reqs                 type binary 32 unsigned.
 02 DP2-messages                type binary 32 unsigned.
 02 DP2-msg-qtime               type binary 64.
 02 DP2-msg-qlen-max            type binary 16 unsigned.
 02 PS-messages                 type binary 32 unsigned.
 02 PS-msg-qtime                type binary 64.
 02 PS-msg-qlen-max             type binary 16 unsigned.
 02 LS-messages                 type binary 32 unsigned.
 02 LS-msg-qtime                type binary 64.
 02 LS-msg-qlen-max             type binary 16 unsigned.
 02 gettime-reqs                type binary 32 unsigned.
 02 settime-reqs                type binary 32 unsigned.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

OSSNS エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧にしています。サブフィールドは、フィールドに指定する順に一覧にしています。

IC-ENTRIES

OSS ネームサーバー上で利用可能な構成済みの i ノードキャッシュレコードの数。

LC-ENTRIES

OSS ネームサーバー上で利用可能な構成済みのリンクキャッシュレコードの数。

PIN

測定対象の OSS ネームサーバープロセスのプロセス識別番号。

PROCESS-NAME

OSS ネームサーバープロセスの名前。

PROGRAM-FILE-NAME

測定対象の OSS ネームサーバープロセスのオブジェクトのファイル。この名前は、NonStop ファイルか SMF ファイルのいずれかに適用できます。PROGRAM-FILE-NAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、プログラムファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、OSS ネームサーバープロセスを開いたときに指定された物理ファイル名を表します。 SMF ファイルでは、OSS ネームサーバープロセスを開いたときに使用された、位置に依存しない論理ファイル名を表します。

OSSNS エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧しています。

CHECKPOINT-BLKS

OSS ネームサーバーによってそのバックアップに送信されるチェックポイントブロックの数。

カウンタータイプ：増分。

CHECKPOINT-REQS

OSS ネームサーバーによって送信される論理チェックポイント要求の数。チェックポイント要求は、複数の要求が各メッセージのバックアップに送信されるように、プライマリのブロックに集められます。

カウンタータイプ：増分。

DP2-DD-REQS

OSS ネームサーバーによって DP2 に送信されるデータ定義要求の数。データ定義操作は、以下のプロシージャーによって開始されます。

creat()	link()	mkdir()	chmod()	utime()	bind()
open()	unlink()	rmdir()	chown()	rename()	

カウンタータイプ：増分。

DP2-MESSAGES

OSS ネームサーバーが DP2 に送信したメッセージの数。合計には、ファイルラベルを変更するデータ定義呼び出しと、ディスクエントリーを変更しないその他の呼び出しが含まれます。

カウンタータイプ：増分。

DP2-MSG-QTIME

DP2 メッセージが OSS ネームサーバー上で処理待ちだったときに、スレッドでの待機に費やした累積サービス時間。

カウンタータイプ：キュー。

GETTIME-REQS

OSS ネームサーバーによってパイプサーバーまたはローカルサーバーに送信される gettime 要求の数。

カウンタータイプ：増分。

IC-DIRTY-QTIME

i ノードキャッシュエントリーが OSS ネームサーバーでダーティだった時間。

カウンタータイプ：キュー。

IC-HITS

OSS ネームサーバーによって発生した i ノードキャッシュヒットの数。i ノードキャッシュミスを導出するには、IC-LOOKUPS から IC-HITS を減算します。

カウンタータイプ：増分。

IC-LOOKUPS

OSS ネームサーバーによって実行された i ノードキャッシュ検索の数。

カウンタータイプ：増分。

LC-HITS

OSS ネームサーバーによって発生したリンクキャッシュヒットの数。LC-LOOKUPS から LC-HITS を減算することで、リンクキャッシュミスを導出できます。

カウンタータイプ：増分。

LC-INUSE-QTIME

リンクキャッシュエントリーが OSS ネームサーバーで使用中だった時間。リンクキャッシュは必要に応じて、設定済みのエントリー数を超えて拡張されます。拡張された場合、OSS ネームサーバープロセスは需要が低下したとき、リンクキャッシュリソースを設定された元のサイズに戻します。

カウンタータイプ：キュー。

LC-LOOKUPS

OSS ネームサーバーによって実行されたリンクキャッシュ検索の数。

カウンタータイプ：増分。

LS-MESSAGES

OSS ネームサーバーがローカル OSS サーバープロセスに送信したメッセージの数。ローカル OSS サーバー（AF_UNIX R2 より前の場合は$ZPLS、$ZLS*nn*、*nn* は CPU 番号、または AF_UNIX R2 の場合は $ZLSNN）は、AF_UNIX ソケットのトランスポートプロバイダーです。

カウンタータイプ：増分。

LS-MSG-QTIME

OSS ネームサーバーによるローカルサーバーメッセージの完了を待機するのに費やした時間。

カウンタータイプ：キュー。

PS-MESSAGES

OSS ネームサーバーがパイプサーバープロセスに送信したメッセージの数。パイプサーバーは各 CPU で実行し（$ZPP*nn*、ここで *nn* は CPU 番号）、OSS パイプおよび FIFO を使用してアプリケーションにパイプサービスを提供します。

カウンタータイプ：増分。

PS-MSG-QTIME

OSS ネームサーバーのパイプサーバーメッセージが未処理だった時間。

カウンタータイプ：キュー。

RR-PROCESSED

測定対象の OSS ネームサーバーが処理した名前解決要求の数。

カウンタータイプ：増分。

RR-REDIR-PROCESSED

OSS ネームサーバーが処理した、リダイレクトされた名前解決要求の数。

カウンタータイプ：増分。

RR-REDIR-SENT

OSS ネームサーバーがリダイレクトした名前解決要求の数。

カウンタータイプ：増分。

SEM-WAIT-QTIME

OSS ネームサーバーの処理スレッドがセマフォ待機状態で費やした時間。OSS ネームサーバーはマルチスレッドプロセスです。セマフォ待機は、名前解決要求を処理しているスレッドに固有です。

カウンタータイプ：キュー。

SEM-WAITS

OSS ネームサーバーの処理スレッドで発生するセマフォ待機の数。

カウンタータイプ：増分。

SETTIME-REQS

OSS ネームサーバーによってパイプサーバーまたはローカルサーバーに送信される settime 要求の数。

カウンタータイプ：増分。

OSSNS エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

DP2-MSG-QLEN-MAX

（Measure G10 PVU のみ）OSS ネームサーバーが実行を開始してから未処理の DP2 メッセージの最大数。

H シリーズ以降の RVU では、このカウンターの値は 1 です。

カウンタータイプ：最大キュー。

LS-MSG-QLEN-MAX

（Measure G10 PVU のみ）OSS ネームサーバーが実行を開始してから OSS ネームサーバー上で未処理の同時ローカルサーバーメッセージの最大数。

H シリーズ以降の RVU では、このカウンターの値は 1 です。

カウンタータイプ：最大キュー。

PS-MSG-QLEN-MAX

（Measure G10 PVU のみ）OSS ネームサーバーが実行を開始してからそのサーバー上で未処理の同時パイプサーバーメッセージの最大数。

H シリーズ以降の RVU では、このカウンターの値は 1 です。

カウンタータイプ：最大キュー。

SEM-WAIT-QLEN-MAX

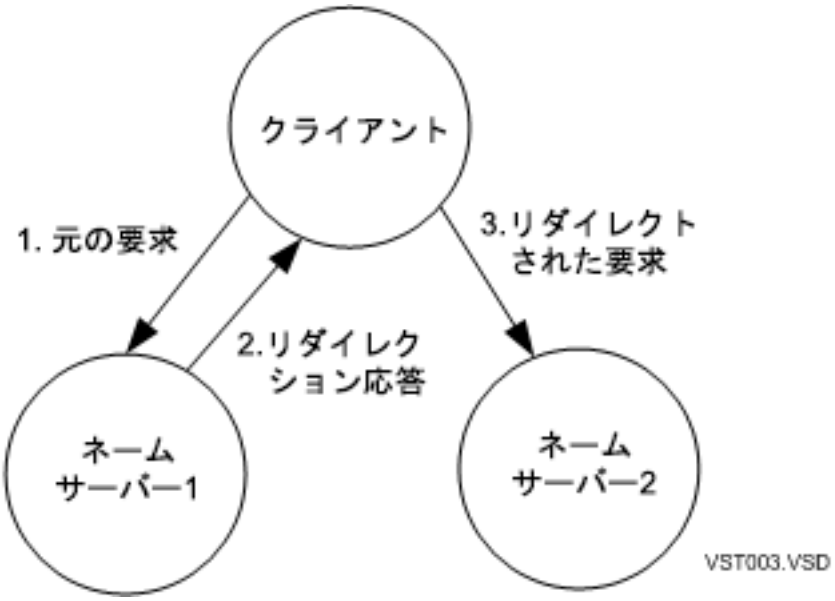
（Measure G10 PVU のみ）実行を開始してからの OSS ネームサーバーでの解決要求セマフォ待機の最大数。

H シリーズ以降の RVU では、このカウンターの値は 1 です。

カウンタータイプ：最大キュー。

OSSNS エンティティの使用上の注意

- 受信またはリダイレクトした解決要求を報告するフィールドには、適切なリソースに要求をルーティングするために、OSS ネームサーバーの容量が反映されます。要求がネームサーバーへ送信されると、ネームサーバーは完全に要求を処理できる場合も、別のネームサーバーのサービスを必要とする場合もあります。後者の場合、最初のネームサーバーは、リダイレクションの応答をクライアントに返信し、これによりクライアント内で実行しているファイルシステムライブラリが新しいリダイレクトされた要求を構成して、2 番目のネームサーバーに送信します。



- i ノードおよびリンクキャッシュカウンターは、OSS ファイルセットカタログアクセスのキャッシュパフォーマンスを報告します。OSS ファイルセットカタログは、次の 3 つの Guardian ディスクファイルから構成されます。

PXINODE	ディレクトリごとに 1 つのレコードを含むキーシーケンスファイル、通常のファイル、FIFO、および AF_UNIX ソケット。
PXLINK	ファイルセット内のファイル名ごとに 1 つのレコードを含むキーシーケンスファイル。
PXLOG	障害後にカタログの整合性を確保するために使用される非構造化ファイル。BUFFERED NONE でファイルセットが構成されている場合にのみ、PXLOG が使用されます。

PXINODE および PXLINK ファイルへのアクセスの頻度を減らすために、ネームサーバーはこれらのファイルからのレコードをメモリにキャッシュします。ファイルごとに個別のキャッシュが使用されます。キャッシュサイズは、IC-ENTRIES と LC-ENTRIES で報告されます。キャッシュのサイズは、個別に変更できます。プライマリおよびバックアップの両方のネームサーバープロセスに PXINODE キャッシュがあります。プライマリプロセスだけに PXLINK キャッシュがあります。PXLINK キャッシュは、PXLINK レコードのコピーを保持するだけでなく、失敗した名前検索も記憶します。名前が再度検索された場合、PXLINK ファイルを再度読み込んで名前が存在しないことを再び検出する必要はありません。

PROCESS

PROCESS エンティティは、ローカルシステム上の 1 つまたは複数のプロセスに関する情報を提供します。

Measure G09 以降の PVU では、PROCESS エンティティタイプは OSS ファイルパス名を扱います。

トピック

PROCESS エンティティのエンティティ指定構文

PROCESS エンティティの DDL レコード

PROCESS エンティティの DDL フィールドの定義

PROCESS エンティティの使用上の注意

PROCESS エンティティのエンティティ指定構文

PROCESS エンティティを記述するには、次の構文を使用します。

```
PROCESS entity-spec [,entity-spec] ...
```

PROCESS

ローカルシステム上の 1 つまたは複数のプロセスに関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                                                    }
{ pid                                                  }
{ $process-name [ ( pid ) ]                            }
{ [[$device.]subvolume.]filename[:crvsn] [ ( pid ) ] }
{ "pname" [ ( pid ) ]                                  }
```

指定する値は次のとおりです。

*****

ローカルシステム上のすべてのプロセスを測定します。

pid

測定対象のプロセスのプロセス ID です。pid は次のいずれかのように指定します。

```
{ cpu,pin           }
{ SYSTEM-PROCESSES }
```

cpu

測定対象のプロセスが実行されている CPU の番号です。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

pin

測定対象のプロセスのプロセス識別番号です。すべてのプロセスを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのプロセスです。

SYSTEM-PROCESSES

SYSGEN プログラムを使用してインストールしたすべてのプロセスを測定します。

$*process-name*

測定対象のプロセスの名前です。すべてのプロセスを指定するには、アスタリスク（*）を使用します。

$*device*

測定対象のプロセスのオブジェクトファイルが置かれているボリューム（デバイス）です。すべてのボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのボリュームです。

subvolume

測定対象のプロセスのオブジェクトファイルが置かれているサブボリュームです。すべてのサブボリュームを指定するには、アスタリスク（*）を使用します。

filename

測定対象の実行中のプロセスのオブジェクトファイルの名前です。すべてのファイル（一時ファイルを除く）を指定するには、アスタリスク（*）を使用します。

:*crvsn*

（Measure G09 以降の PVU）一意のファイル名を形成するために必要な、タイムスタンプ、作成バージョンのシリアル番号（CRVSN）、またはファイル名拡張子です。ファイルの CRVSN を取得するには、Measure のレポートを表示するか、LISTGNAME コマンドを使用します。

"*pname*"

（Measure G09 以降の PVU）完全修飾または部分修飾のいずれかの OSS ファイルパス名です。スラッシュ（/）で始まらない OSS ファイルパス名は、部分的なパス名とみなされ、OSSPATH の現在の設定を前に指定することで拡張されます。/G で始まる OSS ファイルパス名はサポートされていないため、代わりに Guardian ファイル名を使用してください。

注記:

OSS ファイルパス名は大文字と小文字が区別され、二重引用符（""）で囲んで指定する必要があります。有効な OSS ファイルパス名は、特定のファイル、または特定のディレクトリ内のファイルのセットを参照できます。ディレクトリが指定されている場合は、そのディレクトリ内のファイルのみが含まれます。指定したディレクトリの下位のディレクトリ内のファイルは含まれません。

PROCESS エンティティの DDL レコード（ZMS 形式）

PROCESS エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsproc-id.

 02 pin                          type binary 16 unsigned.
 02 priority                     type binary 16 unsigned.
 02 userid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 creatorid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 process-name                 type character 8.
 02 program-file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 osspid                       type binary 32 unsigned.
 02 ancestor-cpu                 type binary 16 unsigned.
 02 ancestor-pin                 type binary 16 unsigned.
 02 ancestor-sysname             type character 8.
 02 ancestor-process-name        type character 8.
 02 device-name                  type character 8.
 02 program-file-name-MID.
    03 PATHID                    type character 24.
    03 CRVSN                     type character 6.
 02 flags                        type binary 16 unsigned.
 02 GMOM.
    03 GMOM-node                 type binary 16 unsigned.
    03 GMOM-cpu                  type binary 16 unsigned.
    03 GMOM-pin                  type binary 16 unsigned.
    03 GMOM-jobid                type binary 16 unsigned.
 02 GMOM-full-id                 type binary 64
                                 redefines GMOM.
 02 GMOM-sysname                 type character 8.
 02 GMOM-process-name            type character 8.
 02 program-accelerated          type binary 16 unsigned.
 02 native-process               type binary 16 unsigned.
 02 system-process               type binary 16 unsigned.
 02 ipus                         type binary 16 unsigned.
 02 hometerm-sysname             type character 8.
 02 hometerm-name.
    03 device                    type character 8.
    03 subdevice                 type character 8.
    03 qualifier                 type character 8.

 02 serverclass-name             type character 32.
 02 reserved-1                   type character 8.

end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsproc-ctrs.
 02 cpu-busy-time              type binary 64.
 02 ready-time                 type binary 64.
 02 mem-qtime                  type binary 64.
 02 dispatches                 type binary 64.
 02 page-faults                type binary 64.
 02 pres-pages-qtime           type binary 64.
 02 ext-segs-qtime             type binary 64.
 02 vsems                      type binary 64.
 02 recv-qtime                 type binary 64.
 02 messages-sent              type binary 64.
 02 sent-bytes                 type binary 64.
 02 returned-bytes             type binary 64.
 02 messages-received          type binary 64.
 02 received-bytes             type binary 64.
 02 reply-bytes                type binary 64.
 02 mqc-allocations            type binary 64
 02 mqc-alloc-failures         type binary 64
 02 mqcs-inuse-qtime           type binary 64
 02 checkpoints                type binary 64.
 02 comp-traps                 type binary 64.
 02 native-busy-time           type binary 64.
 02 accel-busy-time            type binary 64.
 02 tns-busy-time              type binary 64.
 02 alloc-seg-calls            type binary 64.
 02 file-open-calls            type binary 64.
 02 info-calls                 type binary 64.
 02 begin-trans                type binary 64.
 02 abort-trans                type binary 64.
 02 pres-pages-start           type binary 32 unsigned.
 02 pres-pages-end             type binary 32 unsigned.
 02 Abs-segs-qtime             type binary 64.
 02 Abs-segs-start             type binary 32 unsigned.
 02 Abs-segs-end               type binary 32 unsigned.
 02 Msgs-sent-qtime            type binary 64.
 02 Sent-cbytes                type binary 64.
 02 Returned-cbytes            type binary 64.
 02 Received-cbytes            type binary 64.
 02 Reply-cbytes               type binary 64.
 02 OSS-tty-reads              type binary 64.
 02 OSS-tty-writes             type binary 64.
 02 OSS-tty-read-bytes         type binary 64.
 02 OSS-tty-write-bytes        type binary 64.
 02 OSS-tty-wait-time          type binary 64.
 02 OSS-dev-null-ops           type binary 64.
 02 OSSNS-DD-calls             type binary 64.
 02 OSSNS-requests             type binary 64.
 02 OSSNS-message-bytes        type binary 64.
 02 OSSNS-wait-time            type binary 64.
 02 OSSNS-redirects            type binary 64.
 02 launches                   type binary 64.
 02 launch-wait-time           type binary 64.
 02 open-close-wait-time       type binary 64.
 02 ipu-switches               type binary 64.
```

```
* New counters for H05:
 02 ipu-num                    type binary 32 unsigned.
 02 ipu-num-prev               type binary 32 unsigned.

* New counters for H06
 02 initial-priority-start     type binary 16.
 02 initial-priority-end       type binary 16.
 02 current-priority-start     type binary 16.
 02 current-priority-end       type binary 16.


 02 locked-pages-start         type binary 64.
 02 locked-pages-end           type binary 64.

* New counters for H07
 02 X-interrupts               type binary 64.
 02 Y-interrupts               type binary 64.
 02 Interrupts                 type binary 64.

 02 mainstack-size-start       type binary 64.
 02 mainstack-size-end         type binary 64.
 02 mainstack-limit-start      type binary 64.
 02 mainstack-limit-end        type binary 64.
 02 privstack-size-start       type binary 64.
 02 privstack-size-end         type binary 64.
 02 privstack-limit            type binary 64.
 02 heap32-size-start          type binary 64.
 02 heap32-size-end            type binary 64.
 02 heap32-limit               type binary 64.
 02 heap64-size-start          type binary 64.
 02 heap64-size-end            type binary 64.
 02 heap64-limit               type binary 64.
 02 globals-size               type binary 64.
 02 swapspace                  type binary 64.
 02 tles-start                 type binary 64.
 02 tles-end                   type binary 64.
 02 ipu-affinity-class-start   type binary 16 unsigned
 02 ipu-affinity-class-end     type binary 16 unsigned
 02 reserved-2                 type character 52.
```

DDL レコード記述フィールド

```
RECORD zmsproc. FILE is "zmsproc" ENTRY-SEQUENCED.
 02 hdr                        type zmsheader.
 02 ctr                        type zmsproc-ctrs.
 02 id                         type zmsproc-id.
end
```

PROCESS エンティティの DDL レコード（Legacy 形式）

PROCESS エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G シリーズの PROCESS エンティティの DDL レコードは、次の点を除いて D シリーズの PROCESS エンティティのレコードと同じです。

- フィールドのオーバーフローの可能性を減らすために、長い byte-count フィールドが提供されます。32 ビットの各 byte-count フィールドには、対応する 64 ビットがあります。64 ビットのフィールドの

使用については、**G シリーズ PROCESS エンティティの使用上の注意**(386 ページ)を参照してください。

- ERROR フィールドが、32 ビットの byte-count フィールドのフィールドオーバーフローを通知することがあります。
- メモリ処理でのオペレーティングシステムの変更をサポートするカウンターが追加されています。メモリ処理の変更は、既存の CPU-BUSY-TIME、READY-TIME、および PRES-PAGES-QTIME の各カウンターの解釈にも影響します。
- Measure G09 以降の PVU では、OSS ファイルパス名のサポートのためにいくつかの識別子が追加されています。

```
RECORD process. FILE is "process" ENTRY-SEQUENCED.
                        .
                        .
                        .
error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)
                        .
                        .
                        .
* entity identification items:
 02 pin                          type binary 16 unsigned.
 02 process-name                 type character 8.
 02 program-file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 priority                     type binary 16 unsigned.
* counter value items:
 02 cpu-busy-time                type binary 64.
 02 ready-time                   type binary 64.
 02 mem-qtime                    type binary 64.
 02 dispatches                   type binary 32 unsigned.
 02 page-faults                  type binary 32 unsigned.
 02 pres-pages-qtime             type binary 64.
 02 pres-pages-max               type binary 16 unsigned.
 02 ext-segs-qtime               type binary 64.
 02 ext-segs-max                 type binary 16 unsigned.
 02 vsems                        type binary 32 unsigned.
 02 recv-qtime                   type binary 64.
 02 recv-qlen-max                type binary 16 unsigned.
 02 messages-sent                type binary 32 unsigned.
 02 sent-bytes                   type binary 32 unsigned.
 02 returned-bytes               type binary 32 unsigned.
 02 messages-received            type binary 32 unsigned.
 02 received-bytes               type binary 32 unsigned.
 02 reply-bytes                  type binary 32 unsigned.
 02 lcb-allocations              type binary 32 unsigned.
 02 mqc-allocations              type binary 32 unsigned
                                   redefines lcb-allocations.
 02 lcb-alloc-failures           type binary 32 unsigned.
 02 mqc-alloc-failures           type binary 32 unsigned
                                  redefines lcb-alloc-failures.
 02 lcbs-inuse-qtime             type binary 64.
 02 mqcs-inuse-qtime             type binary 64
                                  redefines lcbs-inuse-qtime.
 02 max-lcbs-inuse               type binary 16 unsigned.
```

```
 02 max-mqcs-inuse             type binary 16 unsigned
                                redefines max-lcbs-inuse.
 02 checkpoints                type binary 32 unsigned.
* new fields for entity identification:
 02 userid.
    03 group                   type binary 8 unsigned.
    03 user                    type binary 8 unsigned.
 02 creatorid.
    03 group                   type binary 8 unsigned.
    03 user                    type binary 8 unsigned.
* fields for TNS/R specific counters:
 02 accel-busy-time            type binary 64.
 02 tns-busy-time              type binary 64.
 02 comp-traps                 type binary 32 unsigned.
 02 program-accelerated        type binary 16 unsigned.
* new entity identification items for D10
 02 ancestor-cpu               type binary 16 unsigned.
 02 ancestor-pin               type binary 16 unsigned.
 02 ancestor-sysname           type character 8.
 02 ancestor-process-name      type character 8.
* Native Mode busy time:
 02 tnsr-busy-time             type binary 64.
* Native Mode process:
 02 tnsr-process               type binary 16 unsigned.
* new entity identification items and counters for D25
 02 hometerm-sysname           type character 8.
 02 hometerm-name.
    03 device                  type character 8.
    03 subdevice               type character 8.
    03 qualifier               type character 8.

 02 page-size-bytes            type binary 16 unsigned.
 02 alloc-seg-calls            type binary 32 unsigned.

 02 UCL-qtime                  type binary 64.
 02 UCL-max                    type binary 16 unsigned.

 02 UCL-lock-qtime             type binary 64.
 02 UCL-lock-max               type binary 16 unsigned.

02 file-open-calls             type binary 32 unsigned.
02 info-calls                  type binary 32 unsigned.
* D30 changes:
 02 begin-trans                type binary 32 unsigned.
 02 abort-trans                type binary 32 unsigned.

* SMS changes:
 02 device-name                type character 8.

* F40 new counter value items:
 02 sent-bytes-f               type binary 64.
 02 returned-bytes-f           type binary 64.
 02 received-bytes-f           type binary 64.
 02 reply-bytes-f              type binary 64.
* New counters for G04
 02 pres-pages-start           type binary 32 unsigned.
 02 pres-pages-end             type binary 32 unsigned.
```

```
 02 abs-segs-qtime             type binary 64.
 02 abs-segs-qlen-max          type binary 16 unsigned.
 02 abs-segs-start             type binary 32 unsigned.
 02 abs-segs-end               type binary 32 unsigned.
* New counters for G08:
 02 msgs-sent-qtime            type binary 64.
 02 msgs-sent-qlen-max         type binary 16 unsigned.
 02 sent-cbytes                type binary 64.
 02 returned-cbytes            type binary 64.
 02 received-cbytes            type binary 64.
 02 reply-cbytes               type binary 64.
* New identifiers for OSS file pathname support:
 02 osspid                     type binary 32 unsigned.
 02 program-file-name-MID.
    03 PATHID                  type character 24.
    03 CRVSN                   type character 6.
* New identifiers for NetBatch Job Control:
 02 GMOM.
    03 GMOM-node               type binary 16 unsigned.
    03 GMOM-cpu                type binary 16 unsigned.
    03 GMOM-pin                type binary 16 unsigned.
    03 GMOM-jobid              type binary 16 unsigned.
 02 GMOM-full-id               type binary 64
                               redefines GMOM.
 02 GMOM-sysname               type character 8.
 02 GMOM-process-name          type character 8.
* New counters for G10:

 02 OSS-tty-reads              type binary 32 unsigned.
 02 OSS-tty-writes             type binary 32 unsigned.
 02 OSS-tty-read-bytes         type binary 64.
 02 OSS-tty-write-bytes        type binary 64.
 02 OSS-tty-wait-time          type binary 64.
 02 OSS-dev-null-ops           type binary 64.
 02 OSSNS-DD-calls             type binary 32 unsigned.
 02 OSSNS-requests             type binary 32 unsigned.
 02 OSSNS-message-bytes        type binary 64.
 02 OSSNS-wait-time            type binary 64.
 02 OSSNS-redirects            type binary 32 unsigned.
 02 launches                   type binary 32 unsigned.
 02 launch-wait-time           type binary 64.
 02 open-close-wait-time       type binary 64.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

PROCESS エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧にしています。サブフィールドは、フィールドに指定する順に一覧にしています。

ANCESTOR-CPU

親プロセスが存在する CPU の番号。

ANCESTOR-PIN

親プロセスのプロセス識別番号。

ANCESTOR-PROCESS-NAME

親プロセスの名前。レポートヘッダーには、親の情報が表示されます。この情報を取得できない場合は、「unknown」という文字列が表示されます。

ANCESTOR-SYSNAME

親プロセスが存在するシステムの名前。

CREATORID

作成者アクセス ID（CAID）。このフィールドは、プロセスの作成者を識別します。これは、GROUP と USER の 2 つのサブフィールドに分かれています。

DEVICE-NAME

プログラムファイルが配置されているディスクデバイス。SMF ファイルの場合、このフィールドには、論理プログラムファイル名に対応する物理的な場所が表示されます。NonStop ファイルの場合、このフィールドは、PROGRAM-FILE-NAME VOLUME サブフィールドと同じです。

FLAGS

（ZMS 形式のみ）測定が NSMA システムまたは非 NSMA システムで行われたかどうかを示すために使用される 16 ビットフィールド。測定が割り込みプロセスまたは補助プロセスのいずれであるかを示すために使用されます。

flags.0:10	現在は使用されていません。
flags.11	1 = プロセスは補助プロセスです（AP）**
flags.12	1 = プロセスは X および Y 割り込みを処理します**
flags.13	1 = プロセスは中断プロセスです（IP）**
flags.14	現在は使用されていません。
flags.15	1 = 測定は NSMA システム上で行われました*

* H06.24/J06.13 およびこれ以降の RVU

** L01、H07/J05 およびこれ以降の PVU

GMOM

GMOM 構造は以下の 4 つのサブフィールドで構成されます。

GMOM-NODE	GMOM プロセスが NetBatch ジョブの一部であり、GMOM プロセスがリモートである場合、GMOM プロセスの Expand ノード番号。そうでない場合はゼロ。
GMOM-CPU	GMOM プロセスが NetBatch ジョブの一部である場合、GMOM プロセスのプロセッサー番号。そうでない場合はゼロ。
GMOM-PIN	GMOM プロセスが NetBatch ジョブの一部である場合、GMOM プロセスの PIN。そうでない場合はゼロ。
GMOM-JOBID	指定された GMOM によって割り当てられる、このプロセスを起動した NetBatch ジョブのジョブ ID。そうでない場合はゼロ。

GMOM-FULL-ID

個々の GMOM フィールドの単一値としての 64 ビットの再定義。ゼロの値は、プロセスが NetBatch ジョブストリームの一部でないことを示します。

GMOM-PROCESS-NAME

このプロセスを起動した NetBatch ジョブを開始した GMOM プロセスの名前。そうでない場合は空白。

GMOM-SYSNAME

GMOM プロセスがリモートの場合、GMOM ノードの Expand システム名。そうでない場合は空白。

HOMETERM-NAME

サブコンポーネントで識別される、プロセスに関連付けられるホーム端末（DEVICE 名、SUBDEVICE 名、および QUALIFIER 名）。この識別子は PROCESS レポートではなく、データ レコードにのみが表示されます。

HOMETERM-SYSNAME

プロセスに関連付けられるホーム端末システム名。この識別子は PROCESS レポートではなく、データ レコードにのみが表示されます。

IPUS

（ZMS 形式のみ）Measure J01 以降の PVU で、CPU 内の IPU の数を示します。このフィールドは MEASCOM によって表示されませんが、IPU-SWITCHES フィールドを表示するかどうかを決定します。

Measure H03 以降の H シリーズ PVU では、このフィールドは 1 で、MEASCOM によって表示されません。

NATIVE-PROCESS

（ZMS 形式のみ）ネイティブモードのみで実行するプロセスに設定されているフラグ。

OSSPID

プロセスが OSS 環境で実行される場合、このフィールドには OSS プロセス ID が含まれます。非 OSS プロセスの場合、OSSPID にはゼロが含まれます。

PIN

測定対象のプロセスのプロセス識別番号。

PRIORITY

測定対象のプロセスの作成優先順位。ALTPRI TACL コマンドを使用してプロセスの優先順位を変更しても、この値には影響しません。

PROCESS-NAME

プロセスの名前。

PROGRAM-ACCELERATED

プログラムのユーザーコード（UC）またはユーザーライブラリ（UL）部分が高速化されている場合にゼロ以外に設定されているフラグ。

PROGRAM-FILE-NAME

測定対象のオブジェクトファイル。この名前は、NonStop ファイルか SMF ファイルのいずれかに適用できます。PROGRAM-FILE-NAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、プログラムファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、プロセスを開いたときに指定された物理ファイル名を表します。 SMF ファイルでは、プロセスを開いたときに使用された、位置に依存しない論理ファイル名を表します。

PROGRAM-FILE-NAME-MID

指定した PROGRAM-FILE-NAME の OSS ファイル名の内部表現。PROGRAM-FILE-NAME-MID には次の 2 つのサブフィールドがあります。

PATHID	指定した PROGRAM-FILE-NAME の OSS ファイル名の内部表現です。非 OSS ファイルの場合、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。その他のファイルでは、このフィールドにはゼロが含まれます。

SERVERCLASS-NAME

（ZMS 形式のみ、L02、J06 以降）プロセスのサーバークラス名。このフィールドは、NonStop TS/MP 経路により入力されている場合に表示されます。

SYSTEM-PROCESS

（ZMS 形式のみ、L01、H07/J05 以降）このフィールドは、プロセスがシステムプロセスかどうかを識別し、その場合はシステムプロセスタイプを特定します。

SYSPROC_TYPE_REGULAR	0	システムプロセスではありません
SYSPROC_TYPE_MONITOR	1	モニター

表は続く

SYSPROC_TYPE_MEMMGR	2	メモリマネージャー
SYSPROC_TYPE_MESSENGER	3	メッセンジャー
SYSPROC_TYPE_MEASURE_IP	4	Measure IP
SYSPROC_TYPE_IBRKPT_IP	5	命令ブレークポイント割り込みプロセス
SYSPROC_TYPE_TIMER_IP	6	タイマー IP
SYSPROC_TYPE_MAB_IP	7	メモリアクセスブレークポイント割り込みプロセス
SYSPROC_TYPE_GFH	8	一般障害ハンドラー
SYSPROC_TYPE_SIGTRAP_MONITOR	9	信号トラップモニター
SYSPROC_TYPE_IDLE_AP	10	アイドル状態の補助プロセス
SYSPROC_TYPE_ALIVE_AP	11	アライブ状態の補助プロセス
SYSPROC_TYPE_WATCHDOG_AP	12	ウォッチドッグタイマー補助プロセス
SYSPROC_TYPE_SUSP_ACT_MON_AP	13	モニター補助プロセスを一時停止/アクティブ化します
SYSPROC_TYPE_IB_MISC_IP	14	L シリーズのその他の割り込みプロセス
SYSPROC_TYPE_IB_DRIVER_IP	15	L シリーズのドライバー割り込みプロセス
SYSPROC_TYPE_IB_COMM_IP	16	L シリーズの通信割り込みプロセス
SYSPROC_TYPE_IB_STORAGE_IP	17	L シリーズのストレージ割り込みプロセス
SYSPROC_TYPE_IB_IPC_IP	18	L シリーズのメッセージシステム割り込みプロセス
SYSPROC_TYPE_IB_ST_WQ_AP	19	L シリーズのシングルスレッド作業キュー補助プロセス
SYSPROC_TYPE_IB_MT_WQ_AP	20	L シリーズのマルチスレッド作業キュー補助プロセス
SYSPROC_TYPE_IB_HELPER_AP	21	L シリーズのヘルパー補助プロセス
SYSPROC_TYPE_IB_LNX_HELPER_AP	22	L シリーズの Linux タイマーヘルパー補助プロセス
SYSPROC_TYPE_IMIP_AP	23	L シリーズのメンテナンスインターフェイスプロトコル補助プロセス
SYSPROC_TYPE_TNET_COMM_IP	24	ServerNet 通信割り込みプロセス
SYSPROC_TYPE_TNET_STORAGE_IP	25	ServerNet ストレージ割り込みプロセス
SYSPROC_TYPE_TNET_IPC_IP	26	ServerNet メッセージシステム割り込みプロセス
SYSPROC_TYPE_TNET_ERROR_IP	27	ServerNet エラー割り込みプロセス

表は続く

SYSPROC_TYPE_TNET_TIMER_AP	28	Servernet タイマー補助プロセス
SYSPROC_TYPE_SMIP_IP	29	ServerNet メンテナンスインターフェイスプロトコル割り込みプロセス
SYSPROC_TYPE_SMIP_AP	30	ServerNet メンテナンスインターフェイスプロトコル補助プロセス
SYSPROC_TYPE_IB_NSADI_IP	31	L シリーズのアプリケーション直接インターフェイス割り込みプロセス

USERID

プロセスのプロセスアクセサー ID（PAID）。

PROCESS エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧しています。

ABORT-TRANS

プロセスが TMF ABORTTRANSACTION プロシージャーを呼び出す回数で、これによって TMF トランザクションがロールバックします。

カウンタータイプ：増分。

ABS-SEGS-END

測定間隔の終了時に ABS-SEGS-QTIME で記述されるキューの項目の数。

カウンタータイプ：スナップショット。

ABS-SEGS-QTIME

測定対象のプロセスの代わりにセグメントテーブルエントリーが絶対アドレス指定可能セグメントで使用中であった時間。

カウンタータイプ：キュー。

ABS-SEGS-START

測定間隔の開始時に ABS-SEGS-QTIME で記述されるキューの項目の数。

カウンタータイプ：スナップショット。

ACCEL-BUSY-TIME

CPU がこのプロセス内の高速化されたコードの実行でビジー状態だった時間。このカウンターは、TNS/R または TNS/E システムで行われる測定にのみ適用されます。

H シリーズ以降の PROCESS エンティティの使用上の注意(384 ページ)を参照してください。

カウンタータイプ：ビジー。

ALLOC-SEG-CALLS

SEGMENT_ALLOCATE_または ALLOCATESEGMENT プロシージャーに対する成功した呼び出しの数。

カウンタータイプ：増分。

BEGIN-TRANS

プロセスが TMF トランザクションを開始した回数。

カウンタータイプ：増分。

CHECKPOINTS

CHECKPOINT プロシージャーへの呼び出しの数。このカウンターには、このプロシージャーへのユーザー呼び出しとシステム呼び出しの両方が含まれます。

多くのシステムプロセスは、CHECKPOINT プロシージャーを呼び出すのではなく、独自のチェックを実行します。これらのプロセスは、このカウンターを増分しません。

カウンタータイプ：増分。

COMP-TRAPS

このプロセスの実行中に互換性トラップが発生した回数。互換性トラップは、データのミスアライメントイベント、高速化または非高速化されたコードとの間での予期しない移行、または相対セグメント 2 または 3 の問題です。詳しくは、*EPTRACE Manual* を参照してください。

カウンタータイプ：増分。

CPU-BUSY-TIME

CPU が測定対象のプロセスの実行に費やした時間。

- D シリーズの RVU では、LDONE キューイングが使用中の場合、送信中のプロセスに対してはこのカウンターは加算されません。
- G05 以降の RVU を実行しているシステムでの測定の場合、このカウンターには、このプロセスの代わりに実行されるページデフォルトアクティビティが含まれます。
- G05.00 より前の G シリーズの RVU の場合、すべてのページフォールトはメモリマネージャープロセス$VIRTUAL によって行われます。

TNS/R システムでは、CPU-BUSY-TIME は ACCEL-BUSY-TIME、TNS-BUSY-TIME、および TNSR-BUSY-TIME の合計です。

TNS/E システムでは、CPU-BUSY-TIME は ACCEL-BUSY-TIME、TNS-BUSY-TIME、および NATIVE-BUSY-TIME の合計です。CPU-BUSY-TIME の値は常に表示されます。ただし、ACCEL-BUSY-TIME、TNS-BUSY-TIME、および NATIVE-BUSY-TIME の各カウンター値は、測定中に PROCESSH 測定がアクティブだった場合にのみ計算され表示されます。ACCEL-BUSY-TIME、TNS-BUSY-TIME、および NATIVE-BUSY-TIME の各カウンターの動作について詳しくは、これらの各カウンターの説明を参照してください。

注記:

CPU-BUSY-TIME カウンターは、プロセッサーの使用率をより正確に測定するように拡張されています。D シリーズおよび G シリーズの RVU を実行しているシステムでの測定は、C シリーズの RVU を実行しているシステムでの同等の測定よりもわずかに高い使用率を示すことがあります。違いは主に、低い使用率レベルで実行されているシステムで現れます。

関連するカウンター	説明
CPU CPU-BUSY-TIME	類似していますが、プロセッサー全体のビジー時間です。
PROCESS CPU-NUM	プロセスを実行中の CPU を識別します

カウンタータイプ：ビジー。

CURRENT-PRIORITY-END

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、測定間隔の終了時の現在の実行プロセスの優先順位。 **H シリーズ以降の PROCESS エンティティの使用上の注意**(384 ページ)も参照してください。

カウンタータイプ：スナップショット。

CURRENT-PRIORITY-START

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、測定間隔の開始時の現在の実行優先順位。 **H シリーズ以降の PROCESS エンティティの使用上の注意**(384 ページ)も参照してください。

カウンタータイプ：スナップショット。

DISPATCHES

プロセスがレディーリストから選択され、CPU-NUM フィールドによって識別された CPU で実行された回数。I/O 操作ごとに 1 つのディスパッチと、優先順位の高いプロセスに取って代わられるプロセスによって生じる追加のディスパッチを想定できます。

カウンタータイプ：増分。

EXT-SEGS-QTIME

拡張セグメントが SEGMENT_ALLOCATE_プロシージャーによってプロセスに割り当てられた時間。このカウンターには、このプロシージャーへのユーザー呼び出しとシステム呼び出しの両方が含まれます。

システムでは、プロセスコードセグメントとデータスタックセグメントを割り当てる際に、SEGMENT_ALLOCATE_を使用しません。

カウンタータイプ：キュー。

FILE-OPEN-CALLS

FILE_OPEN_または OPEN プロシージャーの呼び出し回数。通常のファイル、パイプ、FIFO、AF_INET ソケット、AF_UNIX ソケット、ディレクトリ、/dev/null、TTY のオープンを含むすべてのオープンをカウントします。プロセス作成 API の fork(),exec*()ファミリによる暗黙的なオープンを含みます。

カウンタータイプ：増分。

GLOBALS-SIZE

（ZMS 形式のみ、L02、J06 およびそれ以降）グローバルのサイズ。

カウンタータイプ：スナップショット。

HEAP32-LIMIT

（ZMS 形式のみ、L02、J06 以降）ヒープで許容される最大サイズ。

カウンタータイプ：スナップショット。

HEAP32-SIZE-END

（ZMS 形式のみ、L02、J06 以降）測定間隔の終了時のヒープのサイズ。

カウンタータイプ：スナップショット。

HEAP32-SIZE-START

（ZMS 形式のみ、L02、J06 以降）測定間隔の開始時のヒープのサイズ。このカウンターは、MEASCOM では表示されませんが、IF/BY 句とプロットに使用できます。

カウンタータイプ：スナップショット。

HEAP64-LIMIT

（ZMS 形式のみ、L02、J06 以降）64 ビットプロセスのヒープで許容される最大サイズ。

カウンタータイプ：スナップショット。

HEAP64-SIZE-END

（ZMS 形式のみ、L02、J06 以降）測定間隔の終了時の 64 ビットプロセスのヒープのサイズ。

カウンタータイプ：スナップショット。

HEAP64-SIZE-START

（ZMS 形式のみ、L02、J06 以降）測定間隔の開始時の 64 ビットプロセスのヒープのサイズ。このカウンターは、MEASCOM では表示されませんが、IF/BY 句とプロットに使用できます。

カウンタータイプ：スナップショット。

INFO-CALLS

要求された情報を取得するために、メッセージをプロセス（IOP など）に送信する、これらのプロシージャーに対する呼び出しの数。

FILEINFO	DEVICEINFO2	getsockopt()
FILERECINFO	access()	pathconf()
FILE_GETINFO_	fpathconf()	select()
FILE_GETINFOBYNAME_	fstat()	sockatmark()
FILE_GETINFOLISTBYNAME_	fstatvfs()	stat()
FILEGETINFO_LIST	getpeername()	statvfs()
DEVICEINFO	getsockname()	

カウンタータイプ：増分。

INITIAL-PRIORITY-END

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、測定間隔の終了時に有効な、ターゲットとなる（要求された）プロセスの優先順位。測定の開始時にプロセスについて指定された最初の優先順位は、ALTPRI TACL コマンドを使用するか、ALTERPRIORITY、PRIORITY、または PROCESS_SETINFO_ Guardian プロシージャーを使用して実行中に変更できます。 **H シリーズ以降の PROCESS エンティティの使用上の注意**(384 ページ)も参照してください。

カウンタータイプ：スナップショット。

INITIAL-PRIORITY-START

（ZMS 形式のみ）H06.25/J06.14 以降の RVU で、測定間隔の開始時に有効な、ターゲットとなる（要求された）プロセスの優先順位。測定の開始時にプロセスについて指定された最初の優先順位は、ALTPRI TACL コマンドを使用するか、ALTERPRIORITY、PRIORITY、または PROCESS_SETINFO_ Guardian プロシージャーを使用して実行中に変更できます。 **H シリーズ以降の PROCESS エンティティの使用上の注意**(384 ページ)も参照してください。

カウンタータイプ：スナップショット。

INTERRUPTS

（ZMS 形式のみ、L01、H07/J05 以降）このプロセスが処理する割り込みの総数。これは派生カウンターで、X-INTERRUPTS カウンターと Y-INTERRUPTS カウンターの合計です。

カウンタータイプ：増分。

IPU-AFFINITY-CLASS-END

（ZMS 形式のみ、L02、J06 以降）測定間隔の終了時のプロセスの IPU アフィニティクラス。

カウンタータイプ：スナップショット。

IPU-AFFINITY-CLASS-START

（ZMS 形式のみ、L02、J06 以降）測定間隔の開始時のプロセスの IPU アフィニティクラス。

カウンタータイプ：スナップショット。

IPU-NUM

（ZMS 形式のみ）Measure H05/J03 以降の PVU で、サンプルが作成されたときにプロセスが実行された IPU の番号。

非 NSMA システムでは、このフィールドは常に 0（ゼロ）です。

カウンタータイプ：スナップショット。

IPU-NUM-PREV

（ZMS 形式のみ）Measure H05/J03 以降の PVU で、IPU 切り替えが測定間隔中に行われた場合、IPU-NUM-PREV には最近の切り替えの前にプロセスが実行された IPU の番号が表示されます。測定間隔中に IPU 切り替えが発生しなかった場合、このフィールドは表示されません。

非 NSMA システムでは、このフィールドは常に 0（ゼロ）です。

カウンタータイプ：スナップショット。

IPU-SWITCHES

（ZMS 形式のみ）プロセスが実行中に IPU を切り替えた合計回数を示す累積カウンター。非 NSMA システムでは、このフィールドは常に 0（ゼロ）です。

J06.07 以前の RVU では、DP2 プロセスのみが有効な値を示し、他のすべてのプロセスでは、フィールドは 0（ゼロ）として報告されていました。J06.08 以降の RVU では、フィールドにはすべてのプロセスに対して有効な値が含まれています。

カウンタータイプ：累積。

LAUNCH-WAIT-TIME

PROCESS_CREATE_またはプロセス作成 API の fork()/exec*()ファミリへの要求に対してプロセスが待機する時間。

カウンタータイプ：ビジー。

LAUNCHES

プロセス作成プロシージャーに対してプロセスによって実行された呼び出しの数。Guardian プロセス作成の場合、カウンターには作成に失敗した試行が含まれます。

カウンタータイプ：増分。

LOCKED-PAGES-END

（ZMS 形式のみ）Measure H06/J04 以降の PVU で、測定間隔の最後にメインメモリ内にロックされたプロセスに関連付けられたページフレームの数。

このカウンターは、NonStop OS が提供する PROCESS サブシステムバージョン（ZMSPROC.HDR.SUBSYSTEM-VERSION）が 4 以上の場合に有効で、MEASCOM によって ZMS 形式で表示されます。そうでない場合、カウンターにはゼロ値が格納され、MEASCOM によって表示されません。

カウンタータイプ：スナップショット。

LOCKED-PAGES-START

（ZMS 形式のみ）Measure H06/J04 以降の PVU で、測定間隔の最初にメインメモリ内にロックされたプロセスに関連付けられたページフレームの数。

このカウンターは、NonStop OS が提供する PROCESS サブシステムバージョン（ZMSPROC.HDR.SUBSYSTEM-VERSION）が 4 以上の場合に有効で、MEASCOM によって ZMS 形式で表示されます。そうでない場合、カウンターにはゼロ値が格納され、MEASCOM によって表示されません。

カウンタータイプ：スナップショット。

MAINSTACK-LIMIT-END

（ZMS 形式のみ、L02、J06 以降）測定間隔の終了時のメインスタックの許可された最大サイズ。

カウンタータイプ：スナップショット。

MAINSTACK-LIMIT-START

（ZMS 形式のみ、L02、J06 以降）測定間隔の開始時のメインスタックの許可された最大サイズ。

カウンタータイプ：スナップショット。

MAINSTACK-SIZE-END

（ZMS 形式のみ、L02、J06 以降）測定間隔の終了時のメインスタックのサイズ。

カウンタータイプ：スナップショット。

MAINSTACK-SIZE-START

（ZMS 形式のみ、L02、J06 以降）測定間隔の開始時のメインスタックのサイズ。

カウンタータイプ：スナップショット。

MEM QTIME

測定対象のプロセスがページフォールトの待機に費やした時間。ページフォールトが発生しても、システムはレディーリストからプロセスを削除しません。

関連するカウンター	説明
PROCESS PAGE-FAULTS	測定対象のプロセスによって生成されたページフォールトの数をカウントします。
CPU MEM-QTIME	特定のプロセッサーのすべてのプロセスがページフォールトの待機に費やした時間を測定します。
CPU CPU-QTIME	ページフォールトの待機に費やした時間を含め、すべてのプロセスがレディーリストで費やした時間を測定します。

カウンタータイプ：キュー。

MESSAGES-RECEIVED

プロセスがそのメッセージ入力キューから読み取ったメッセージの数。

ユーザープロセスの場合、メッセージ入力キューは$RECEIVE です。

次の操作では、このカウンターは加算されません。

- SIGNALTIMEOUT と SIGNALPROCESSTIMEOUT によりプロセスが受信したメッセージ
- LDONE キューイングが使用中のときに送信中のプロセス
- タイムアウト通知（TLE）

カウンタータイプ：増分。

MESSAGES-SENT

リンカープロセスが送信したメッセージの数。ユーザー I/O 操作（$RECEIVE 以外）と一部のシステムプロシージャーの呼び出しによって、このカウンターはインクリメントします。

カウンタータイプ：増分。

MQC-ALLOC-FAILURES

失敗した MQC 要求の数。

カウンタータイプ：増分。

MQC-ALLOCATIONS

プロセスに現在割り当てられている MQC の数。

カウンタータイプ：増分。

MQCS-INUSE-QTIME

MQC がプロセスに割り当てられていた時間。

カウンタータイプ：キュー。

MSGS-SENT-QTIME

プロセスで送信されるメッセージが未処理であった時間（マイクロ秒）。

カウンタータイプ：キュー。

NATIVE-BUSY-TIME

（ZMS 形式のみ）このプロセスで CPU がネイティブコードの実行中にビジー状態だった時間。

H シリーズ以降の PROCESS エンティティの使用上の注意(384 ページ)を参照してください。

カウンタータイプ：ビジー

OPEN-CLOSE-WAIT-TIME

プロセスがオープンおよびクローズ要求を待機する時間。OSS プロセスの場合、待機時間はネームサーバー操作を除き、ファイルのオープンおよびクローズ操作に対して累積されます。ネームサーバー操作は、OSSNS-BUSY-TIME でカウントされます。

カウンタータイプ：ビジー。

OSS-DEV-NULL-OPS

/dev/null として開かれたすべてのファイルの読み取りおよび書き込みの数。

カウンタータイプ：増分。

OSS-TTY-READ-BYTES

/dev/tty として開かれたすべてのファイルからプロセスで読み取られたバイト数。

カウンタータイプ：累積。

OSS-TTY-READS

/dev/tty のすべての OSS ファイルオープンによって実行された READ 操作の数。

カウンタータイプ：増分。

OSS-TTY-WAIT-TIME

/dev/tty として開かれたすべてのファイルへの要求をプロセスが待機する時間。

カウンタータイプ：ビジー。

OSS-TTY-WRITE-BYTES

/dev/tty として開かれたすべてのファイルにプロセスが書き込んだバイト数。

カウンタータイプ：累積。

OSS-TTY-WRITES

/dev/tty のすべての OSS ファイルオープンに対してプロセスが実行した WRITE 操作の数。

カウンタータイプ：増分。

OSSNS-DD-CALLS

OSS ネームサーバーに送信されたデータ定義要求の数。これらの API で次のカウンターをインクリメントします。

bind()	open(, O_CREAT)	mkfifo()	utime()
chmod()	link()	rename()	

表は続く

chown()	mkdir()	rmdir()
creat()	mknod()	unlink()

カウンタータイプ：増分。

OSSNS-MESSAGE-BYTES

OSS ネームサーバーへの要求で送受信されたメッセージデータのバイト数。

カウンタータイプ：累積。

OSSNS-REDIRECTS

OSS ネームサーバーに送信されたリダイレクト要求の数。

カウンタータイプ：増分。

OSSNS-REQUESTS

OSS ネームサーバーに送信された要求の数。要求には、FILEINFO 呼び出し、データ定義要求、ディレクトリ操作、解決要求があります。

カウンタータイプ：増分。

OSSNS-WAIT-TIME

プロセスがすべての OSS ネームサーバーへの要求を待機する時間。

カウンタータイプ：ビジー。

PAGE-FAULTS

プロセスによって生成されたページフォールトの数。

カウンタータイプ：増分。

PRES-PAGES-END

測定間隔の最後に測定対象のプロセスでスポンサリングされるメインメモリ内のページ数。

カウンタータイプ：スナップショット。

PRES-PAGES-QTIME

D シリーズの RVU で、プロセスが所有するページがメインメモリ内で費やした時間。

G シリーズ RVU で、測定対象のプロセスによってスポンサリングされるページがメインメモリ内で費やした時間。スポンサリングの説明については、**G シリーズ PROCESS エンティティの使用上の注意**(386 ページ)を参照してください。

カウンタータイプ：キュー。

PRES-PAGES-START

測定間隔の開始時に測定対象のプロセスでスポンサリングされるメインメモリ内のページ数。

カウンタータイプ：スナップショット。

PRIVSTACK-LIMIT

（ZMS 形式のみ、L02、J06 以降）特権スタックで許容される最大サイズ。

カウンタータイプ：スナップショット。

PRIVSTACK-SIZE-END

（ZMS 形式のみ、L02、J06 以降）測定間隔の終了時の特権スタックのサイズ。

カウンタータイプ：スナップショット。

PRIVSTACK-SIZE-START

（ZMS 形式のみ、L02、J06 以降）測定間隔の開始時の特権スタックのサイズ。このカウンターは、MEASCOM では表示されませんが、IF/BY 句とプロットに使用できます。

カウンタータイプ：スナップショット。

READY-TIME

測定対象のプロセスがレディーリストにある状態、または実行している状態で費やした時間。

測定オーバーヘッドを最小限に抑えるため、このカウンターは特定のシステムアクティビティ中にはオフになりません。このカウンターは次の時間をカウントします。

- プロセスのディスパッチに必要な時間。
- 実行中のプロセスに非同期で呼び出されるソフトウェア割り込みに費やした時間。
- メモリマネージャーによるページフォールトの処理の待機に費やした時間。（D シリーズ）
- メッセージシステムルーチンに費やした時間。（D シリーズ）
- プロセスがデータまたはメッセージを受信し、データ転送が完了しない場合に、低レベルのメッセージシステムルーチンに費やした時間。プロセスは続いて、レディーリストタイマーから外されずに一時停止されます。たとえば、ServerNet または FOX データの転送が完了するまで待機しているときに、プロセスは実行せず準備完了タイマーに残っていることがあります。（G シリーズ）

メモリの処理方法が変更されたため、G05 以降の RVU の READY-TIME カウンターの結果は、以前の測定とは直接比較できません。以前、プロセスのレディーリストタイマーは、ページフォールトを待機している間、メモリマネージャーが続いて必要なページを I/O で待機している場合でも、アクティブなままでした。G05 以降では、それぞれのプロセスが独自のページフォールト要求を処理します。必要なページの I/O の完了を待機している間に要求が停止した場合、準備完了タイマーはオフになります。

カウンタータイプ：ビジー。

注記:

Measure データは、PROCESS.CPU-BUSY-TIME が低く、全体のプロセッサーの利用率が低いのに、高い PROCESS.READY-TIME と CPU.CPU-QTIME の組み合わせを表すことがあります。G05 以前では、これは、ページフォールトの完了を待機している間に PROCESS.CPU-BUSY-TIME が一時停止したためでした。ただし、どの RVU でも、未完了のメッセージシステム転送が完了を試みるまで待機している間に、プロセスが一時停止することもあります。どちらの場合でも、PROCESS.READY-TIME カウンターと CPU.CPU-QTIME カウンターは累積され続けます。このメッセージシステム動作はめったに起こりませんが、これが起きると、おそらくネットワーク内のパスがダウンしたとマークされたり、ネットワーク内の基盤となるハードウェアに問題があったり、データのオーバーロードの結果、輻輳した FOX または ServerNet/FX ネットワークにシグナルが送られることがあります。

RECEIVED-BYTES

リスナープロセスが受信したメッセージバイト数。メッセージバイトは、メッセージを送信したリンカープロセスの SENT-BYTES カウンターにも含まれます。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RECEIVED-BYTES-F フィールドは、RECEIVED-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

RECEIVED-CBYTES

このプロセスの代わりにメッセージシステムが受信したプロトコル制御バイト数。このカウンターは、プロセスがリスナープロセスとして動作しているときに更新されます。プロトコル制御バイトは、カウンター RECEIVED-BYTES には加算されません。

カウンタータイプ：累積。

RECV-QTIME

メッセージがプロセスメッセージ入力キューの待機に費やした時間。

ユーザープロセスの場合、メッセージ入力キューは$RECEIVE です。

カウンタータイプ：キュー。

REPLY-BYTES

リンカープロセスからのメッセージに応答して、リスナープロセスが送信したメッセージのバイト数。メッセージバイトは、メッセージを受信したリンカープロセスの RETURNED-BYTES カウンターにも含まれます。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの REPLY-BYTES-F フィールドは、REPLY-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

REPLY-CBYTES

リンカープロセスからのメッセージに応答して、このプロセスの代わりにメッセージシステムが送信したプロトコル制御バイト数。このカウンターは、プロセスがリスナープロセスとして動作しているときに更新されます。プロトコル制御バイトは、カウンター REPLY-BYTES には含まれません。

カウンタータイプ：累積。

RETURNED-BYTES

リンカープロセスが受信したメッセージバイト数。メッセージバイトは、メッセージを送信したリスナープロセスの REPLY-BYTES カウンターにも含まれます。

このカウンターでは、ダイレクトバルク I/O で転送されるバイトは加算されません。DBIO のバイト数については、FILE エンティティタイプの DBIO-READ-BYTES カウンターのリンカープロセスに起因する測定値を参照してください。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの RETURNED-BYTES-F フィールドは、RETURNED-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

RETURNED-CBYTES

このプロセスの代わりにメッセージシステムが返したプロトコル制御バイト数。このカウンターは、プロセスがリンカープロセスとして動作しているときに更新されます。プロトコル制御バイトは、カウンター RETURNED-BYTES には加算されません。

カウンタータイプ：累積。

SENT-BYTES

リンカープロセスが送信したメッセージバイト数。メッセージバイトは、メッセージを受信したリスナープロセスの RECEIVED-BYTES カウンターにも加算されます。

このカウンターでは、ダイレクトバルク I/O で転送されるバイトは加算されません。DBIO のバイト数については、FILE エンティティタイプの DBIO-WRITE-BYTES カウンターのリンカープロセスに起因する測定値を参照してください。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、これは 64 ビットのカウンターです。

G シリーズの SENT-BYTES-F フィールドは、SENT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

SENT-CBYTES

このプロセスの代わりにメッセージシステムが送信したプロトコル制御バイト数。このカウンターは、プロセスがリンカープロセスとして動作しているときに更新されます。プロトコル制御バイトは、カウンター SENT-BYTES には加算されません。

カウンタータイプ：累積。

SWAPSPACE

（ZMS 形式のみ、L02、J06 以降）保証される KMSF スワップ領域のサイズ。

カウンタータイプ：スナップショット。

TLES-END

（ZMS 形式のみ、L02、J06 以降）測定間隔の最後で割り当てられた TLE の数。

カウンタータイプ：スナップショット。

TLES-START

（ZMS 形式のみ、L02、J06 以降）測定間隔の最初に割り当てられた TLE の数。

カウンタータイプ：スナップショット。

TNS-BUSY-TIME

CPU がこのプロセスで TNS コードの実行でビジー状態だった時間。このカウンターは、TNS/R または TNS/E システムで行われる測定にのみ適用されます。

H シリーズ以降の PROCESS エンティティの使用上の注意(384 ページ)を参照してください。

カウンタータイプ：ビジー。

VSEMS

セマフォ機能によってリソースが保護されている別のプロセスで使用されているリソースをプロセスが待機していた回数。

カウンタータイプ：増分。

X-INTERRUPTS

（ZMS 形式のみ、L01、H07/J05 以降）このプロセスで処理された X ファブリック割り込みまたは非ファブリック割り込みの数。

カウンタータイプ：増分。

Y-INTERRUPTS

（ZMS 形式のみ、L01、H07/J05 以降）このプロセスが処理した Y ファブリック割り込みの数。

カウンタータイプ：増分。

PROCESS エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

PAGE-SIZE-BYTES

（G シリーズ、Legacy 形式のみ）測定対象の CPU のメモリページフレーム内のバイト数。 **H シリーズ以降の PROCESS エンティティの使用上の注意**(384 ページ)も参照してください。

TNSR-PROCESS

（Legacy 形式のみ）ネイティブモードでのみ実行するプロセスに設定されているフラグ。

PROCESS エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

ABS-SEGS-QLEN-MAX

（Legacy 形式のみ）ABS-SEGS-QTIME で記述されるキューの項目の最大数。H シリーズ以降の RVU では、この値は常に 1 です。

カウンタータイプ：最大キュー。

EXT-SEGS-MAX

（Legacy 形式のみ）EXT-SEGS-QTIME カウンターで記述されるキューの項目の最大数。

カウンタータイプ：最大キュー。

LCB-ALLOC-FAILURES

（Legacy 形式のみ）再定義されました。MQC-ALLOC-FAILURES を参照してください。

LCB-ALLOCATIONS

（Legacy 形式のみ）再定義されました。MQC-ALLOCATIONS を参照してください。

LCBS-INUSE-QTIME

（Legacy 形式のみ）再定義されました。MQCS-INUSE-QTIME を参照してください。

MAX-LCBS-INUSE

（Legacy 形式のみ）再定義されました。MAX-MQCS-INUSE を参照してください。

MAX-MQCS-INUSE

（Legacy 形式のみ）キューの MQC の最大数。

カウンタータイプ：最大キュー。

MSGS-SENT-QLEN-MAX

（Legacy 形式のみ）MSGS-SENT-QTIME カウンターで記述されるキューの項目の最大数。H シリーズ以降の RVU では、この値は常に 1 です。

カウンタータイプ：最大キュー。

PRES-PAGES-MAX

（Legacy 形式のみ）PRES-PAGES-QTIME で記述されるキューの項目の最大数。

カウンタータイプ：最大キュー。

RECEIVED-BYTES-F

（Legacy 形式のみ）G シリーズの RVU では、RECEIVED-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

RECV-QLEN-MAX

（Legacy 形式のみ）RECV-QTIME カウンターで記述されるキューの項目の最大数。

カウンタータイプ：最大キュー。

REPLY-BYTES-F

（Legacy 形式のみ）G シリーズの RVU では、REPLY-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

RETURNED-BYTES-F

（Legacy 形式のみ）G シリーズの RVU では、RETURNED-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

SENT-BYTES-F

（Legacy 形式のみ）G シリーズの RVU では、SENT-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

TNSR-BUSY-TIME

（Legacy 形式のみ）プロセッサーがこのプロセスに対するネイティブコードの実行にビジー状態だった時間。このカウンターは、TNS/R システムで行われる測定にのみ適用されます。このカウンターは他のカウンターから派生します。

TNSR-BUSY-TIME = CPU-BUSY-TIME - (ACCEL-BUSY-TIME + TNS-BUSY-TIME)

カウンタータイプ：ビジー。

UCL-LOCK-MAX

（Legacy 形式のみ）使用されなくなりました。

UCL-LOCK-QTIME

（Legacy 形式のみ）使用されなくなりました。

UCL-MAX

（Legacy 形式のみ）

Measure G10 以降の PVU、ゼロ

Measure G09 以前の PVU では、UCL-QTIME カウンターで記述されているキューの項目の最大数。

カウンタータイプ：最大キュー。

UCL-QTIME

（Legacy 形式のみ）Measure G10 以降の PVU、ゼロ。最も近い同等表現については、**PRES-PAGES-QTIME**(378 ページ)を参照してください。ここには、ユーザーコードとデータフレームのほか、このプロセスによってメモリ内に存在するようになったシステムコードとデータフレームについて記されています。

Measure G09 以前の PVU で、ユーザーコードとライブラリページがマッピングされた時間。このカウンターは PRES-PAGES-QTIME カウンター（への追加ではなく）のサブセットです。

REPORT RATE OFF にした場合、このカウンターはメモリカテゴリのキュー時間の合計です。REPORT RATE ON にした場合、このカウンターは UCL ページの平均数です。

PRES-PAGES-QTIME（プロセス用にマッピングされたすべてのページの合計）から UCL-QTIME（合計のコードおよびライブラリページ）を減算することによってプロセスデータページ情報を取得できます。

カウンタータイプ：キュー。

All PROCESS エンティティの使用上の注意

プロセスをディスパッチする方法、実行するタイミングなどについて詳しくは、該当するシステム解説マニュアルを参照してください。

H05/J03 以降の PVU では、CPU 内の IPU に CPU 負荷がどのように分散されるかをユーザーが確認できるように、Measure では一連の IPU-BUSY-TIME カウンターを CPU エンティティに追加しました。プロセスが実行していた IPU をユーザーが確認できるように、IPU-NUM、IPU-SWITCHES、および IPU-NUM-PREV の各カウンターも PROCESS エンティティに追加されました。

測定または間隔中にプロセスが IPU を切り替えた場合、IPU-SWITCHES にはゼロ以外の値が指定され、行われた IPU 切り替えの数が示されます。IPU-SWITCHES がゼロ以外の場合、IPU-NUM-PREV は使用された最後の IPU を示します。CPU エンティティとは異なり、PROCESS エンティティには、IPU BUSY TIME カウンターはなく、CPU-BUSY-TIME のみがあります。したがって、IPU 切り替えが行われた間隔の間に、IPU-NUM と IPU-NUM-PREV、および以前の IPU（IPU-SWITCHES が 1 より大きい場合）に CPU-BUSY-TIME がどのように分布していたかは特定できません。

つまり、特定の IPU に対する CPU 内のすべての PROCESS CPU-BUSY-TIME を合計し（たとえば LIST PROCESS 0,*, IF IPU-NUM = 1, TOTALS ONLY, RATE OFF）、IPU の切り替えが行われた場合、この合計が CPU エンティティ内の対応する IPU-BUSY-TIME に一致する可能性は低くなります。

J シリーズ PROCESS エンティティの使用上の注意

J06.13 以降の RVU では、有効な IPU/CPU の数を動的に増やすことができます。IPU/CPU の変更に及ぶ PROCESS 測定は、初期 IPU 数を含みます。

H シリーズ以降の PROCESS エンティティの使用上の注意

- プロセスの優先順位は、プロセスの実行中に変更できます。測定におけるプロセスの優先順位の追跡により、Measure クライアントは、PROCESS エンティティごとに、測定間隔の開始時および終了時の特定のプロセスまたは一連のプロセスの優先順位を確認でき、アプリケーションの調整に役立てることができます。H06.25/J06.14 以降の RVU では、2 セットの PROCESS エンティティカウンターが、測定中のプロセス優先順位の変更を追跡するために実装されています（INITIAL-PRIORITY-START

および INITIAL-PRIORITY-END と、CURRENT-PRIORITY-START および CURRENT-PRIORITY-END）。

- INITIAL-PRIORITY-START および INITIAL-PRIORITY-END スナップショットカウンターは、測定間隔の開始時と終了時にプロセスのターゲットとなる（要求された）優先順位を反映します。初期優先順位は、ALTPRI TACL コマンドを使用するか、ALTERPRIORITY、PRIORITY、または PROCESS_SETINFO_の Guardian プロシージャーを使用してプロセス実行中に変更できます。たとえば、100 の優先順位で開始されたプロセスは、100 の初期優先順位があるとみなされます。上記のプロシージャーのいずれかを使用してそのプロセスの優先順位が 150 に変更された場合、その後、その初期優先順位は 150 になります。

  注記:

  プロセス元の初期優先順位値（作成優先順位とも呼ばれます）が、PROCESS ヘッダーの最初の行に表示され、ZMSPROC レコードの ID セクションの PRIORITY フィールドに格納されます。この PRIORITY 値は測定中には静的です。

- CURRENT-PRIORITY-START および CURRENT-PRIORITY-END スナップショットカウンターは、測定間隔の開始時と終了時に現在の実行優先順位を反映します。プロセス制御によってプロセスの優先順位がデクリメントされている可能性があるので、現在の優先順位は初期優先順位と異なる場合があります。たとえば、150 の初期優先順位から始まった CPU バウンドプロセスが、実行し、その後、I/O バウンドになったときに（初期優先順位に一致する）150 の現在の優先順位に戻ると、その現在の優先順位は 149、148 と低下することがあります。

間隔を使用して PROCESS エンティティ測定を構成し、続いて、INITIAL-PRIORITY-START/INITIAL-PRIORITY-END および CURRENT-PRIORITY-START/CURRENT-PRIORITY カウンターをプロットすると、測定間隔にわたって 1 つのプロセスまたは一連のプロセスの優先順位の変化の履歴的な表示を提供できます。

注記:

実行している測定中に、INITIAL-PRIORITY-START/INITIAL-PRIORITY-END と CURRENT-PRIORITY-START/CURRENT-PRIORITY-END のカウンター値のペアを、LISTACTIVE、MEASREADACTIVE、MEAS_READACTIVE_、または MEAS_READACTIVE_MANY_を使用して表示すると、START カウンターにゼロの値が、END カウンターに既存の値が生成されます。

- PROCESS エンティティの PAGE-SIZE-BYTES フィールドは、H シリーズ以前の Measure データファイルでは Legacy 形式のレコードでのみサポートされます。このフィールドは、Measure G12 以降の G シリーズ PVU ではまだ入力できますが、H シリーズ以降の RVU では廃止されています。
- H シリーズ以降の RVU では、すべてのバイト数フィールドで 64 ビットに対応しています。-F で終わるフィールド名は、ZMS 形式レコードではもはや使用されていませんが、Legacy 形式でデータを要求するアプリケーションでは引き続き使用できます。
- 最大数を報告するカウンター（名前に MAX が含まれるカウンター）は、Legacy 形式でのみ有効です。H シリーズ以降の RVU では、このようなカウンターの値は 1 です。
- Measure H02、H03、H04、J01、および J02 の PVU では、ZMS 形式の PROCESS 表示で、2 のサブシステムバージョンが報告されます。
- Measure H05/J03 以降の PVU では、ZMS 形式の PROCESS 表示で、3 のサブシステムバージョンが報告されます。
- Measure H06/J04 以降の PVU では、ZMS 形式の PROCESS および ZMS 形式の CPU で、LOCKED-PAGES-END と LOCKED-PAGES-START のカウンターがサポートされます。これらのカウンターを使用するときには、次の使用上の留意事項に注意してください。
  - CPU のすべての PROCESS エンティティでロックされていると報告されたページフレームの合計は、対応する CPU エンティティでロックされていると報告されたページフレームと同じである必要はありません。
  - ページフレームは、メモリ内に置かれ、存続するようにマークされると、ロックされます。このようなマークがすべて削除されたり、ページフレームが割り当て解除されると、ロック解除されます。

- ロックされたページフレームは最初、それをロックしたプロセスに関連付けられています（スポンサリングされています）。ページが複数のプロセス間で共有され、元のプロセスが終了した場合、ページはスポンサリング解除され、その後、ページを共有している残りのプロセスのいずれかによってスポンサリングされるようになります。ページフレームがグローバルアドレス空間にマップされており、どのプロセスの一意のアドレス空間にもない場合、そのページはメモリマネージャープロセス PIN 1 によってスポンサリングされるようになり、このプロセスはまた、システムによってシステム用に割り当てられたページもスポンサリングします。
- ロックされたページフレームがスポンサリング解除される（どのプロセスにも関連付けられなくなる）と、そのページフレームは、メモリマネージャープロセスが次のルーチンに達するまでスポンサリング解除されたままになります。

- PROCESS エンティティの最初のレコードが作成されるのは、プロセスが開始されたときであり、測定の開始時ではありません。アクティブな取得中には、初期レコードと現在のレコードの間のデルタ時間を考慮して、現在のカウンター値が表示されます。プロセスが実行している CPU で PROCESSH 測定がアクティブである場合にのみ、ACCEL-BUSY-TIME、NATIVE-BUSY-TIME、および TNS-BUSY-TIME のカウンターは計算され、返され、表示されます。そうでない場合、カウンターは表示されず、返されたレコードの値はゼロになります。プロセスが開始してから PROCESSH 測定がアクティブになっていない場合（これはほとんどのシステムプロセスで起こりえます）、カウンターは正しくなくなります。

  これらのカウンターは、該当するコード領域で観察された PROCESSH サンプルの数に基づいて計算されるので、より正確な数値を得るために、PROCESSH サンプルの頻度を増やすことができます。CPU エンティティの PROCESSH-SAMPLES カウンターは、現在のサンプリング頻度を報告します。

  Measure H04/J02 以降の PVU で有効ですが、測定間隔の開始から終了まで PROCESSH サンプル数が変更していない場合、PROCESSH 測定がアクティブであっても、MEASCOM で、ACCEL-BUSY-TIME、NATIVE-BUSY-TIME、および TNS-BUSY-TIME のフィールドは表示されません。LISTALL コマンドでは、これは、これらのフィールドがある間隔で表示され、他の間隔では表示されないことを意味します。

  MEASCOM コマンド LISTACTIVE FOR *time* は、当該の期間全体にわたって PROCESSH 測定が実行していた場合にのみ、ACCEL-BUSY-TIME、NATIVE-BUSY-TIME、および TNS-BUSY-TIME のカウンターの正しい値がもたらされます。
- INITIAL-PRIORITY-START、INITIAL-PRIORITY-END、CURRENT-PRIORITY-START、および CURRENT-PRIORITY-END のカウンターは、TOTALS レコードでは意味がありません。H06.28/J06.17 以降の RVU の TOTALS レコードでは、これらは MEASCOM によって表示されません。

G シリーズ PROCESS エンティティの使用上の注意

- -F で終わる 64 ビットのバイト数フィールドは、既存の 32 ビットのバイト数フィールドと同じデータを収集します。たとえば、64 ビットフィールドの RETURNED-BYTES-F は、32 ビットフィールドの RETURNED-BYTES と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

  32 ビットフィールドは、現在アクティブであり、値を返し続けます （フィールドオーバーフローは、オーバーフローしたフィールドと測定対象のエンティティの ERROR フィールドの両方で、-1 の値で示されます）。ただし、64 ビットフィールドを使用するようにアプリケーションを変換する必要があります。32 ビットのフィールドは、今後の RVU では使用できなくなる可能性があります。

  MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットのフィールドの名前を使用します。たとえば、LIST PROCESS BY RETURNED-BYTES-F ではなく LIST PROCESS BY RETURNED-BYTES コマンドを発行します。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。
- G05 以降の RVU では、メモリ処理における変更で、メモリページのスポンサリングとスポンサリング解除の概念が導入されました。ページを存在させるプロセス（いつもではないが通常は、セグメントを定義するプロセス）によって、ページはメモリ内でスポンサリングされます。共有ページのスポンサーであるプロセスが終了すると、ページは一時的にスポンサリング解除されます。スポンサリング解除されたページを参照する次の共有プロセスが、そのページのスポンサーになります。

PRES-PAGES-QTIME カウンターは、測定対象のプロセスによりスポンサリングされたページに関する情報を提供します。

- さまざまなタイプのメッセージシステム転送プロトコルおよび概念（事前プッシュ、事後プル、リンカー、およびリスナー）の説明については、*NonStop S-Series Server Description Manual* を参照してください。

PROCESSH

PROCESSH エンティティは、プログラム内の 1 つまたは複数のコード範囲の相対的な実行時間に関する情報を提供します。

プロシージャー名とアドレス範囲を指定するか、コード領域カテゴリ（ユーザーコード、ユーザーライブラリ、システムコード、またはシステムライブラリ）を指定することにより、コード範囲を測定できます。

複数のディスクファイルに適用されるファイル名でワイルドカードを使用する場合（$SYSTEM.WORK.* など）、プロシージャー名やアドレス範囲ではなくコード領域カテゴリを指定する必要があります。

PROCESSH エンティティの PROCESS エンティティを明示的に追加する必要はありません。PROCESSH エンティティを測定に追加する場合は、測定対象のプロセスの対応する PROCESS エンティティが自動的に追加されます。PROCESS エンティティを使用して、ALLTIME と ALLINTR の 2 つの測定値を除き、測定対象のプロセスに関する詳細を得ることができます。Measure はこれらの測定値のそれぞれに PROCESS レコードを作成しますが、これらの 2 つの測定値は特定のプロセスの測定を含まないため、これらの PROCESS レコードはプレースホルダーに過ぎません。

注記:

H シリーズ以降の RVU では、PROCESSH サンプリングの ALLINTR オプションは無効です。H シリーズ以降の RVU を実行しているシステムで、すべての割り込みハンドラーは、別々の割り込み/補助プロセスであるため、個別にサンプリングし測定できます。

Measure G09 以降の PVU では、PROCESSH エンティティは、Guardian ファイル名に代わる OSS ファイルパス名の使用をサポートします。G09 PROCESSH は PROCESSH レポートで OSS ファイルパス名を表示し、外部の構造化レコードと新しい OSSNAMES 構造化レコード出力との直接のマッピングを提供します。

G09 PVU PROCESSH エンティティのエンティティ指定は、D シリーズおよび G09 以前の PVU エンティティの構文とは異なります。DDL レコードも異なります。

UNMATCHED SAMPLES 32-BIT と UNMATCHED SAMPLES 64-BIT の 2 つのビンは、Java プログラムなど、データセグメントで実行するコードのサンプルを保持しています。

トピック

PROCESSH エンティティのエンティティ指定構文

PROCESSH エンティティの DDL レコード

PROCESSH エンティティの DDL フィールドの定義

PROCESSH エンティティの使用上の注意

PROCESSH 測定の例

PROCESSH エンティティのエンティティ指定構文

PROCESSH エンティティを記述するには、次の構文を使用します。

```
PROCESSH entity-spec
```

PROCESSH

1 つまたは複数のプロセス内のコード範囲に関する情報を収集します。

entity-spec

次のいずれかです。

```
{ process-spec ( code-file-spec ) [, ( code-file-spec ) ] ...          }
{ process-spec [, process-spec ] ...                                   }
{ ALLTIME | ALLINTR cpu ( code-file-spec ) [, ( code-file-spec ) ] ... }
```

指定する値は次のとおりです。

process-spec

プロセス指定です。*process* は次のいずれかのように指定します。

```
{ *                                                    }
{ pid                                                  }
{ $process-name [ ( pid ) ]                            }
{ [[$device.]subvolume.]filename[:crvsn] [ ( pid ) ]   }
{ "pname" [ ( pid ) ]                                  }
```

指定する値は次のとおりです。

*****

ローカルシステム上のすべてのプロセスを測定します。

pid

測定対象のプロセスのプロセス ID です。pid は次のいずれかのように指定します。

```
{ cpu,pin          }
{ SYSTEM-PROCESSES }
```

cpu

測定対象のプロセスが実行されている CPU の番号です。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

pin

測定対象のプロセスのプロセス識別番号です。すべてのプロセス識別番号を指定するには、アスタリスク（*）を使用します

SYSTEM-PROCESSES

SYSGEN プログラムを使用してインストールしたすべてのプロセスを測定します。

$process-name

測定対象のプロセスの名前です。すべてのプロセスを指定するには、アスタリスク（*）を使用します。

$*device*

測定対象のプロセスのオブジェクトファイルが置かれているボリューム（デバイス）です。すべてのボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのボリュームです。

subvolume

測定対象のプロセスのオブジェクトファイルが置かれているサブボリュームです。すべてのサブボリュームを指定するには、アスタリスク（*）を使用します。

filename

測定対象の実行中のプロセスのオブジェクトファイルの名前です。すべてのファイル（一時ファイルを除く）を指定するには、アスタリスク（*）を使用します。

:*crvsn*

（Measure G09 以降の PVU）一意のファイル名を形成するために必要な、タイムスタンプ、作成バージョンのシリアル番号（CRVSN）、またはファイル名拡張子です。ファイルの CRVSN を取得するには、Measure のレポートを表示するか、LISTGNAME コマンドを使用します。

"*pname*"

（Measure G09 以降の PVU）完全修飾または部分修飾のいずれかの OSS ファイルパス名です。スラッシュ（/）で始まらない OSS ファイルパス名は、部分的なパス名とみなされ、OSSPATH の現在の設定を前に指定することで拡張されます。/G で始まる OSS ファイルパス名はサポートされていないため、代わりに Guardian ファイル名を使用してください。

注記:

OSS ファイルパス名は大文字と小文字が区別され、二重引用符（""）で囲んで指定する必要があります。有効な OSS ファイルパス名は、特定のファイル、または特定のディレクトリ内のファイルのセットを参照できます。ディレクトリが指定されている場合は、そのディレクトリ内のファイルのみが含まれます。指定したディレクトリの下位のディレクトリ内のファイルは含まれません。

code-file-spec

コードファイル指定です。*code-file-spec* は次のように指定します。

```
(  code-space  code-file  )
```

指定する値は次のとおりです。

code-space

コード領域指定です。Measure G12 以降の PVU では、*code-space* は TNS および高速化された TNS コードに対してのみ必須です。それ以外の場合、このパラメーターは受け入れられますが、無視されます。

高速化された TNS コードの場合、*code-space* 指定は次のとおりです。

領域	指定	高速化	TNS
ユーザーコード	UC[.*n*]、*n* は次の範囲内です。	0～31	0～15
ユーザーライブラリ	UL[.*n*]、*n* は次の範囲内です。	0～31	0～15

表は続く

領域	指定	高速化	TNS
システムコード	SC[.*n*]、*n* は次の範囲内です。	0～31	0
システムライブラリ	SL[.*n*]、*n* は次の範囲内です。	0～31	0～31

n が指定されていない場合、その値はデフォルトで指定したタイプのすべてのコード領域に設定されます。

TNS システムの場合、タイプ UC および UL のコード領域の数は累積的で、合計で 32 を超えることはできません。このため、許可される UL タイプのコード領域の最大数は、指定した UC タイプのコード領域の数が 0～15 の範囲内にある場合にのみ 16（0～15）です。UC タイプのコード領域の数が 16 を超える（16～31 の範囲内にある）場合、UL タイプのコード領域の最大数は 31 マイナス UC.*n* です。たとえば、UC.*n* が 25 に等しい場合、指定できる最大の UL.*n* は 6、つまり、31 - 25 = 6 です。

ネイティブコードの場合、*code-space* 指定は次のとおりです。

領域	指定
ユーザーコード	UCR
ユーザーライブラリ	ULR
システムコード	SCR
システムライブラリ	SLR

code-file

TNS オブジェクトファイル、EDIT ファイル、または Executable and Linking Format（ELF）オブジェクトファイル（TNS/X ファイルコード 500、TNS/R ファイルコード 700、または TNS/E ファイルコード 800）の Guardian ファイル名、または、Measure G09 以降の PVU の場合は、Guardian ファイル名を使用して指定できる同じセットのファイルタイプのいずれかを指定する OSS ファイルパス名（"*pname*"）です。

code-file がオブジェクトファイルの場合、MEASFH はプロシージャー名とアドレスを調べて、各プロシージャーを構成に追加します。*code-file* が EDIT ファイルの場合は、次のような形式で、アドレスの昇順でリストした名前/アドレスのタプルのセットを含める必要があります。

code-range-name code-range-address

指定する値は次のとおりです。

code-range-name

1 から 1024 の英数字、曲折アクセント符号（^）、ハイフン（-）、またはアンダースコア（_）のコード範囲（プロシージャー）名です。最初の文字には、英字、曲折アクセント符号、またはアンダースコアを使用できます。TNS と高速化されたコードを除いて、最初の文字にドル記号（$）を使用することもできます。

code-range-address

TNS コードを記述する EDIT（コード 101）ファイルでは、*code-range-address* は 0 から 177777 の範囲内の 8 進数のアドレスまたはアドレスのオフセットです。

TNS/X、TNS/R、または TNS/E コードを記述する EDIT（コード 101）ファイルの場合、*code-range-address* はシステムで有効な 16 進数の仮想アドレスまたはアドレスのオフセットです。前の *code-range-address* からのオフセットを示すには、8 進数または 16 進数の前にプラス記号（+）を付けます。プラス記号を含まない最後の *code-range-address* がオフセットに加算されて、有効なアドレスが求められます。

EDIT ファイルの最初の行には、キーワード OBJECT とその後の Guardian ファイル名または OSS パス名で構成されるオブジェクトファイル名指定子が含まれている必要があります。Guardian ファイル名の例は OBJECT $A.B.MYAPP で、OSS パス名の例は OBJECT "/a/b/myapp"です。

注記:

Measure H02 以降の PVU では、EDIT ファイルを使用した PROCESSH 測定のコード範囲の指定のサポートは、TNS 実行可能ファイルでは使用できなくなりました。

ALLTIME

割り込みを含む、すべてのプロセスの実行時間の測定を指定します。

Measure G12 より前の PVU では、ALLTIME を指定する場合は、TNS のコードファイル測定に対して *code-space* を SC.*n* または SL.*n* として指定する必要があります。Measure G12 以降の PVU では、TNS および高速化されたコードに対してのみ *code-space* を指定してください。（必要でないときに *code-space* を指定すると、パラメーターは受け入れられますが、無視されます。）

H シリーズ以降の RVU では、ALLTIME を指定する場合、指定する *code-space* はシステムライブラリである必要があります。次に例を示します。

```
+ ADD PROCESSH ALLTIME 3 ($SYSTEM.SYS02.INITDLL)
+ ADD PROCESSH ALLTIME 3 ($SYSTEM.ZDLL009A.ZCREDLL)
```

ALLINTR

すべてのシステム割り込みの実行時間の測定を指定します。

Measure G12 より前の PVU で ALLINTR を指定する場合、*code-space* を SC.*n*、TNS のコードファイル測定の場合は SL.*n* と指定する必要があります。Measure G12 以降の PVU では、TNS および高速化されたコードに対してのみ *code-space* を指定してください。（必要でないときに *code-space* を指定すると、パラメーターは受け入れられますが、無視されます。）

このオプションは、H シリーズ以降の RVU では無効です。**H シリーズ以降の PROCESS エンティティの使用上の注意**(384 ページ)を参照してください。

cpu

測定対象のプロセスが実行されている CPU の番号です。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルト値はありません。

PROCESSH エンティティの DDL レコード（ZMS 形式）

PROCESSH エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

ID フィールドの DDL 定義

```
DEFINITION zmsproch-id.
 02 pin                          type binary 16 unsigned.
 02 priority                     type binary 16 unsigned.
 02 userid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 creatorid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 process-name                 type character 8.
 02 program-file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 osspid                       type binary 32 unsigned.
 02 ancestor-cpu                 type binary 16 unsigned.
 02 ancestor-pin                 type binary 16 unsigned.
 02 ancestor-sysname             type character 8.
 02 ancestor-process-name        type character 8.
 02 device-name                  type character 8.
 02 program-file-name-MID.
    03 PATHID                    type character 24.
    03 CRVSN                     type character 6.
 02 code-space                   type binary 16 unsigned.
 02 object-device-name           type character 8.
 02 object-file-name-MID.
    03 PATHID                    type character 24.
    03 CRVSN                     type character 6.
 02 code-range-flags             type binary 16 unsigned.
 02 code-range-length            type binary 16 unsigned.
 02 reserved-1                   type character 6.
 02 code-range                   type character 1024.
 end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsproch-ctrs.
 02 process-busy-samples         type binary 32 unsigned.
 02 code-space-busy-samples      type binary 32 unsigned.
 02 code-range-busy-samples      type binary 32 unsigned.
 02 accel-busy-samples           type binary 32 unsigned.
 02 tns-busy-samples             type binary 32 unsigned.
 02 native-busy-samples          type binary 32 unsigned.
end
```

DDL レコード記述フィールド

```
RECORD zmsproch. FILE is "zmsproch" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 ctr                          type zmsproch-ctrs.
 02 id                           type zmsproch-id.
end
```

PROCESSH エンティティの DDL レコード（Legacy 形式）

PROCESSH エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

```
RECORD processh.  FILE is "processh" ENTRY-SEQUENCED.
                        .
                        .
                        .
error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)
                        .
                        .
                        .
* entity identification items:
 02 pin                          type binary 16 unsigned.
 02 process-name                 type character 8.
 02 program-file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 priority                     type binary 16 unsigned.
* code range identifiers and value:
 02 process-busy-samples         type binary 32 unsigned.
 02 code-space                   type binary 16 unsigned.
 02 code-space-busy-samples      type binary 32 unsigned.
 02 code-range                   type character 32.
 02 code-range-busy-samples      type binary 32 unsigned.
* new entity identification field:
 02 userid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 creatorid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
* fields for Liberty specific counters as of C30:
 02 accel-busy-samples           type binary 32 unsigned.
 02 tns-busy-samples             type binary 32 unsigned.
* new entity identification items for D10:
 02 ancestor-cpu                 type binary 16 unsigned.
 02 ancestor-pin                 type binary 16 unsigned.
 02 ancestor-sysname             type character 8.
 02 ancestor-process-name        type character 8.
* Native Mode busy samples:
 02 tnsr-busy-samples            type binary 32 unsigned.
* SMS changes:
 02 device-name                  type character 8.
* new identifiers for OSS file pathname support:
 02 osspid                       type binary 32 unsigned.
 02 program-file-name-mid.
    03 pathid                    type character 24.
    03 crvsn                     type character 6.
 02 object-device-name           type character 8.
 02 object-file-name-MID.
    03 PATHID                    type character 24.
    03 CRVSN                     type character 6.
 02 reserved-1                   type character 6.
 02 code-range-flags             type binary 16 unsigned.
```

```
 02 code-range-length          type binary 16 unsigned.
 02 code-range                 type character 1024.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

PROCESSH エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧にしています。サブフィールドは、フィールドに指定する順に一覧にしています。

ANCESTOR-CPU

親プロセスが存在する CPU の番号。

ANCESTOR-PIN

親プロセスのプロセス識別番号。

ANCESTOR-PROCESS-NAME

親プロセスの名前。レポートヘッダーには、親の情報が表示されます。この情報を取得できない場合は、「unknown」という文字列が表示されます。

ANCESTOR-SYSNAME

親プロセスが存在するシステムの名前。

CODE-RANGE

測定対象のコード範囲の名前。

CODE-RANGE-FLAGS

現在は使用されていません。

CODE-RANGE-LENGTH

現在は使用されていません。

CODE-SPACE

これらのコード領域識別子のいずれかです。

0	UC.*n*	（ユーザーコード—TNS または高速化）
1	UCR	（ユーザーコード—TNS/R ネイティブ）
2	UL.*n*	（ユーザーライブラリ—TNS または高速化）
3	ULR	（ユーザーライブラリ—TNS/R ネイティブ）

表は続く

4	SC.0	（システムコード—TNS または高速化）
5	SCR	（システムコード—TNS/R ネイティブ）
6	SL.*n*	（システムライブラリ—TNS または高速化）
7	SLR	（システムライブラリ—TNS/R ネイティブ）

CREATORID

作成者アクセス ID（CAID）。このフィールドはプロセスの作成者を識別し、GROUP と USER の 2 つのサブフィールドに分かれています。

DEVICE-NAME

プログラムファイルが配置されているディスクデバイス。SMF ファイルの場合、このフィールドには、論理ファイル名に対応する物理的な場所が表示されます。NonStop ファイルの場合、このフィールドは、FILE-NAME VOLUME サブフィールドと同じです。

OBJECT-DEVICE-NAME

現在は使用されていません。

OBJECT-FILE-NAME-MID

現在は使用されていません。

OSSPID

Measure G09 以降の PVU で、プロセスが OSS 環境で実行される場合、このフィールドには OSS プロセス ID が含まれます。別のプロセスの場合、このフィールドにはゼロが含まれます。

PIN

プロセスのプロセス識別番号。

PRIORITY

プロセスの優先順位。

PROCESS-NAME

プロセスの名前。

PROGRAM-FILE-NAME

プロセスが実行しているオブジェクトファイルの名前。この名前は、NonStop ファイルか SMF ファイルのいずれかに適用できます。PROGRAM-FILE-NAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、プログラムファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、プロセスが実行されたときに指定された物理ファイル名を表します。 SMF ファイルでは、プロセスが実行されたときに使用された、位置に依存しない論理ファイル名を表します。

PROGRAM-FILE-NAME-MID

PATHID	指定した PROGRAM-FILE-NAME の OSS ファイル名の内部形式表現です。その他のファイルでは、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。その他のファイルでは、このフィールドにはゼロが含まれます。

Measure G09 以降の PVU では、指定した PROGRAM-FILE-NAME の OSS ファイル名の内部形式表現です。PROGRAM-FILE-NAME-MID には次の 2 つのサブフィールドがあります。

USERID

プロセスのプロセスアクセサー ID（PAID）。

PROCESSH エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧しています。

ACCEL-BUSY-SAMPLES

サンプリング操作で、測定対象のコードによる高速化されたコードの実行が検出された回数。このカウンターは、TNS/X、TNS/R、または TNS/E システムで行われる測定にのみ適用されます。ACCEL-BUSY-SAMPLES + TNS-BUSY-SAMPLES = CODE-RANGE-BUSY-SAMPLES。

カウンタータイプ：サンプリング。

CODE-RANGE-BUSY-SAMPLES

サンプリング操作で測定対象のコード範囲の実行が検出された回数。

TNS および高速化されたコードを実行している TNS/X、TNS/R、および TNS/E システムでは、CODE-RANGE-BUSY-SAMPLES=ACCEL-BUSY-SAMPLES + TNS-BUSY-SAMPLES です。

TNS/R ネイティブコードを実行している TNS/R および TNS/E システムでは、CODE-RANGE-BUSY-SAMPLES = TNSR-BUSY-SAMPLES です。

TNS/E ネイティブコードを実行している TNS/E システムでは、CODE-RANGE-BUSY-SAMPLES = NATIVE-BUSY-SAMPLES です。

カウンタータイプ：サンプリング。

CODE-SPACE-BUSY-SAMPLES

サンプリング操作で測定対象のコード領域の実行が検出された回数。

カウンタータイプ：サンプリング。

NATIVE-BUSY-SAMPLES

（ZMS 形式のみ）サンプリング操作で、測定対象のコードによるネイティブコードの実行が検出された回数。

カウンタータイプ：サンプリング。

PROCESS-BUSY-SAMPLES

サンプリング操作でプロセスの実行が検出された回数。

カウンタータイプ：サンプリング。

TNS-BUSY-SAMPLES

サンプリング操作で、測定対象のコードによる TNS コードの実行が検出された回数。このカウンターは、TNS/X、TNS/R、または TNS/E システムで行われる測定にのみ適用されます。ACCEL-BUSY-SAMPLES + TNS-BUSY-SAMPLES = CODE-RANGE-BUSY-SAMPLES.

カウンタータイプ：サンプリング。

PROCESSH エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

このカウンターフィールドは、Legacy 形式の DDL レコードでのみ使用されます。

TNSR-BUSY-SAMPLES

（Legacy 形式のみ）サンプリング操作で、測定対象のコードによる TNS/R ネイティブコードの実行が検出された回数。このカウンターは、TNS/R システムで行われる測定にのみ適用されます。TNSR-BUSY-SAMPLES = CODE-RANGE-BUSY-SAMPLES.

カウンタータイプ：サンプリング。

H シリーズ以降の PROCESSH エンティティの使用上の注意

- H シリーズ以降の RVU では、PROCESSH サンプリングの ALLINTR オプションは無効です。H シリーズ以降の RVU を実行しているシステムで、すべての割り込みハンドラーは、別々の割り込み/補助プロセスであるので、個別にサンプリングし測定できます。ALLINTR PROCESSH 測定が構成されていない場合、エラーは返されませんが、レコードは返されません。割り込み/補助プロセスは、PROCESS レコード内のその優先順位の値で識別できます。これらのプロセスのみで PRI 値が 255 になります。LIST PROCESS 0,*, IF PRI = 255 の MEASCOM コマンドは、すべてのプロセスで測定が終了した場合、プロセッサー 0 で実行しているすべての割り込み/補助プロセスのリストを返します。これらのレコードからの情報（プログラムファイル名、ピンなど）を使用して、これらの項目に関する PROCESSH 測定を必要に応じて構成できます。
- H シリーズ以降の RVU では、PROCESSH サンプリングレートが 1 秒当たり約 500 サンプルに増やされています。サンプリングレートは J シリーズの RVU では IPU 単位です。このサンプリングレートは固定されていません。短い期間で多くのサンプルが必要な場合は、レートを増やすことができます。ただし、これは他のプロセスのパフォーマンスに悪影響を及ぼす可能性があるので、この機能は必要な場合にのみ使用してください。PROCESSH サンプリングレートを増やすには、START MEASSUBSYS コマンドを発行する前に、この DEFINE 文を TACL セッションコンテキストに追加します。指定したファイル名により、サンプリングレートのスケーリング係数が決まります。

DEFINE 文	スケーリング係数
ADD DEFINE =_SET_PROCESSH_RATE, file SCALE2X	デフォルトレートの 2 倍
ADD DEFINE =_SET_PROCESSH_RATE, file SCALE4X	4 倍
ADD DEFINE =_SET_PROCESSH_RATE, file SCALE8X	8 倍
ADD DEFINE =_SET_PROCESSH_RATE, file SCALE16X	16 倍
ADD DEFINE =_SET_PROCESSH_RATE, file SCALE32X	32 倍

おおよそのサンプリングレートを決定するには、プロセッサー内で連続した PEEK INT 要求を発行し、累積したサンプル割り込みを表示するか、ZMSCPU レポートで、または MEASCOM LIST CPU コマンドを通じてカウンター PROCESSH-SAMPLES を調べます。次に例を示します。

```
TACL 12>  ADD DEFINE  =_SET_PROCESSH_RATE, file SCALE4X
TACL 13>  MEASCOM START MEASSUBSYS
```

- *process-spec* が *process-name* か *cpu,pin* の組み合わせのどちらかである場合、*code-file-spec* を指定して PROCESSH 測定を構成する必要があります。オブジェクトファイル名が *process-spec* と指定されている場合、*code-file-spec* はオプションです。LIST コマンドの場合は、*code-file-spec* を指定する必要はありません。

G シリーズ PROCESSH エンティティの使用上の注意

- TNS/R プロセッサーには、非ネイティブからネイティブへの移行呼び出しを処理するゲートウェイと呼ばれる短いコードシーケンスが含まれています。サンプリング対象のプロセスがゲートウェイで実行されている場合、PROCESSH は、サンプル数を増加しますが、特定のコード領域にそのサンプルを割り当てません。このため、サンプルの合計数が各種の *code-space* 数の合計よりも大きくなることがあります。
- 個々の *code-space* サンプル数の合計がプロセスのサンプルの合計数を大幅に下回る場合、ネイティブオブジェクトファイルツール（NOFT）ユーティリティを使用して、共有実行時ライブラリ（SRL）が PROCESSH 構成に追加されている必要があるかどうかを確認します。その場合は、MEASCOM で URL としてそれらを PROCESSH 構成に追加します。例については、**PROCESSH 測定の例**を参照してください。

> **注記:**
>
> サンプリングはプロセッサーミリコードやゲートウェイでは行われないので、少数のサンプルが欠落する可能性があります。

PROCESSH 測定の例

- この例では、CPU 7 でのすべてのプロセスによるシステムプロシージャーの使用を追跡します。ここでは TNS/R システムについてのシステムコードとシステムライブラリアクティビティの使用を示し、TNS/R システム上で実行されるすべてのシステムプロシージャーのサンプリングレポートが生成されます。ALLTIME を使用することにより、CPU 内のすべてのプロセスについてのすべてのシステムコードとシステムライブラリアクティビティが測定されます。

```
+ ADD PROCESSH ALLTIME 7 (SC $SYSTEM.SYSnn.TSC)
+ ADD PROCESSH ALLTIME 7 (SL $SYSTEM.SYSnn.TSL)
```

> **注記:**
>
> TNS/R システムには、1 つ以上のシステムコード（SC）ファイルがあります。

- この例では、SCR および SLR のコード領域を測定します。

TNS/R オペレーティング環境をサポートするために、SYSGEN プログラムは TSYSCLR という名前の新しいオブジェクトファイル（ファイルコード 700）を作成し、これは$SYSTEM.SYSnn サブボリュームに配置されます。このオブジェクトファイルは、TNS/R システムコード（SCR）と TNS/R システムライブラリ（SLR）のサンプリングに使用できます。TNS/R コード領域にはコードマップが含まれていないため、SCR および SLR についてのサンプルのみが表示されます。この一連の例で後述するように、TSYSCLR ファイルは H シリーズ以降の RVU で置き換えられます。

SCR および SLR についての TNS/R コードをサンプリングするには、次のようにします。

```
+ ADD PROCESSH proc spec (SCR $SYSTEM.SYSnn.TSYSCLR)
+ ADD PROCESSH proc spec (SLR $SYSTEM.SYSnn.TSYSCLR)
```

- この例では、プログラム内の共有ランタイムライブラリ（SRL）を確認する方法と、これらをプログラムの PROCESSH 測定に含める方法について説明します。

  1. ネイティブオブジェクトファイルツール（NOFT）ユーティリティを使用して、特定のプログラムに関連付けられているすべての SRL を識別します。
  2. SRL をユーザーライブラリファイル（URL）として MEASCOM に追加します。

     プログラムについての PROCESSH の合計サンプル数のアカウンティングおよびマッピングは、個々のコード領域サンプルカウントの合計と、現在できるだけ近くなるようにされています。

  プログラムで SRL を使用する場合、次のようになります。

  1. NOFT LISTSRLINFO コマンドを使用して、SRL を識別します。

```
noft> FILE $VOL.SUBVOL.FILENAME

              Object File : \node.$system.sysnn.ztcpsrl
              File Format : ELF
                    Scope : (none)
                     Case : Sensitive
   noft> LISTSRLINFO

                 SRL Name :      #
                  ZLANCSRL:      0
                  ZLANDSRL:      1
```

  2. MEASCOM で、次の SRL を ULR としてプログラムの PROCESSH 構成に追加します。

```
+ ADD PROCESSH $VOL.SUBVOL.FILENAME (UCR FILE-NAME)
+ ADD PROCESSH $VOL.SUBVOL.FILENAME (ULR ZLANCSRL)
+ ADD PROCESSH $VOL.SUBVOL.FILENAME (ULR ZLANDSRL)
```

- H シリーズ以降の RVU には TSYSCLR ファイルが存在しません。このファイルは、2 つのシステム DLL である INITDLL および MCPDLL に置き換えられます。この例は、H シリーズ以降のライブラリ構造を反映しています。コード領域識別子 SL は、非ネイティブのシステムライブラリコードを指定します。

```
+ ADD PROCESSH $SYSTEM.SYS01.TSYSDP2 (2, *) ($SYSTEM.SYS01.TSYSDP2)
+ ADD PROCESSH $SYSTEM.SYS01.TSYSDP2 (2, *) ($SYSTEM.SYS01.INITDLL)
+ ADD PROCESSH $SYSTEM.SYS01.TSYSDP2 (2, *) ($SYSTEM.SYS01.MCPDLL)
+ ADD PROCESSH ALLTIME 1 ($SYSTEM.SYS01.INITDLL)
+ ADD PROCESSH ALLTIME 1 ($SYSTEM.SYS01.MCPDLL)
+ ADD PROCESSH ALLTIME 1 (SL $SYSTEM.SYS01.TSL)
```

SERVERNET

L シリーズの RVU では、SERVERNET エンティティは、システム相互接続のクライアント、つまりメッセージシステム、ストレージサブシステム、および通信サブシステムに関係した I/O 操作に関する情報を提供します。ユーザーはこのカウンターを使って、これらのサブシステムのすべてのシステム相互接続アクティビティを監視できます。L01、H07/J05 以降の PVU では、このエンティティの名前として

ENDPOINT が SERVERNET に置き換わっています。ただし、SERVERNET はコマンドラインで受け入れられます。

L シリーズの RVU では、NCSL および RDMA サービス層が ServerNet ミリコード層に置き換わっています。L シリーズの RVU には、NCSL カウンターを使用して、FABRIC エンティティが備えられています。FABRIC エンティティは、より低く、より詳細なレベルのインストゥルメンテーションを提供します。

SERVERNET エンティティは、システム内トラフィックとシステム間トラフィックの両方を測定します（たとえば、G、H、および J シリーズの RVU における Expand-over-ServerNet クラスタリングか、L シリーズの RVU における Expand-over-IB クラスタリングのどちらかで接続したリモートノードのすべてのプロセッサー間トラフィック）。Measure では後者を、RIPC（リモートプロセッサー間通信）と呼んでいます。

注記:

すべてのコントローラーを指定している場合（ADD CONTROLLER *）や、CPU 番号だけを指定している場合（ADD CONTROLLER 2）は、D シリーズの RVU 測定アプリケーションで、ServerNet アドレス指定可能コントローラーを測定できます。特定の SAC を測定するには、エンティティ識別子を変更する必要があります。

トピック

SERVERNET エンティティのエンティティ指定構文

SERVERNET エンティティの DDL レコード

SERVERNET エンティティの DDL フィールドの定義

SERVERNET エンティティの使用上の注意

SERVERNET 測定の例

SERVERNET エンティティのエンティティ指定構文

SERVERNET エンティティを記述するには、次の構文を使用します。

```
{SERVERNET | ENDPOINT } entity-spec [, entity-spec ] ...
```

SERVERNET

SAC を伴う I/O 操作の測定を収集します。

entity spec

次のいずれかのように指定します。

- IPC ノードクラスの構文：

```
{ cpu [ ( IPC [, group [, module [, slot [, other-cpu ] ] ] ] ) ]
      [ ( type ) ] }
```

- RIPC ノードクラスの構文：

```
{ cpu [ ( RIPC [, group [, remote-node [, slot
      [, remote-cpu ] ] ] ] ) ] [ ( type ) ] }
```

- L シリーズ RVU の CLIM、CLMI、CLMO、CLMS ノードクラスの構文：

```
{ cpu [ ( node-class ) ] [ ( type ) ] }
{ cpu [, clim-name ] [ ( type ) ]}
```

- H および J シリーズシステムの CLIM、CLMI、CLMO、CLMS ノードクラスの構文：

```
{ cpu [ ( node-class [, group [, module [, slot
      [, port [, fiber ] ] ] ] ] ) ] }
{ cpu [, clim-name ] [ ( type ) ] }
```

- G および H シリーズシステムからのデータファイルの他のすべてのノードクラスの構文：

```
{ cpu [ ( node-class [, group [, module [, slot
      [, subdevice] ] ] ] ) ] [ ( type ) ] }
```

- D シリーズ RVU からのデータファイルの構文：

```
{ * [ ( ctrl-type ) ] }
```

指定する値は次のとおりです。

*****

すべての CPU のすべてのデバイスを測定します。

cpu

CPU 番号です。0～15 の範囲の整数にする必要があります。デフォルトは、すべての CPU です。

node-class

測定対象の SAC のクラスです。CLIM、CLMI、CLMO、CLMS、COLO、ENET、IPC、MONT、NIOC、RIPC、SCSI、および SWAN。

リモートプロセッサー間通信（RIPC）は、リモートノード上の CPU との通信を記述します。

すべてのノードクラスを指定するには、アスタリスク（*）を使用します。*node-class* を指定しないと、すべてのノードクラスが測定されます。

L シリーズ RVU では、IPC、RIPC、および CLIM（CLMI、CLMS、および CLMO を含む）ノードクラスのみがサポートされます。

group

測定対象の SAC のグループ番号です。（グループは物理的なエンクロージャーに相当します）。すべてのグループを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのグループです。IPC または RIPC ノードクラスの場合は無視されます。L シリーズ RVU で作成される測定には使用されません。

module

測定対象の SAC のモジュール番号です。すべてのモジュールを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのモジュールです。IPC または RIPC ノードクラスの場合は無視されます。L シリーズ RVU で作成される測定には使用されません。

remote-node

ノード番号です。リモートプロセッサー間通信（RIPC）に対してのみ有効です。すべてのノードを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのノードです。

slot

測定対象の SAC のスロット番号です。すべてのスロットを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのスロットです。IPC または RIPC ノードクラスの場合は無視されます。L シリーズ RVU で作成される測定には使用されません。

port

CLIM が接続されている ServerNet PIC 内（または NonStop BladeSystem 上の ServerNet ブレードスイッチ内）の特定の ServerNet コネクタを識別します。NonStop BladeSystem の場合のみ、有効なポートの値は 3～8、または"*"です。他のすべてのシステムでは、ポートの有効な値は 1～4、または"*"です。デフォルトは、すべてのポートです。L シリーズ RVU で作成される測定には使用されません。

fiber

CLIM が接続されている ServerNet ケーブル内の特定のコネクタを識別します。有効なファイバーの値は、1～4 です。ファイバーは NonStop BladeSystem にのみ関連します。デフォルトは、すべてのファイバー接続です。L シリーズ RVU で作成される測定には使用されません。

other-cpu

エンティティと通信中のローカル CPU の番号です。IPC に対してのみ有効です。デフォルトは、すべての CPU です。

remote-cpu

エンティティと通信中のリモートノード上の CPU の番号です。RIPC に対してのみ有効です。デフォルトは、すべての CPU です。

type

ServerNet アドレス指定可能コントローラーのクラスの部品番号、または CLMS のタイプです。

注記:

type 指定は、ストレージ CLIM（CLMS）の場合を除き、CLIM ノードクラスには使用されません。

- ストレージサブシステムから報告される CLMS アクティビティの場合、タイプは 2 です。
- CIP サブシステムから報告される CLMS アクティビティの場合、タイプは 3 です。

clim-name

エンティティと通信中の CLIM の名前です。CLIM ノードクラスに対してのみ有効です。デフォルトは、すべての CLIM です。

SERVERNET エンティティの DDL レコード（ZMS 形式）

SERVERNET エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmssvnet-id.
 02 node-class                  type binary 32 unsigned.
 02 node-class-s                type character 4
                                redefines node-class.
 02 ctrl-type                   type binary 16 unsigned.
 02 flags                       type binary 16 unsigned.
 02 GMS.
    03 group                    type binary 32 unsigned.
    03 module                   type binary 32 unsigned.
    03 remote-cluster           type binary 32 unsigned
                                redefines module.
    03 slot                     type binary 32 unsigned.
 02 subdevice                   type binary 32 unsigned.
```

```
 02 remote-cpu                  type binary 32 unsigned
                                redefines subdevice.
 02 PF                          redefines subdevice.
    03 port                     type binary 16 unsigned.
    03 fiber                    type binary 16 unsigned.
 02 adapter-name                type character 64.
 02 SAC-name                    type character 64.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmssvnet-ctrs.
 02 requests                    type binary 64.
 02 total-io-bytes              type binary 64.
 02 io-qtime                    type binary 64.
 02 io-qbusy-time               type binary 64.
 02 read-requests               type binary 64.
 02 write-requests              type binary 64.
 02 read-bytes                  type binary 64.
 02 read-cbytes                 type binary 64.
 02 read-qtime                  type binary 64.
 02 read-qbusy-time             type binary 64.
 02 write-bytes                 type binary 64.
 02 write-cbytes                type binary 64.
 02 write-qtime                 type binary 64.
 02 write-qbusy-time            type binary 64.
 02 server-qtime                type binary 64.
 02 retries                     type binary 64.
 02 acks                        type binary 64.
 02 x-defrd-busy-time           type binary 64.
 02 y-defrd-busy-time           type binary 64.
 02 read-crequests              type binary 64.
 02 write-crequests             type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmssvnet.  FILE is "zmssvnet" ENTRY-SEQUENCED.
 02 hdr                         type zmsheader.
 02 ctr                         type zmssvnet-ctrs.
 02 id                          type zmssvnet-id.
end
```

SERVERNET エンティティの DDL レコード（Legacy 形式）

SERVERNET エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD servernet.  FILE is "svnet" ENTRY-SEQUENCED.
                        .
                        .
                        .
```

(error、time items、および measurement identification items については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください)

```
                        .
                        .
```

```
                        .
* entity identification items:
 02 channel                        type binary 16 unsigned.
 02 ctrl                           type binary 16 unsigned.
 02 ctrl-type                      type binary 16 unsigned.
* counter value items:
 02 requests                       type binary 32 unsigned.
 02 total-io-bytes                 type binary 64.
 02 io-qtime                       type binary 64.
 02 io-qlen-max                    type binary 16 unsigned.
* F40 new entity identification items:
 02 adapter-name                   type character 64.
 02 SAC-name                       type character 64.
 02 node-class                     type binary 32 unsigned.
 02 node-class-s                   type character 4
                                   redefines node-class.
 02 GMS.
    03 group                       type binary 32 unsigned.
    03 module                      type binary 32 unsigned.
    03 remote-cluster              type binary 32 unsigned
                                   redefines module.
    03 slot                        type binary 32 unsigned.
 02 subdevice                      type binary 32 unsigned.
 02 remote-CPU                     type binary 32 unsigned
                                   redefines subdevice.
* F40 new counter value items:
 02 io-qbusy-time                  type binary 64.
* G01 counter value items:
 02 read-requests                  type binary 32 unsigned.
 02 read-bytes                     type binary 64.
 02 read-qtime                     type binary 64.
 02 read-qlen-max                  type binary 16 unsigned.
 02 read-qbusy-time                type binary 64.
 02 write-requests                 type binary 32 unsigned.
 02 write-bytes                    type binary 64.
 02 write-qtime                    type binary 64.
 02 write-qlen-max                 type binary 16 unsigned.
 02 write-qbusy-time               type binary 64.

* G08 new counters:
 02 read-cbytes                    type binary 64.
 02 write-cbytes                   type binary 64.
 02 server-qtime                   type binary 64.
 02 server-qlen-max                type binary 16 unsigned.
 02 retries                        type binary 32.
 02 acks                           type binary 32.
 02 x-defrd-busy-time              type binary 64.
 02 y-defrd-busy-time              type binary 64.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

SERVERNET エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧にしています。サブフィールドは、フィールドに指定する順に一覧にしています。

ADAPTER-NAME

SAC が存在するアダプターに関連付けられた論理名です。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

CTRL-TYPE

3128 などの製品番号。

IPC および RIPC レコードでの CTRL-TYPE は、リンカーの場合は 0、リスナーの場合は 1 です。IPC レコードについて詳しくは、**SERVERNET エンティティの使用上の注意**(410 ページ)を参照してください。

FLAGS

（ZMS 形式のみ、L シリーズ以降の RVU）測定が L シリーズおよびこれ以降の RVU で行われたかどうかを示すために使用される 16 ビットフィールド。

flags. 0:13	現在は使用されていません。
flags.14	1 = 測定は L シリーズ以降の RVU で行われました
flags.15	現在は使用されていません。

GMS

物理的位置のアドレス（グループ、モジュールまたは ServerNet ノード、およびスロット番号）。GMS フィールドは以下のサブフィールドに分かれています。

GROUP	グループの数です。
MODULE または REMOTE-CLUSTER（MODULE の再定義）	エンティティが通信しているシステムのモジュール番号（MODULE）または ServerNet ノード番号（REMOTE-CLUSTER）。REMOTE-CLUSTER は RIPC についてのみ有効です。
SLOT	スロット番号です。

IPC レコードの場合、GROUP、MODULE、および SLOT の値はすべて 0 です。

RIPC レコードの場合、GROUP および SLOT の値は両方とも 0 です。

GROUP、MODULE、REMOTE-CLUSTER、および SLOT キーワードは、LIST コマンドおよび LISTALL コマンドの IF 句と BY 句で使用できます。

L シリーズの RVU では、CLIM は CPU と名前または名前でのみ指定でき、GMS では指定できません。

NODE-CLASS

システム構成データベースで定義されている SAC タイプ。

NODE-CLASS-S

NODE-CLASS を 4 文字の英数字フィールドとして再定義します。指定可能な値（空白を埋める 4 文字）は、次のとおりです。

- “CLIM”
- “CLMI”
- “CLMO”
- “CLMS”
- "COLO"
- "ENET"
- "IPC "
- "MONT"
- "NIOC"
- "RIPC"
- "SCSI"
- "SWAN"
- "* "

L シリーズの RVU でサポートされるノードクラスは、CLIM、CLMI、CLMO、CLMS、IPC、および RIPC です。

注記:

CLIM は、すべての CLIM ノードクラスのワイルドカードです。

PF

（ZMS 形式のみ）ポートフィールドとファイバーフィールドを保持する構造体。

PORT	ポートは、CLIM が接続されている ServerNet PIC 内（または NonStop BladeSystem 上の ServerNet ブレードスイッチ内）の特定の ServerNet コネクタを識別します。
FIBER	（NonStop BladeSystems のみ）ファイバーは、CLIM の接続先である ServerNet ケーブル内の特定のコネクタを識別します。

REMOTE-CPU

エンティティと通信中のリモートクラスター上の CPU の番号。RIPC に対してのみ有効です。このフィールドは、DDL レコードの `subdevice` を再定義します。

SAC-NAME

物理ディスクへのこのパスで使用される SAC の論理名。論理名はシステム構成データベースによって保持されます。この名前は null で埋められ null で終了します。

IPC レコードの場合、Measure は SAC 名の代わりにこの識別子を生成します。

`IPC-CPU`*`xx`*`-TO-CPU`*`yy`*

指定する値は次の通りです。*xx* はリンカー CPU 番号で、*yy* はリスナー CPU 番号です。

SUBDEVICE

マルチファンクション I/O ボード（MFIOB）上の特定の SCSI デバイスのサブデバイス識別子。

識別子	デバイスタイプ
0	ServerNet バスインターフェイス（SBI）
1	内部 SCSI コントローラー番号 1
2	内部 SCSI コントローラー番号 2
3	COMM と SP
4	SP 間通信
5	外部 SCSI ポートコントローラー

IPC レコードの場合、SUBDEVICE は宛先（リスナー）プロセッサーに適用されます。詳しくは、**SERVERNET エンティティの使用上の注意**(410 ページ)を参照してください。L シリーズ RVU で作成される測定には使用されません。

SERVERNET エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧しています。

ACKS

明示的な肯定応答メッセージの送信回数。このカウンターは、IPC および RIPC レコードにのみ適用されます。

カウンタータイプ：増分。

IO-QBUSY-TIME

任意の数やタイプの要求がこのエンティティのキューに入れられた状態で消費された時間。

カウンタータイプ：キュービジー。

IO-QTIME

SAC が I/O を実行中のビジー時間です。

REPORT RATE ON の場合、これはこのコントローラーのキューに入れられる未処理 I/O の平均数です。I/O の平均数が 1 より小さい場合、この数値はコントローラーのビジーパーセンテージとして解釈できます。

カウンタータイプ：キュー。

READ-BYTES

読み取り要求またはプル操作の結果として転送されたバイトと、リンカー CPU からプッシュ前操作で受信したバイトの合計数。

カウンタータイプ：累積。

READ-CBYTES

IPC および RIPC ノードクラスでは、要求の結果としてプロセッサーに転送されたメッセージシステムプロトコル制御バイトの合計数。CLIM ノードクラスでは、CLIM から CPU への受信制御要求での合計バイト数。どの場合でも、これらの制御バイトはカウンター READ-BYTES には加算されません。

カウンタータイプ：累積。

READ-CREQUESTS

（ZMS 形式のみ）CLIM では、このエンティティから CPU への受信制御要求の数。このカウンターは、NORMAL 表示には示されますが、BRIEF 表示には示されません。

カウンタータイプ：増分。

READ-QBUSY-TIME

このエンティティからメモリへのデータ転送の要求がキューに入れられた状態で費やした時間。

カウンタータイプ：キュービジー。

READ-QTIME

このエンティティに対する､キューに入れられた読み取り要求の合計時間。REPORT RATE を ON にした場合、このエンティティに対する、キューに入れられた読み取り要求の平均数。

カウンタータイプ：キュー。

READ-REQUESTS

このエンティティから CPU へデータを転送した要求の数。

カウンタータイプ：増分。

REQUESTS

このコントローラーパスに接続したすべての IOP による SAC への要求の合計数。

カウンタータイプ：累積。

RETRIES

メッセージシステムのメッセージの送信が成功するまでの試行回数。このカウンターは、IPC および RIPC レコードにのみ適用されます。

カウンタータイプ：増分。

SERVER-QTIME

このプロセッサーでのメッセージの受信から返信の発行までの時間（ミリ秒単位）。この値には、CPU にデータをプルするメッセージシステム読み取り操作の READ-QTIME は含まれません。このカウンターは、受信プロセスが要求に応答するまでに必要な時間から、メッセージシステムのレイテンシ時間を切り離すときに役立ちます。これは、IPC および RIPC レコードにのみ適用されます。

カウンタータイプ：累積。

TOTAL-IO-BYTES

このコントローラーパスに接続したすべての IOP によって SAC へ転送された、または SAC から転送された合計バイト数。

カウンタータイプ：累積。

WRITE-BYTES

書き込み要求の結果として転送された合計バイト数。

カウンタータイプ：累積。

WRITE-CBYTES

IPC および RIPC ノードクラスでは、要求の結果としてプロセッサーに転送されたメッセージシステムプロトコル制御バイトの合計数。CLIM ノードクラスでは、CPU から CLIM への送信制御要求での合計バイト数。どの場合でも、これらの制御バイトはカウンター WRITE-BYTES には加算されません。

カウンタータイプ：累積。

WRITE-CREQUESTS

（ZMS 形式のみ）CLIM では、CPU からこのエンティティへの送信制御要求の数。このカウンターは、NORMAL 表示には示されますが、BRIEF 表示には示されません。

カウンタータイプ：増分。

WRITE-QBUSY-TIME

メモリからこのエンティティへのデータ転送の要求がキューに入れられた状態で費やした時間。

カウンタータイプ：キュービジー。

WRITE-QTIME

このエンティティに対して、キューに入れられた書き込み要求の合計時間。REPORT RATE を ON にして、このエンティティに対して、キューに入れられた書き込み要求の平均数。IPC および RIPC ノードクラスエンティティの場合、このカウンターは、要求が完了するまでリンカーが待機する時間全体を含めるように変更されました（Measure G08 以降の PVU）。

カウンタータイプ：キュー。

WRITE-REQUESTS

CPU からこのエンティティへデータを転送した要求の数。

カウンタータイプ：増分。

X-DEFRD-BUSY-TIME

メッセージシステムが、Y ファブリックの使用から X ファブリックの使用に切り替えられなかった時間（マイクロ秒単位）。このカウンターの値が大きい場合、Y ファブリックに未処理のメッセージが生じていたり、X ファブリックに問題が起きていたりする可能性があります。このカウンターは、IPC および RIPC レコードにのみ適用されます。

カウンタータイプ：ビジー。

Y-DEFRD-BUSY-TIME

メッセージシステムが、X ファブリックの使用から Y ファブリックの使用に切り替えられなかった時間（マイクロ秒単位）。このカウンターに大きな値を指定すると、X ファブリックに未処理のメッセージが生じたり、Y ファブリックに問題が起きたりすることがあります。このカウンターは、IPC および RIPC レコードにのみ適用されます。

カウンタータイプ：ビジー。

SERVERNET エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

CHANNEL

（Legacy 形式のみ）使用されません。0 を返します。

CTRL

（Legacy 形式のみ）使用されません。0 を返します。

SERVERNET エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

IO-QLEN-MAX

（Legacy 形式のみ）IO-QTIME カウンターで記述されるキューの未処理 I/O の最大数。

カウンタータイプ：最大キュー。

READ-QLEN-MAX

（Legacy 形式のみ）カウンターレコードが割り当てられてから読み取り要求キューに入れられた要求の最大数。

カウンタータイプ：最大キュー。

SERVER-QLEN-MAX

（Legacy 形式のみ）SERVER-QTIME カウンターで記述されるキューの項目の最大数。このカウンターは、IPC および RIPC レコードにのみ適用されます。

H シリーズ以降の RVU では、このカウンターの値は 1 です。

カウンタータイプ：最大キュー。

WRITE-QLEN-MAX

（Legacy 形式のみ）カウンターが割り当てられてから読み取り要求キューに入れられた要求の最大数。

カウンタータイプ：最大キュー。

SERVERNET エンティティの使用上の注意

- SERVERNET エンティティは、システム全体での SERVERNET のすべての操作を表示します。たとえば、複数のデバイスが CLIM を共有している場合、SERVERNET エンティティは、CLIM を使用するすべてのデバイスの複合的なアクティビティを測定します。
- ZMSSVNET レコードの SUBSYSTEM-VERSION は、レコードの node^class で表される特定のサブシステムによって提供されます。
  - CLIM ノードクラスの場合、CIP（クラスター I/O プロトコル）サブシステム
  - IPC および RIPC の場合、メッセージシステム
  - SCSI および COLO の場合、SCSI モジュールドライバー
  - NIOC の場合、SLSA モジュールドライバー
- 単一のデバイスが 1 つのコントローラーに接続されている場合、SERVERNET エンティティとデバイスエンティティタイプ（DEVICE、DISK など）との間に重複が存在しています。
- MFIOB デバイスなどのストレージ SAC の場合、IO-QBUSY-TIME カウンターが各要求のビジー時間とサービス時間を最も正確に示すインジケーターです。READ-QBUSY-TIME と

WRITE-QBUSY-TIME のカウンターは同様のデータを提供しますが、これら 2 つのカウンターは、要求のキューイングのため、ServerNet 環境では多少正確さが低下します。

SERVERNET IPC および RIPC の使用上の注意

- プロセッサー間通信（IPC）アクティビティは、一部が開始側プロセッサー（リンカーと呼ばれます）で測定され、一部が受信側または応答側プロセッサー（リスナーと呼ばれます）で測定されます。各プロセッサーでは、1 つの SERVERNET レコードが別のプロセッサーとのリンカーアクティビティを追跡し、2 番目の SERVERNET レコードがリスナーアクティビティを追跡します。Measure パフォーマンスモニターは、IPC SERVERNET レコードの SAC Name フィールドに次の識別子を生成します。

  ```
  IPC-CPUxx-TO-CPUyy
  ```

  ここで *xx* はリンカー CPU 番号で、*yy* はリスナー CPU 番号です。
- クラスター接続をサポートするために、リンカーおよびリスナーのレコードのデータは、リンカーレコード（開始側プロセッサー）がメッセージトラフィックを完全に明らかにできるように変更しました。このサポートを実現するには、リンカーとリスナーの両方のレコードが、同じメッセージ交換数と、対応するバイト数をカウントするようにします。このように変更すると、クラスター接続のリスナーレコードが別のノードに置かれ、1 つのノードのみからの Measure データファイルでは簡単には使用できなくなるので、リンカーレコードからの報告がより適切になります。
- リスナー READ-BYTES は、対応するリンカーレコード WRITE-BYTES を超える場合があります。この状態は、メッセージシステムの MQC 割り当てが頻繁に失敗し、場合に起こり、これによりメッセージデータの再転送が発生します。
- 事前プッシュ（非クラスター）メッセージの場合、READ-REQUESTS の数はリスナーレコードではインクリメントしません。リスナーレコードの READ-REQUESTS の数は、実行されたメッセージシステムプル操作の数を反映し、WRITE-REQUESTS カウンターのサブセットになります。READS と READ-BUSY-QTIME の比較により、プル操作の正確な Measure システムレイテンシが得られます。
- リモートノードメッセージ（これはメッセージを事前プッシュしません）では、リモートプロセッサーまたはリスナーが WRITE-REQUEST に応じようとし、メッセージが大きい場合に、READ-REQUESTS の数が REQUESTS の数の 2 倍になることがあります。readlinkcache サイズよりも大きなメッセージの場合（たとえば 2048 バイトより大きいメッセージ）、リスナープロセスは 2 つのプル（制御用のプルとデータ用のプル）を実行します。
- メッセージシステムリスナーレコードの SERVER-QTIME カウンターは、リスナープロセッサーによるメッセージの受信からその後の応答までの時間を測定します。関連付けられているリンカープロセッサーレコードの WRITE-QTIME カウンターと合わせて使用した場合、このカウンターから、メッセージ転送の Measure システムレイテンシが得られます。平均メッセージレイテンシは、リスナーレコード SERVER-QTIME をリンカーレコード WRITE-QTIME から引いた後、リンカーレコード WRITE-REQUESTS で割って計算できます。

  平均メッセージレイテンシの計算：

  ```
  WRITE-QTIME(linker) - SERVER-QTIME(listener)/WRITE-REQEUSTS(linker)
  ```
- Measure は、システム間で測定間隔を調整しません。別々のクラスターにあるプロセッサー間の平均メッセージレイテンシを計算するには、同様の間隔で SERVER-QTIME および WRITE-QTIME の平均値を計算してから、結果の値を減算します。
- NonStop S70000 サーバー（NSR-G プロセッサー）への接続では、RETRIES カウンターは高い値になると予想されます。これらのプロセッサーの ServerNet インターフェイス（MITE）の不具合は、ソフトウェアのリカバリによって処理されます。これらの問題は NonStop S72000 サーバー（NSR-T プロセッサー）では修正されていますが、NonStop S70000 サーバー（NSR-G プロセッサー）と通信するときには、S72000 サーバーは依然として高い RETRIES の数を報告することがあります。
- 場合によっては、リモートプロセッサーまたはリスナーが WRITE-REQUEST に応じようとし、メッセージが非常に大きい場合に、READ-REQUESTS の数が REQUESTS の数の 2 倍になることがあります。readlinkcache サイズよりも大きなメッセージの場合（たとえば 2048 バイトより大きいメッセージ）、リスナープロセスは 2 つのプル（制御用のプルとデータ用のプル）を実行します。

CLIM の使用上の注意

- Measure H03/J01 以降の PVU では、Measure は、IP CLIM に対し、CLIM *node-class* 指定 CLMI をサポートし、ストレージ CLIM に対して CLMS をサポートします。これらのノードクラスの場合、*node-class* 後の 4 番目のオプション指定が *port* です。CLIM 以外のノードクラスの場合、これは *subdevice* です。

  Measure H04/J02 以降の PVU では、Measure は、Telco CLIM に対し CLIM *node-class* 指定 CLMO をサポートします。

  CLIM は、すべての CLIM ノードクラスのワイルドカードです。

  NonStop BladeSystem および L シリーズの RVU を除き、各 CLIM は、接続先の P-Switch の物理的な場所と、接続しているポートの面から CLIM を識別するグループモジュールスロットポートで一意に識別されます。グループ（システムエンクロージャー）とモジュール（グループのサブセット）が P-Switch を識別します。スロット（モジュール内のラベル付きの物理的な領域）は、その P-Switch 内の特定の ServerNet PIC とポートを識別します。ポートは、CLIM が接続されている ServerNet PIC 内で特定の ServerNet コネクタを識別します。NonStop BladeSystem では、CLIM を識別するために追加のファイバー番号が必要です。

  L シリーズの RVU で、CLIM は CLIM 名でのみ識別されます。
- 各ストレージ CLIM のアクティビティは、2 つの SERVERNET レコードで報告されます。一方のレコードは、ストレージサブシステムからのアクティビティを報告し、もう一方のレコードは、CIP サブシステムからのアクティビティを報告します。

  ストレージサブシステムレコードは、CLIM でサポートされているすべてのデバイスの I/O アクティビティ（READ-BYTES、WRITE-BYTES、READ-REQUESTS、WRITE-REQUESTS、およびさまざまなキューカウンター）を報告します。制御 IO アクティビティ（READ-CBYTES、WRITE-CBYTES、READ-CREQUESTS、および WRITE-CREQUESTS カウンター）は一切報告しません。

  CIP サブシステムは、CLIM で実行される制御 IO アクティビティを報告し、どの IO アクティビティも報告しません。通常、制御 IO アクティビティは、CIP サブシステムと CLIM の間の低周波数 ping に過ぎません。

  次に、ストレージ CLIM の 2 つのレポートレコードの例を示します。最初の例は CIP サブシステムのもので、2 番目の例はストレージサブシステムのものです。

  CIP サブシステムの例：

```
SAC   C1002531
Adpt  C1002531
CPU   6    Node Class  CLMS   GMS  100,2,5     Port  3   Fiber  1
Format Version:  H04  Data Version:  H04  Subsystem Version:  1
Local System \BLNSK2  From  26 Mar 2009, 17:13:10   For  33.6 Seconds
--------------------------------------------------------------------------
-
Requests                         12 #    Total-IO-Bytes                672 #
IO-Qtime                                 IO-Qbusy-Time
--------- Data into this CPU --------  --------- Data out from this CPU
----
Read-Requests                            Write-Requests
Read-Bytes                               Write-Bytes
Read-Cbytes                     336 #    Write-Cbytes                  336 #
Read-Qtime                               Write-Qtime
Read-Qbusy-Time                          Write-Qbusy-Time
Read-Crequests                    6 #    Write-Crequests                 6 #
--------------------------------------------------------------------------
-
Server-Qtime
Retries                                  Acks
```

```
X-Defrd-Busy-Time                       Y-Defrd-Busy-Time
```

ストレージサブシステムの例：

```
SAC   C1002531
Adpt  C1002531
CPU   6    Node Class  CLMS   GMS  100,2,5     Port  3   Fiber  1
Format Version:  H04  Data Version:  H04  Subsystem Version:  1
Local System \BLNSK2  From  26 Mar 2009, 17:13:10   For  33.6 Seconds
------------------------------------------------------------------------
-
Requests                    2,520 #    Total-IO-Bytes         82,281,472 #
IO-Qtime                     1.18 sec  IO-Qbusy-Time                1.17
sec
--------- Data into this CPU --------  --------- Data out from this CPU
----
Read-Requests               2,514 #    Write-Requests                 6 #
Read-Bytes             82,231,808 #    Write-Bytes                49,664 #
Read-Cbytes                            Write-Cbytes
Read-Qtime                   1.14 sec  Write-Qtime                 36.71 ms
Read-Qbusy-Time              1.14 sec  Write-Qbusy-Time            36.71 ms
Read-Crequests                         Write-Crequests
------------------------------------------------------------------------
-
Server-Qtime
Retries                                Acks
X-Defrd-Busy-Time                      Y-Defrd-Busy-Time
```

SERVERNET クラスターの測定の設定の例

ADD、DELETE、LIST、LISTACTIVE、および LISTALL コマンドは、1 つまたは複数の ENDPOINT エンティティを参照できます。この例では ADD コマンドが使用されていますが、DELETE、LIST、LISTACTIVE、および LISTALL コマンドはすべて、リモートクラスターアクティビティについての測定を設定または表示するために同じコマンド構文を共有します。

コマンド	以下の範囲のすべてのリモート IPC アクティビティを測定...
ADD SERVERNET * (RIPC)	ローカルシステム上のすべての CPU とすべてのリモートクラスター上のすべての CPU
ADD SERVERNET * (RIPC, *, 4)	ローカルシステム上のすべての CPU とリモートノード番号 4 上のすべての CPU
ADD SERVERNET * (RIPC, *, 4, *, 5)	ローカルシステム上のすべての CPU とリモートノード番号 4 上の CPU 5
ADD SERVERNET * (RIPC,*,4)	ローカルシステム上の CPU 1 とリモートノード番号 4 上のすべての CPU
ADD SERVERNET 1 (RIPC, *,4 , *, 0)	ローカルシステム上の CPU 1 とリモートノード番号 4 上の CPU 0

CLIM の測定の設定の例

ADD、DELETE、LIST、LISTACTIVE、および LISTALL コマンドは、1 つまたは複数の ENDPOINT エンティティを参照できます。この例では ADD コマンドが使用されていますが、DELETE、LIST、

LISTACTIVE、および LISTALL コマンドはすべて、CLIM の測定を設定または表示するために同じコマンド構文を共有します。

コマンド	測定対象...
ADD SERVERNET *(CLIM)	すべての CPU 上のすべての CLIM
ADD SERVERNET 1 (CLIM)	CPU 1 のすべての CLIM
ADD SERVERNET * (CLMS, 1,2,3)	すべての CPU で GMS が 1,2,3 に等しいすべてのストレージ CLIM 上のすべてのポート（H または J シリーズの RVU）
ADD SERVERNET * (CLMI, 1,2,3,1)	すべての CPU で GMS が 1,2,3 に等しいすべての IP CLIM 上のポート 1（H または J シリーズの RVU）
ADD SERVERNET * (CLMI, 1,2,3,8,1)	すべての CPU で GMS が 1,2,3 に等しいすべての IP CLIM 上のポート 8、ファイバー 1（H または J シリーズの RVU）
ADD SERVERNET * (CLMO)	すべての CPU 上のすべての Telco CLIM

SQLPROC

SQLPROC エンティティは、1 つまたは複数の SQL プロセスに関する情報を提供します。SQL プロセスごとに 1 つの SQLPROC カウンターレコードがあります。

Measure G09 以降の PVU では、SQLPROC エンティティで OSS ファイルのパス名を使用できます。

トピック
SQLPROC エンティティのエンティティ指定構文
SQLPROC エンティティの DDL レコード
SQLPROC エンティティの DDL フィールドの定義
新しい形式の SQLPROC エンティティの使用上の注意

SQLPROC エンティティのエンティティ指定構文

SQLPROC エンティティを記述するには、次の構文を使用します。

```
SQLPROC entity-spec [,entity-spec] ...
```

SQLPROC

ローカルシステム上の 1 つまたは複数の SQL プロセスに関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                            }
{ $process-name [ ( pid ) ] }
{ disk-filename [ ( pid ) ] }
{ "pname" [ ( pid ) ]        }
{ pid                          }
```

指定する値は次のとおりです。

*****

ローカルシステム上のすべての SQL プロセスを測定します。

$*process-name*

測定対象の SQL プロセスの名前です。

pid

プロセス ID です。プロセス ID は、測定対象の SQL プロセスの所有者を識別します。*pid* は次のように指定します。

```
{ cpu,pin }
```

cpu

SQL プロセスが実行されている CPU の番号です。

pin

SQL プロセスのプロセス識別番号です。すべての PIN を指定するには、アスタリスク（*）を使用します。

disk-filename

測定対象の SQL/MP コードが含まれているオブジェクトファイルの名前です。このファイルは、実行中のプロセスのオブジェクトファイルである必要があります。*disk-filename* は次のいずれかのように指定します。

```
{ [ \system.]$device.subvol.filename[:crvsn]          }
{ subvol.filename[:crvsn]                             }
{ filename[:crvsn]                                    }
```

system

システム名です。すべてのデバイスを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのシステムです。

$*device*

ボリューム（デバイス）の名前です。すべてのボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのボリュームです。

subvol

サブボリューム名です。すべてのサブボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのサブボリュームです。

filename

ファイル名です。すべてのファイル（一時ファイルを除く）を指定するには、アスタリスク（*）を使用します。

:*crvsn*

（Measure G09 以降の PVU）一意のファイル名を形成するために必要な、タイムスタンプ、作成バージョンのシリアル番号（CRVSN）、またはファイル名拡張子です。このオプションを使用して、ファイル名の一意性を保証します。CRVSN は Measure レポートおよび LISTGNAME コマンドから使用できます。

"*pname*"

（Measure G09 以降の PVU）完全修飾または部分修飾のいずれかの OSS ファイルパス名です。スラッシュ（/）で始まらない OSS ファイルパス名は、部分的なパス名とみなされ、OSSPATH の現

在の設定を前に指定することで拡張されます。/G で始まる OSS ファイルパス名はサポートされていないため、代わりに Guardian ファイル名を使用してください。

注記:

OSS ファイルパス名は大文字と小文字が区別され、二重引用符（""）で囲んで指定する必要があります。有効な OSS ファイルパス名は、特定のファイル、または特定のディレクトリ内のファイルのセットを参照できます。ディレクトリが指定されている場合は、そのディレクトリ内のファイルのみが含まれます。指定したディレクトリの下位のディレクトリ内のファイルは含まれません。

SQLPROC エンティティの DDL レコード（ZMS 形式）

SQLPROC エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmssqlp-id.
 02 pin                          type binary 16 unsigned.
 02 priority                     type binary 16 unsigned.
 02 userid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 creatorid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 process-name                 type character 8.
 02 program-file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 osspid                       type binary 32 unsigned.
 02 ancestor-cpu                 type binary 16 unsigned.
 02 ancestor-pin                 type binary 16 unsigned.
 02 ancestor-sysname             type character 8.
 02 ancestor-process-name        type character 8.
 02 device-name                  type character 8.
 02 program-file-name-MID.
    03 PATHID                    type character 24.
    03 CRVSN                     type character 6.
 02 reserved-1                   type character 2.
 02 GMOM.
    03 GMOM-node                 type binary 16 unsigned.
    03 GMOM-cpu                  type binary 16 unsigned.
    03 GMOM-pin                  type binary 16 unsigned.
    03 GMOM-jobid                type binary 16 unsigned.
 02 GMOM-full-id                 type binary 64
                                 redefines GMOM.
 02 GMOM-sysname                 type character 8.
 02 GMOM-process-name            type character 8.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmssqlp-ctrs.
 02 sql-obj-recompiles           type binary 64.
 02 sql-obj-recompile-time       type binary 64.
 02 sql-stmt-recompiles          type binary 64.
 02 sql-stmt-recompile-time      type binary 64.
 02 sql-newprocesses             type binary 64.
 02 sql-newprocess-time          type binary 64.
 02 opens                        type binary 64.
 02 open-time                    type binary 64.

 02 reserved-1                   type character 16.

end
```

DDL レコード記述フィールド

```
RECORD zmssqlp. FILE is "zmssqlp" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 ctr                          type zmssqlp-ctrs.
 02 id                           type zmssqlp-id.
end
```

SQLPROC エンティティの DDL レコード（Legacy 形式）

SQLPROC エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD sqlproc. FILE is "sqlproc" ENTRY-SEQUENCED.
                      .
                      .
                      .
```

error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)

```
                      .
                      .
                      .
* entity identification items:
 02 pin                          type binary 16 unsigned.
 02 process-name                 type character 8.
 02 program-file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 priority                     type binary 16 unsigned.
 02 userid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
* counter value items:
 02 sql-obj-recompiles           type binary 16 unsigned.
 02 sql-obj-recompile-time       type binary 64.
 02 sql-stmt-recompiles          type binary 16 unsigned.
 02 sql-stmt-recompile-time      type binary 64.
 02 sql-newprocesses             type binary 16 unsigned.
 02 sql-newprocess-time          type binary 64.
 02 opens                        type binary 16 unsigned.
 02 open-time                    type binary 64.
```

```
 02 creatorid.
    03 group                      type binary 8 unsigned.
    03 user                       type binary 8 unsigned.
* new entity identification items for D10:
 02 ancestor-cpu                 type binary 16 unsigned.
 02 ancestor-pin                 type binary 16 unsigned.
 02 ancestor-sysname             type character 8.
 02 ancestor-process-name        type character 8.
* SMS changes:
 02 device-name                   type character 8.
* new identifiers for OSS file pathname support:
 02 osspid                       type binary 32 unsigned.
 02 program-file-name-mid.
    03 pathid                    type character 24.
    03 crvsn                     type character  6.
* New identifiers for NetBatch Job Control:
 02 GMOM.
    03 GMOM-node                 type binary 16 unsigned.
    03 GMOM-cpu                  type binary 16 unsigned.
    03 GMOM-pin                  type binary 16 unsigned.
    03 GMOM-jobid                type binary 16 unsigned.
 02 GMOM-full-id                 type binary 64
                                 redefines GMOM.
 02 GMOM-sysname                 type character 8.
 02 GMOM-process-name            type character 8.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

SQLPROC エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧にしています。サブフィールドは、フィールドに指定する順に一覧にしています。

ANCESTOR-CPU

親プロセスが存在する CPU の番号。

ANCESTOR-PIN

親プロセスのプロセス識別番号。

ANCESTOR-PROCESS-NAME

親プロセスの名前。レポートヘッダーには、親の情報が表示されます。この情報を取得できない場合は、「unknown」という文字列が表示されます。

ANCESTOR-SYSNAME

親プロセスが存在するシステムの名前。

CREATORID

作成者アクセス ID（CAID）。このフィールドはプロセスの作成者を識別し、GROUP と USER の 2 つのサブフィールドに分かれています。

DEVICE-NAME

プログラムファイルが配置されているディスクデバイス。SMF ファイルの場合、このフィールドには、論理ファイル名に対応する物理的な場所が表示されます。NonStop ファイルの場合、このフィールドは、PROGRAM-FILE-NAME VOLUME サブフィールドと同じです。

GMOM

GMOM 構造は以下の 4 つのサブフィールドで構成されます。

GMOM-NODE	GMOM プロセスが NetBatch ジョブの一部であり、GMOM プロセスがリモートである場合、GMOM プロセスの Expand ノード番号。そうでない場合はゼロ。
GMOM-CPU	GMOM プロセスが NetBatch ジョブの一部である場合、GMOM プロセスのプロセッサー番号。そうでない場合はゼロ。
GMOM-PIN	GMOM プロセスが NetBatch ジョブの一部である場合、GMOM プロセスの PIN。そうでない場合はゼロ。
GMOM-JOBID	指定された GMOM によって割り当てられる、このプロセスを起動した NetBatch ジョブのジョブ ID。そうでない場合はゼロ。

GMOM-FULL-ID

個々の GMOM フィールドの単一値としての 64 ビットの再定義。ゼロの値は、プロセスが NETBATCH ジョブストリームの一部でないことを示します。

GMOM-PROCESS-NAME

このプロセスを同様に開始した、NETBATCH ジョブを開始した GMOM プロセスの名前。そうでない場合は空白。

GMOM-SYSNAME

GMOM プロセスがリモートの場合、GMOM ノードの Expand システム名。そうでない場合は空白。

OSSPID

このプロセスが OSS 環境で実行される場合、このフィールドには OSS プロセス ID が含まれます。非 OSS プロセスの場合、このフィールドにはゼロが含まれます。

PIN

測定対象の SQL プロセスのプロセス識別番号。

PRIORITY

プロセスの作成優先順位。ALTPRI TACL コマンドを使用してプロセスの優先順位を変更しても、この値には影響しません。

PROCESS-NAME

SQL プロセスの名前。

PROGRAM-FILE-NAME

プロセスが実行しているオブジェクトファイルの名前。この名前は、NonStop ファイルか SMF ファイルのいずれかに適用できます。PROGRAM-FILE-NAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、プログラムファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、プロセスが実行されたときに指定された物理ファイル名を表します。 SMF ファイルでは、プロセスが実行されたときに使用された、位置に依存しない論理ファイル名を表します。

PROGRAM-FILE-NAME-MID

OSS ファイルパス名の内部形式表現。PROGRAM-FILE-NAME-MID には次の 2 つのサブフィールドがあります。

PATHID	OSS ファイルパス名の内部形式表現です。非 OSS ファイルの場合、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。非 OSS ファイルの場合、このフィールドにはゼロが含まれます。

USERID

プロセスのプロセスアクセサー ID（PAID）。

SQLPROC エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧しています。

OPEN-TIME

このプロセスが OPEN の実行に費やした時間。

カウンタータイプ：経過時間。

OPENS

このプロセスの代わりに SQL エグゼキューターによって実行された OPEN 呼び出しの数。

カウンタータイプ：増分。

SQL-NEWPROCESS-TIME

このプロセスが SQL-NEWPROCESS の実行に費やした時間。

カウンタータイプ：経過時間。

SQL-NEWPROCESSES

このプロセスの代わりに SQL エグゼキューターによって生成された NEWPROCESS 呼び出しの数。

カウンタータイプ：増分。

SQL-OBJ-RECOMPILE-TIME

このプロセスがオブジェクトの再コンパイルに費やした時間。SQL/MX では、このカウンターはサポートされていません。

カウンタータイプ：経過時間。

SQL-OBJ-RECOMPILES

このプロセスに対して再コンパイルされたオブジェクトの数。SQL/MX では、このカウンターはサポートされていません。

カウンタータイプ：増分。

SQL-STMT-RECOMPILE-TIME

このプロセスが文の再コンパイルに費やした時間。

カウンタータイプ：経過時間。

SQL-STMT-RECOMPILES

このプロセスに対して再コンパイルされた文の数。SQL-STMT-RECOMPILES カウンターは、類似性チェックが失敗し、自動再コンパイルが行われるとインクリメントされます。類似性チェックと自動再コンパイルについては、*SQL/MX Programming Manual for C and COBOL* で説明されています。

カウンタータイプ：増分。

新しい形式の SQLPROC エンティティの使用上の注意

ZMSSQLP レコードの SUBSYSTEM-VERSION は、SQL/MP または SQL/MX サブシステムにより提供されます。

SQLSTMT

SQLSTMT エンティティは、SQL プロセス内のすべての SQL 文に関する情報を提供します。SQL 関数を含むプログラムの各プロシージャーまたはモジュール内の SQL 文ごとに 1 つの SQLSTMT カウンターレコードがあります。

注記:

SQL 文の測定のオーバーヘッドコストは、他のエンティティの測定のコストよりも高くなります。実用的に使用するには、アプリケーションプログラムのパフォーマンス調整のためだけに定期的に SQLSTMT を測定します。必ず、調べるプログラムだけを測定するように SQLSTMT を構成してください。

Measure G09 以降の PVU では、SQLSTMT エンティティ操作は SQL/MX と OSS ファイルパス名をサポートします。SQL/MX のプログラムオブジェクトおよび実行単位は、OSS ファイルシステム領域でのみ定義されます。SQL/MP 環境では、SQL プロシージャーおよび実行単位は、それらを実行するプログラムオブジェクトファイルの一部です。ただし、SQL/MX では、SQL プロシージャーおよびモジュールは、それらを実行するプログラムオブジェクトファイルと別々にコンパイルされ保管されます。

Measure G11 以降の PVU では、SQLSTMT エンティティは SQLSTMT レポートで ANSI SQL 名を表示します。

Measure H01 以降の PVU では、MEASCOM LIST コマンドは、SQLSTMT エンティティ指定で、ANSI SQL/MX リリース 2 モジュールオブジェクト名（実行単位名）を受け入れます。

SQL/MX モジュール名（以前には実行単位名と呼ばれていました）は次のように指定できます。

Measure PVU	長さ（文字単位）
G10 以前	最大 128
G11 以降	最大 784

特定のモジュール名が SQL/MX で指定されていない場合、前処理時点でのユリウス暦タイムスタンプに基づいたシステム生成名が割り当てられます。SQL/MX モジュールにおける文の命名およびインデックス指定に関する情報については、プリプロセッサー出力ファイルを参照してください。

トピック

SQLSTMT エンティティのエンティティ指定構文

SQLSTMT エンティティの DDL レコード

SQLSTMT エンティティの DDL フィールドの定義

SQLSTMT エンティティの使用上の注意

SQLSTMT 測定の例

SQLSTMT エンティティのエンティティ指定構文

SQLSTMT エンティティを記述するには、次の構文を使用します。

```
SQLSTMT entity-spec [,entity-spec] ...
```

SQLSTMT

SQL プロセス内の SQL 文についての情報を収集します。

entity-spec

ADD および DELETE コマンドの場合、*entity-spec* は次のように指定します。

```
{ *                                                    }
{ $process-name [ ( pid ) ]                            }
{ disk-filename [ ( pid ) ]                            }
{ pid                                                  }
```

LIST および LISTALL コマンドの場合、*entity-spec* は次のように指定します。

```
{ * ["run unit"[,#index]]                              }
{ * ["run-unit"[[Index]#index]]                        }
{ $process-name [ ( pid ) ] ["run-unit"[,#index]]      }
{ $process-name [ ( pid ) ] ["run-unit"[Index]#index]] }
{ disk-filename [ ( pid ) ] ["run-unit"[,#index]]      }
{ disk-filename [ ( pid ) ] ["run-unit"[[Index] #index]] }
{ "pname" [ ( pid ) ] ["run-unit"[,#index]]            }
{ "pname" [ ( pid ) ] ["run-unit"[[Index] #index]]     }
{ pid ["run-unit"[,#index]]                            }
{ pid ["run-unit"[[Index] #index]]                     }
```

LISTACTIVE コマンドの場合、*entity-spec* は次のように指定します。

```
{ $process-name ( pid ) "run-unit", #index                }
{ $process-name ( pid ) "run-unit" [Index] #index         }
{ disk-filename ( pid ) "run-unit", #index                }
{ disk-filename ( pid ) "run-unit", [Index] #index        }
{ "pname" (pid) "run-unit", #index                        }
{ "pname" (pid) "run-unit" [Index] #index                 }
{ pid "run-unit", #index                                  }
{ pid "run-unit" [Index] #index                           }
```

指定する値は次のとおりです。

*****

ローカルシステム上のすべてのプロセスのすべての SQL 文を測定します。

$*process-name*

測定対象の SQL 文が含まれるプロセスの名前です。

pid

プロセス ID です。プロセス ID は、測定対象のプロセスの所有者を識別します。*pid* は次のように指定します。

```
{ cpu,pin }
```

cpu

プロセスが実行されている CPU の番号です。

pin

プロセスのプロセス識別番号です。すべての PIN を指定するには、アスタリスク（*）を使用します。

run-unit

Measure G09 以前の PVU では、測定対象のプロシージャーの名前です。次に例を示します。

```
+ LIST SQLSTMT * "max32bytelongrununitname"
```

Measure G09 以降の PVU：

- 二重引用符で囲まれた最大 32 文字の SQL/MP プロシージャー名です。次に例を示します。

  ```
  + LIST SQLSTMT * "max32bytelongrununitname"
  + LISTACTIVE SQLSTMT 3,99 "max32bytelongrununitname", #1
  ```

- 二重引用符で囲まれた最大 128 文字の SQL/MX リリース 1 のモジュール名です。次に例を示します。

  ```
  + LIST SQLSTMT * "max128bytelongrununitname"
  + LISTACTIVE SQLSTMT 3,99 "max128bytelongrununitname", #1
  ```

Measure H01 以降の PVU：

- 二重引用符で囲まれた、最大 128 バイトの区切られていない識別子のみを含む SQL/MX リリース 2 のモジュール名です。次に例を示します。

  + **LIST SQLSTMT * "catA.schemaB.moduleM"**

- 単一引用符で囲まれた、SQL/MX でサポートされている最大長（現在は最大 783 文字）の SQL/MX リリース 2 のモジュール名です。次に例を示します。

  + **LIST SQLSTMT * 'catA."schema%B".moduleM'**
  + **LIST SQLSTMT * 'moduleM'**
  + **LISTACTIVE SQLSTMT 3,99 'catA."schema%B".moduleM', #1**

Index

オプションです。Measure G09 以降の PVU で使用する場合、Index は、切り取りと貼り付けのアプリケーションで実行ユニットとインデックス値（SQLSTMT レポートに表示される）を使用できるようにします。

index

測定対象の文のインデックス番号です。省略すると、現在のすべてのインデックス付きの文が測定されます。

disk-filename

測定対象の SQL コードが含まれているオブジェクトファイルの名前です。このファイルは、実行中のプロセスのオブジェクトファイルである必要があります。disk-filename は次のように指定します。

```
{ [ \system.]$device.subvol.filename[:crvsn]}
{ subvol.filename[:crvsn]                   }
{ filename[:crvsn]                          }
```

system

システム名です。すべてのデバイスを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのシステムです。

$*device*

ボリューム（デバイス）の名前です。すべてのボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのボリュームです。

subvol

サブボリューム名です。すべてのサブボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのサブボリュームです。

filename

オブジェクトファイル名です。すべてのファイル（一時ファイルを除く）を指定するには、アスタリスク（*）を使用します。

:*crvsn*

（Measure G09 以降の PVU）一意のファイル名を形成するために必要な、タイムスタンプ、作成バージョンのシリアル番号（CRVSN）、またはファイル名拡張子です。このオプションを使用して、ファイル名の一意性を保証します。CRVSN は Measure レポートおよび LISTGNAME コマンドから使用できます。

"*pname*"

（Measure G09 以降の PVU）完全修飾または部分修飾のいずれかの OSS ファイルパス名です。完全修飾の有効な OSS ファイルパス名を取得するには、*pname* を現在の OSSPATH 設定と組み合わせます。OSS ファイルパス名がバックスラッシュ（/）で始まらない場合は、バックスラッシュの前に OSSPATH の現在の設定を置くことで拡張します。/G で始まる OSS ファイルパス名はサポートされていないため、代わりに Guardian ファイル名を使用してください。

注記:

OSS ファイルパス名は大文字と小文字が区別され、二重引用符（""）で囲んで指定する必要があります。有効な OSS ファイルパス名は、特定のファイル、または特定のディレクトリ内のファイルのセットを参照できます。ディレクトリが指定されている場合は、そのディレクトリ内のファイルのみが含まれます。指定したディレクトリの下位のディレクトリ内のファイルは含まれません。

SQLSTMT エンティティの DDL レコード（ZMS 形式）

SQLSTMT エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmssqls-id.
 02 pin                          type binary 16 unsigned.
 02 priority                     type binary 16 unsigned.
 02 userid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 creatorid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 process-name                 type character 8.
 02 program-file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 osspid                       type binary 32 unsigned.
 02 ancestor-cpu                 type binary 16 unsigned.
 02 ancestor-pin                 type binary 16 unsigned.
 02 ancestor-sysname             type character 8.
 02 ancestor-process-name        type character 8.
 02 device-name                  type character 8.
 02 program-file-name-MID.
    03 PATHID                    type character 24.
    03 CRVSN                     type character 6.
 02 statement-index              type binary 16 unsigned.
 02 GMOM.
    03 GMOM-node                 type binary 16 unsigned.
    03 GMOM-cpu                  type binary 16 unsigned.
    03 GMOM-pin                  type binary 16 unsigned.
    03 GMOM-jobid                type binary 16 unsigned.
 02 GMOM-full-id                 type binary 64
                                 redefines GMOM.
 02 GMOM-sysname                 type character 8.
 02 GMOM-process-name            type character 8.
```

```
 02 version                      type binary 16 unsigned.
 02 creator-type                 type binary 16 unsigned.
 02 run-unit-len                 type binary 16 unsigned.
 02 reserved-1                   type character 2.
 02 run-unit                     type character 784.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmssqls-ctrs.
 02 calls                        type binary 64.
 02 elapsed-busy-time            type binary 64.
 02 records-used                 type binary 64.
 02 records-accessed             type binary 64.
 02 disc-reads                   type binary 64.
 02 messages                     type binary 64.
 02 message-bytes                type binary 64.
 02 sorts                        type binary 64.
 02 elapsed-sort-time            type binary 64.
 02 recompiles                   type binary 64.
 02 elapsed-recompile-time       type binary 64.
 02 lock-waits                   type binary 64.
 02 timeouts                     type binary 64.
 02 escalations                  type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmssqls. FILE is "zmssqls" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 ctr                          type zmssqls-ctrs.
 02 id                           type zmssqls-id.
end
```

SQLSTMT エンティティの DDL レコード（Legacy 形式）

SQLSTMT エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD sqlstmt. FILE is "sqlstmt" ENTRY-SEQUENCED.
                      .
                      .
                      .
```

error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)

```
                      .
                      .
                      .
* entity identification items:
 02 pin                          type binary 16 unsigned.
 02 process-name                 type character 8.
 02 program-file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 priority                     type binary 16 unsigned.
 02 userid.
```

```
    03 group                      type binary 8 unsigned.
    03 user                       type binary 8 unsigned.
* counter record identifiers and counter values:
 02 run-unit                     type character 32.
 02 statement-index              type binary 16 unsigned.
 02 calls                        type binary 32.
 02 elapsed-busy-time            type binary 64.
 02 records-used                 type binary 32 unsigned.
 02 records-accessed             type binary 32 unsigned.
 02 disc-reads                   type binary 32 unsigned.
 02 messages                     type binary 32 unsigned.
 02 message-bytes                type binary 32 unsigned.
 02 sorts                        type binary 16 unsigned.
 02 elapsed-sort-time            type binary 64.
 02 recompiles                   type binary 16 unsigned.
 02 elapsed-recompile-time       type binary 64.
 02 lock-waits                   type binary 16 unsigned.
 02 timeouts                     type binary 16 unsigned.
 02 escalations                  type binary 16 unsigned.
 02 creatorid.
    03 group                    type binary 8 unsigned.
    03 user                     type binary 8 unsigned.
* new entity identification items for D10:
 02 ancestor-cpu                 type binary 16 unsigned.
 02 ancestor-pin                 type binary 16 unsigned.
 02 ancestor-sysname             type character 8.
 02 ancestor-process-name        type character 8.

* SMS changes:
 02 device-name                  type character 8.

* F40 new counter value items:
 02 message-bytes-f              type binary 64.
* G02 new counter value items:
 02 calls-f                      type binary 64.
 02 records-used-f               type binary 64.
 02 records-accessed-f           type binary 64.
 02 disc-reads-f                 type binary 64.
 02 messages-f                   type binary 64.*

* New identifiers for OSS file pathname suport:
 02 osspid                       type binary 32 unsigned.
 02 program-file-name-mid.
    03 pathid                   type character 24.
    03 crvsn                    type character  6.

* New identifiers for SQL/128Support
 02 run-unit-128                 type character 128.

* New identifiers for NetBatch Job Control:
 02 GMOM.
    03 GMOM-node                type binary 16 unsigned.
    03 GMOM-cpu                 type binary 16 unsigned.
    03 GMOM-pin                 type binary 16 unsigned.
    03 GMOM-jobid               type binary 16 unsigned.
 02 GMOM-full-id                 type binary 64
                                redefines GMOM.
```

```
 02 GMOM-sysname                 type character 8.
 02 GMOM-process-name            type character 8.

* New Version Field
 02 VERSION                      type binary 16 unsigned.
 02 CREATOR-TYPE                 type binary 16 unsigned.

* New identifiers for full ANSI SQL name
 02 FULL-NAME-OFFSET             type binary 16 unsigned.
 02 FULL-NAME-LEN                type binary 16 unsigned.

* Variable data goes here, pointed to by VARSTRING (offset/length)

* VAR-DATA needs to be defined for the maximum expected variable length data.

 02 FULL-NAME                    type character 784.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

SQLSTMT エンティティの共通および ZMS 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧にしています。サブフィールドは、フィールドに指定する順に一覧にしています。

ANCESTOR-CPU

親プロセスが存在する CPU の番号。

ANCESTOR-PIN

親プロセスのプロセス識別番号。

ANCESTOR-PROCESS-NAME

親プロセスの名前。

ANCESTOR-SYSNAME

親プロセスが存在するシステムの名前。

CREATOR-TYPE

作成者タイプ：0 = 使用されません。1 = SQL/MP。2 = SQL/MX 1.*x*。3 = SQL/MX 2.*x*。

CREATORID

作成者アクセス ID（CAID）。このフィールドはプロセスの作成者を識別し、GROUP と USER の 2 つのサブフィールドに分かれています。

DEVICE-NAME

プログラムファイルが配置されているディスクデバイス。SMF ファイルの場合、このフィールドには、論理ファイル名に対応する物理的な場所が表示されます。NonStop ファイルの場合、このフィールドは、FILE-NAME VOLUME サブフィールドと同じです。

GMOM

GMOM 構造は以下の 4 つのサブフィールドで構成されます。

GMOM-NODE	GMOM プロセスが NetBatch ジョブの一部であり、GMOM プロセスがリモートである場合、GMOM プロセスの Expand ノード番号。そうでない場合はゼロ。
GMOM-CPU	GMOM プロセスが NetBatch ジョブの一部である場合、GMOM プロセスのプロセッサー番号。そうでない場合はゼロ。
GMOM-PIN	GMOM プロセスが NetBatch ジョブの一部である場合、GMOM プロセスの PIN。そうでない場合はゼロ。
GMOM-JOBID	指定された GMOM によって割り当てられる、このプロセスを起動した NetBatch ジョブのジョブ ID。そうでない場合はゼロ。

GMOM-FULL-ID

個々の GMOM フィールドの単一値としての 64 ビットの再定義。ゼロの値は、プロセスが NETBATCH ジョブストリームの一部でないことを示します。

GMOM-PROCESS-NAME

このプロセスを同様に開始した、NETBATCH ジョブを開始した GMOM プロセスの名前。そうでない場合は空白。

GMOM-SYSNAME

GMOM プロセスがリモートの場合、GMOM ノードの Expand システム名。そうでない場合は空白。

OSSPID

このプロセスが OSS 環境で実行される場合、このフィールドには OSS プロセス ID が含まれます。OSSPID が非 OSS プロセスである場合、このフィールドにはゼロが含まれます。

PIN

測定対象のプロセスのプロセス識別番号。

PRIORITY

測定対象のプロセスの作成優先順位。ALTPRI TACL コマンドを使用してプロセスの優先順位を変更しても、この値には影響しません。

PROCESS-NAME

測定対象のプロセスの名前。

PROGRAM-FILE-NAME

測定対象のプロセスのオブジェクトファイルの名前。この名前は、NonStop ファイルか SMF ファイルのいずれかに適用できます。PROGRAM-FILE-NAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、プログラムファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、プロセスが実行されたときに指定された物理ファイル名を表します。 SMF ファイルでは、プロセスが実行されたときに使用された、位置に依存しない論理ファイル名を表します。

PROGRAM-FILE-NAME-MID

OSS ファイルパス名の内部形式表現。PROGRAM-FILE-NAME-MID には次の 2 つのサブフィールドがあります。

PATHID	OSS ファイルパス名の内部形式表現です。非 OSS ファイルの場合、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。非 OSS ファイルの場合、このフィールドにはゼロが含まれます。

RUN-UNIT

（ZMS 形式のみ）左詰めにした外部形式の ANSI SQL 名。このフィールドは 784 文字として定義されていますが（配置の理由で切り上げ）、実際の名前の長さは RUN-UNIT-LEN で報告されます。残りの文字はゼロで初期化されます。

RUN-UNIT-LEN

（ZMS 形式のみ）RUN-UNIT フィールドに保存されている外部形式の ANSI SQL 名の長さ（バイト単位）。最大値は 783 です。

STATEMENT-INDEX

指定した実行単位内での文のインデックス番号（0～255）。これらの番号は、コンパイラのリスト表示でのセクション位置テーブル（SLT）のインデックス番号に対応します。SLT インデックスを使用して、STATEMENT‑INDEX 番号を、ソースコードの対応する EXEC SQL 文に関連付けます。

各 SQL 文の SLT インデックス番号を確認するには、SQL コンパイラ指示文の SQLMAP オプションを使用します。SQLMAP オプションを使用すると、SQL マップをリストの最後に表示できます。SQL マップは、各 SQL 文の SLT インデックス番号とともにソースファイル名および行番号を表示したテーブルです。

詳しくは、使用している言語に応じた SQL/MP のプログラミングマニュアルを参照してください。

USERID

測定対象のプロセスのプロセスアクセサー ID。

VERSION

レコードバージョン：0、1、2 = 使用されません。3 = G11 PVU。

SQLSTMT エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。記載されているのは、ZMS 形式の DDL レコードのみで使用されるフィールドです。

フィールドはアルファベット順に一覧しています。

CALLS

測定対象の SQL 文が実行された回数。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、64 ビットのカウンターです。

G シリーズの RVU では、CALLS-F フィールドは CALLS の 64 ビットバージョンです。

カウンタータイプ：増分。

DISC-READS

この文に対して実行される物理ディスク読み取りの回数。REPORT RATE 値がオンの場合、このレートは 1 秒あたりのディスク読み取り数です。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、64 ビットのカウンターです。

G シリーズの RVU では、DISC-READS-F フィールドは DISC-READS の 64 ビットバージョンです。

カウンタータイプ：増分。

ELAPSED-BUSY-TIME

プロセスが測定対象の SQL 文の実行に費やした時間。

REPORT RATE がオンの場合、出力値はこの期間中のビジー時間の割合です。

ELAPSED-BUSY-TIME カウンター値を CALLS カウンター値で除算することによって、呼び出しごとの平均経過時間を計算できます。

カウンタータイプ：経過時間。

ELAPSED-RECOMPILE-TIME

プロセスがこの文の再コンパイルに費やした時間。

カウンタータイプ：経過時間。

ELAPSED-SORT-TIME

この文のソート時にプロセスが費やした時間。

カウンタータイプ：経過時間。

ESCALATIONS

レコードロックがファイルロックにエスカレーションされた回数。

カウンタータイプ：増分。

LOCK-WAITS

このステートメントがロック要求を待機した回数。

カウンタータイプ：増分。

MESSAGE-BYTES

この文についてシステムが送信したメッセージのバイト数。このカウンターは、MESSAGES カウンターで報告されたメッセージについて送信されたバイト数を累積します。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、64 ビットのカウンターです。

G シリーズの RVU では、MESSAGE-BYTES-F フィールドが MESSAGE-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

MESSAGES

ファイルシステムからディスクプロセスに送信されたメッセージや、再コンパイルのために SQL エグゼキューターから SQL コンパイラに送信されたメッセージを含む、この文に代わってシステムにより送信されたメッセージの数。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、64 ビットのカウンターです。

G シリーズの RVU では、MESSAGES-F フィールドが MESSAGES の 64 ビットバージョンです。

カウンタータイプ：増分。

RECOMPILES

文が再コンパイルされた回数。

カウンタータイプ：増分。

RECORDS-ACCESSED

この文を評価するためにディスクプロセスまたはファイルシステムがアクセスするレコードの数。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、64 ビットのカウンターです。

G シリーズの RVU では、RECORDS-ACCESSED-F フィールドは RECORDS-ACCESSED の 64 ビットバージョンです。

カウンタータイプ：累積。

RECORDS-USED

この文を通じて SQL エグゼキューターが挿入、更新、削除、または読み取ったレコードの数。

G シリーズ以前の RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、64 ビットのカウンターです。

G シリーズの RVU では、RECORDS-USED-F フィールドは RECORDS-USED の 64 ビットバージョンです。

カウンタータイプ：累積。

SORTS

この文に対して外部ソートプロセスが呼び出された回数。

カウンタータイプ：増分。

TIMEOUTS

要求がタイムアウトした回数。タイムアウトは、ロック待機か、ディスクプロセスやネットワークの輻輳のいずれかによって生じます。

カウンタータイプ：増分。

SQLSTMT エンティティの Legacy 形式のみの ID フィールドの DDL 定義

これらの ID フィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

FULL-NAME

（Legacy 形式のみ）左詰めにした外部形式の ANSI SQL 名。このフィールドは 784 文字として定義されますが、実際の名前の長さは FULL-NAME-LEN で報告されます。残りの文字は定義されません。

FULL-NAME-LEN

（Legacy 形式のみ）FULL-NAME フィールドに保存されている外部形式の ANSI SQL 名のバイト単位の長さ。最大値は 783 です。

FULL-NAME-OFFSET

（Legacy 形式のみ）FULL-NAME フィールドの SQLSTMT レコードの先頭からのオフセット（バイト単位）。

RUN-UNIT

（Legacy 形式のみ）測定対象の SQL 文を含むプロシージャーの名前。SQL/MX では、プロシージャーの実際の名前は許可されている 32 文字を超えた場合、ERROR フィールドでフィールドオーバーフローが報告され、最下位 31 バイト（前に波ダッシュ（~）が付く）のみが含められます。フルネームは RUN-UNIT-128 フィールドから取得できます。**SQLSTMT エンティティの DDL レコード（Legacy 形式）**(426 ページ)を参照してください。

RUN-UNIT-128

（Legacy 形式のみ）測定対象の SQL 文を含む SQL/MP プロシージャーまたは SQL/MX モジュールの名前。

SQLSTMT エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

CALLS-F

（G シリーズ、Legacy 形式のみ）CALLS と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

DISC-READS-F

（G シリーズ、Legacy 形式のみ）DISC-READS と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

MESSAGE-BYTES-F

（G シリーズ、Legacy 形式のみ）MESSAGE-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

MESSAGES-F

（G シリーズ、Legacy 形式のみ）MESSAGES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

RECORDS-ACCESSED-F

（G シリーズ、Legacy 形式のみ）RECORDS-ACCESSED と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

RECORDS-USED-F

（G シリーズ、Legacy 形式のみ）RECORDS-USED と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

すべての SQLSTMT エンティティの使用上の注意

- SQLSTMT エンティティの SQLPROC エンティティを明示的に追加する必要はありません。SQLSTMT エンティティを測定に追加すると、測定対象のプロセスに対応する SQLPROC エンティティが自動的に追加されます。SQLPROC エンティティを使用して、測定対象の SQL プロセスに関する詳細を取得できます。ただし、Measure は、SQLSTMT エンティティが追加されても、PROCESS エンティティを自動的には追加しません。PROCESS エンティティ情報も必要な場合は、PROCESS エンティティを具体的に追加する必要があります。
- ZMSSQLS レコードの SUBSYSTEM-VERSION は、SQL/MP または SQL/MX サブシステムにより提供されます。
- ファイル名の指定時に、アスタリスクをワイルドカードとして使用できます。ただし、このアスタリスクは要素全体（ボリューム、サブボリューム、またはファイル名）を置き換えます。

  または$.*.*は、すべてのファイルを意味します。

  $*vol*.*.*は、指定したボリューム内のすべてのファイルを意味します。

  $*vol.subvol*.*は、指定したボリュームの指定したサブボリューム内のすべてのファイルを意味します。

  $*.*subvol*.*は、すべてのボリュームの指定したサブボリューム内のすべてのファイルを意味します。

  $*vol*.*.*file* は、指定したボリュームのすべてのサブボリューム内の指定したファイルを意味します。

  $*.*.*file* は、すべてのボリュームのすべてのサブボリューム内の指定したファイルを意味します。
- SQL/MP アプリケーションを監視するときに、確実に SQLSTMT エンティティをキャプチャーするように、SQL アプリケーションプロセスを起動する前に SQLSTMT エンティティ測定を開始して、SQL プロセスが停止してから SQLSTMT エンティティ測定を停止してください（つまり、SQL アプリケーションの実行全体に及ぶように、SQLSMT エンティティ測定を開始および停止します）。

H シリーズ以降の SQLSTMT エンティティの使用上の注意

H シリーズ以降では、すべてのバイト数フィールドで 64 ビットに対応しています。-F で終わるフィールドは、ZMS 形式レコードではもはや使用されていませんが、Legacy 形式でデータを要求するアプリケーションでは引き続き使用できます。

G シリーズ SQLSTMT エンティティの使用上の注意

G シリーズでは、64 ビットのバイト数フィールド（-F で終わるフィールド）は、従来の 32 ビットのバイト数フィールドと同じデータを収集します。たとえば、64 ビットフィールドの MESSAGE-BYTES-F は、32 ビットフィールドの MESSAGE-BYTES と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

32 ビットフィールドは、現在アクティブであり、値を返し続けます （フィールドオーバーフローは、オーバーフローしたフィールドと測定対象のエンティティの ERROR フィールドの両方で、-1 の値で示され

ます）。ただし、64 ビットフィールドを使用するようにアプリケーションを変換する必要があります。32 ビットフィールドは、今後の RVU では使用できなくなる可能性があります。

MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットのフィールドの名前を使用します。たとえば、LIST DEVICE BY INPUT-BYTES-F ではなく LIST DEVICE BY INPUT-BYTES のコマンドを発行します。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。

例

SQLSTMT エンティティのレポート出力には、名前タイプに応じて二重引用符の SQL 実行ユニット名または一重引用符の ANSI SQL モジュール名が含まれています。次に例を示します。

```
SQL Statement
Procedure 'CATALOG_12.SCHEMA_34.MODULE_56'   Index #
Process                        Pri            OSSPID:
Program
OSSPath:
Userid             Creatorid           Ancestor
Format Version:  H01  Data Version:  H01  Subsystem Version:  1
Local System
------------------------------------------------------------------------------
Elapsed-Busy-Time            4.17 ms   Calls                          0.17 /s
Elapsed-Sort-Time                      Sorts
Elapsed-Recompile-Time                 Recompiles
Messages                     0.22 /s   Message-Bytes                931.00 /s
Lock-Waits                             Escalations
Timeouts                               Disc-Reads
Records-Accessed             0.17 /s   Records-Used                   0.17 /s
```

SYSTEM

SYSTEM エンティティは、Expand 回線ハンドラーを通じてネットワークトラフィックに関する情報を提供します。ネットワークリンクには、リンクを開始したリンカーと、リンクを受け入れたリスナーの少なくとも 2 つのプロセスが含まれます。Measure サブシステムは、ローカルシステムしか測定できないので、測定データは、各リンクの片側（リンカーまたはリスナー）のみを反映し、両方は反映しません。

どのネットワークリンクが測定されるかは、測定構成でのエンティティ指定によって異なります。

- すべてのシステムを指定すると、Measure サブシステムは、ローカルシステムとリモートシステム間で渡されるすべてのメッセージのローカル側をカウントします。
- ローカル CPU を指定すると、Measure サブシステムは、指定した CPU 内の回線ハンドラープロセス（プライマリまたはバックアップ）を通じて、ローカルシステムとリモートシステム間で渡されるすべてのメッセージのローカル側をカウントします。
- リモートシステムを指定すると、Measure サブシステムは、ローカルシステムとリモートシステム間で渡されるすべてのメッセージのローカル側をカウントします。
- マルチ CPU パス（superpath）の場合、Measure サブシステムはアクティブなパスを 1 つだけレポートします。

エンティティ指定構文と DDL レコードの Legacy 形式は、G シリーズの PVU SYSTEM エンティティと D シリーズの PVU SYSTEM エンティティで同じです。

トピック

SYSTEM エンティティのエンティティ指定構文

SYSTEM エンティティの DDL レコード

SYSTEM エンティティの DDL フィールドの定義

SYSTEM エンティティの使用上の注意

SYSTEM エンティティのエンティティ指定構文

SYSTEM エンティティを記述するには、次の構文を使用します。

```
SYSTEM entity-spec [,entity-spec] ...
```

SYSTEM

ネットワークトラフィックに関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                     }
{ \system [ ( cpu ) ] }
```

指定する値は次のとおりです。

*****

ローカルシステムとすべてのリモートシステム間のトラフィックを測定します。

system

ネットワークトラフィックを測定するリモートシステムのシステム名またはシステム番号です。

cpu

ネットワークトラフィックを測定するリモートシステム上の CPU の番号です。すべての CPU を指定するには、アスタリスク（*）を使用します。デフォルトは、すべての CPU です。

すべての SYSTEM エンティティの DDL レコード（ZMS 形式）

SYSTEM エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmssys-id.
 02 remote-system-name          type character 8.
 02 remote-system               type binary 16 unsigned.
 02 reserved-1                  type character 6.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmssys-ctrs.
 02 links                       type binary 64.
 02 link-time                   type binary 64.
```

```
 02 sent                          type binary 64.
 02 received                      type binary 64.
 02 sent-forward                  type binary 64.
 02 received-forward              type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmssys. FILE is "zmssys" ENTRY-SEQUENCED.
 02 hdr                           type zmsheader.
 02 ctr                           type zmssys-ctrs.
 02 id                            type zmssys-id.
end
```

すべての SYSTEM エンティティの DDL レコード（Legacy 形式）

SYSTEM エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD system. File is "system" ENTRY-SEQUENCED.
                        .
                        .
                        .
```

error、time items、および measurement identification items については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください)

```
                        .
                        .
                        .
* entity identification items:
 02 remote-system                type binary 16 unsigned.
 02 remote-system-name            type character 8.
* counter value items:
 02 links                         type binary 32 unsigned.
 02 link-time                     type binary 64.
 02 sent                          type binary 32 unsigned.
 02 received                      type binary 32 unsigned.
 02 sent-forward                  type binary 32 unsigned.
 02 received-forward              type binary 32 unsigned.

end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

SYSTEM エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧しています。

REMOTE-SYSTEM

ネットワークトラフィックが測定されたリモートシステムのシステム番号。

REMOTE-SYSTEM-NAME

ネットワークトラフィックが測定されたリモートシステムのシステム名。

SYSTEM エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧しています。

LINK-TIME

ローカルシステムによってリモートシステムに送信されるメッセージの、リモートシステムからの肯定応答を待機するのに消費された時間。

カウンタータイプ：Syslink。

LINKS

ローカルシステムからリモートシステムに送信されるメッセージの数（それぞれ 1 つまたは複数のパケットで構成される）。このカウンターは、レベル 4、ファイルシステムインターフェイス、肯定応答メッセージを含みません。

カウンタータイプ：増分。

RECEIVED

ローカルシステムを最終的な宛先として、リモートシステムからローカルシステムに送信されたパケットの数。RECEIVED-FORWARD カウンターは、第 3 のシステムを宛先とするパケットをカウントします。

カウンタータイプ：増分。

RECEIVED-FORWARD

ネットワーク内の別のシステムに転送される、リモートシステムからローカルシステムに送信されたパケットの数。RECEIVED カウンターは、ローカルシステムを最終宛先とするパケットをカウントします。

カウンタータイプ：増分。

SENT

ローカルシステムが作成し、リモートシステムに送信したパケットの数。SENT-FORWARD カウンターは、ネットワーク内の別のシステムに代わってローカルシステムが送信したパケットをカウントします。

カウンタータイプ：増分。

SENT-FORWARD

ネットワーク内の別のシステムの代わりに、ローカルシステムからリモートシステムに送信されたパケットの数。

カウンタータイプ：増分。

All SYSTEM エンティティの使用上の注意

- SYSTEM カウンターは、Expand 回線ハンドラーによってリンクされたシステム間でのネットワーク操作中に、ネットワーク回線ハンドラープロセスによって変更されます。これらの操作には、リモートシステムへのログオンやリモートシステムへのファイルの移動があります。プロトコルにより、使用される Expand 回線ハンドラープロセスが決まります。
- ZMSSYS レコードの SUBSYSTEM-VERSION は Expand によって提供されます。
- 次のエンティティタイプが関連しています。

- CLUSTER は FOX トラフィックを測定します。
- NETLINE はネットワーク通信回線を測定します。

- ネットワークおよびネットワークプロトコルについて詳しくは、*Expand Configuration and Management Manual*、*Expand Network Management and Troubleshooting Manual*、および *X25AM Configuration and Management Manual* を参照してください。

TERMINAL

TERMINAL エンティティは、端末 I/O に関する情報を提供します。

トピック

TERMINAL エンティティのエンティティ指定構文

TERMINAL エンティティの DDL レコード

TERMINAL エンティティの DDL フィールドの定義

TERMINAL エンティティの使用上の注意

TERMINAL エンティティのエンティティ指定構文

TERMINAL エンティティを記述するには、次の構文を使用します。

```
TERMINAL entity-spec [,entity-spec] ...
```

注記:

すべての TERMINAL エンティティ（ADD TERMINAL *）を指定しているか、*$device* または*$device (cpu)*のみを指定している場合は、サブデバイス名があってもなくても、D シリーズ RVU の測定アプリケーションで G シリーズ PVU の TERMINAL エンティティを測定できます。アプリケーションが *controller*、*channel*、または *unit* を指定している場合は、G シリーズ PVU の TERMINAL エンティティを測定するようにエンティティ識別子を変更する必要があります。

TERMINAL

ローカルシステム上の 1 つまたは複数の端末に関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                                                                    }
{ $device[.#subdevice ]                                                }
{ cpu [ ( %Htrackid [, clip [, line ] ] ) ] [ ( type [, subtype ] ) ] }
```

また、D シリーズのデータのみの場合は、次のように指定します。

```
{ $device [ ( cpu [ , channel [ , ctrl [ , unit ] ] ] ) ]          }
```

指定する値は次のとおりです。

*****

すべてのプロセッサーのすべての回線（端末接続）を測定します。

$device

測定対象の回線のデバイス（回線）名です。すべての回線を指定するには、アスタリスク（*）を使用します。

#subdevice

測定対象の回線のサブデバイス名です。すべてサブデバイスを指定するには、アスタリスク（*）を使用します。

cpu

測定する回線が構成されている CPU の番号です。デフォルトは、すべての CPU です。

%Htrackid

（G シリーズ）3 つの CLIP の特定の 3880 SWAN コンセントレーターのエンクロージャーの識別子です。（各 CLIP は 2 回線を制御します。）すべての TRACKID を指定するには、アスタリスク（*）を使用します。

TRACKID は、コントローラーの SEEPROM に記録されます。通常はエンクロージャーの外部のラベルにも印字されています。

clip

（G シリーズ）SWAN コンセントレーターのエンクロージャー内の特定の CLIP を示す場合は 1、2、または 3、SWAN 2 コンセントレーターのエンクロージャー内の特定の CLIP を示す場合は 1、2、3、4、5、または 6 です。すべての CLIP を指定するには、アスタリスク（*）を使用します。

line

（G シリーズ）3880 SWAN コンセントレーターの CLIP によって管理される特定の回線を示す場合は、0 または 1 です。すべての回線を指定するには、アスタリスク（*）を使用します。

type

（G シリーズ）特定のプロトコルを示す場合は、61（X.25）などのタイプ番号です。回線タイプの有効な値を確認するには、SCF LISTDEV を使用します。すべての回線タイプを指定するには、アスタリスク（*）を使用します。

subtype

（G シリーズ）62（X25AM）などのサブタイプ番号です。有効なサブタイプ番号を確認するには、SCF LISTDEV を使用します。デフォルトは、*type* に適用されるすべてのサブタイプです。

chan

（D シリーズ）測定する端末のチャネル番号です。デフォルトは、すべてのチャネルです。

ctrl

（D シリーズ）測定する端末のコントローラー番号です。デフォルトは、すべてのコントローラーです。

unit

（D シリーズ）測定する端末のユニット番号です。デフォルトは、すべてのユニットです。

TERMINAL エンティティの DDL レコード（ZMS 形式）

TERMINAL エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmsterm-id.
 02 terminal-name.
    03 line-name                type character 8.
    03 subdev-name              type character 8.
 02 logical-device              type binary 32 unsigned.
 02 reserved-1                  type character 4.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsterm-ctrs.
 02 requests                    type binary 64.
 02 reads                       type binary 64.
 02 writes                      type binary 64.
 02 input-bytes                 type binary 64.
 02 output-bytes                type binary 64.
 02 transactions                type binary 64.
 02 response-time               type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsterm. FILE is "zmsterm" ENTRY-SEQUENCED.
 02 hdr                         type zmsheader.
 02 ctr                         type zmsterm-ctrs.
 02 id                          type zmsterm-id.
end
```

TERMINAL エンティティの DDL レコード（Legacy 形式）

TERMINAL エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

G10 Measure PVU からの G シリーズの TERMINAL エンティティの Legacy 形式の DDL レコードは、次の点を除いて、D シリーズの TERMINAL エンティティと同じです。

- フィールドのオーバーフローの可能性を減らすために、長い byte-count フィールドが提供されます。G シリーズの RVU では、32 ビットの各 byte-count フィールドに対応する 64 ビットがあります。64 ビットのフィールドの使用については、**G シリーズ TERMINAL エンティティの使用上の注意**(444 ページ)

  を参照してください。
- ERROR フィールドが、32 ビットの byte-count フィールドのフィールドオーバーフローを通知することがあります。

```
RECORD terminal. FILE is "terminal" ENTRY-SEQUENCED.
                      .
                      .
                      .
error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)
                      .
                      .
                      .
* entity identification items:
 02 terminal-name.
    03 line-name                type character 8.
```

```
    03 subdev-name                type character 8.
 02 logical-device                type binary 16 unsigned.
* counter value items:
 02 requests                      type binary 32 unsigned.
 02 reads                         type binary 32 unsigned.
 02 writes                        type binary 32 unsigned.
 02 input-bytes                   type binary 32 unsigned.
 02 output-bytes                  type binary 32 unsigned.
 02 transactions                  type binary 32 unsigned.
 02 response-time                 type binary 64.
* F40 new counter value items:
 02 input-bytes-f                 type binary 64.
 02 output-bytes-f                type binary 64.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

TERMINAL エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧にしています。サブフィールドは、フィールドに指定する順に一覧にしています。

LOGICAL-DEVICE

測定された端末の論理デバイス番号。

TERMINAL-NAME

測定対象の端末のデバイス名。このフィールドは、LINE-NAME（デバイス名）と SUBDEV-NAME（サブデバイス名）の 2 つのサブフィールドに分かれています。

TERMINAL エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧しています。

INPUT-BYTES

端末から読み取られたバイト数。端末へ、または端末から転送されたバイトの合計数は、このカウンターに OUTPUT-BYTES を加えた値です。SNAX 行の場合、このカウンターには、6 バイトの伝送ヘッダー（TH）と 3 バイトの要求/応答（RH）SNA ヘッダーが含まれます。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、64 ビットのカウンターです。

G シリーズの RVU では、INPUT-BYTES-F フィールドが INPUT-BYTES の 64 ビット版です。

カウンタータイプ：累積。

OUTPUT-BYTES

端末に書き込まれたバイト数。

端末間で相互転送されたバイトの合計数は、INPUT-BYTES にこのカウンターを加えた値になります。

I/O 操作の前に端末プロセスがこのカウンターを変更するので、書き込み命令が失敗した場合、このカウンターは正確でなくなることがあります。

SNAX 行の場合、このカウンターには、6 バイトの伝送ヘッダー（TH）と 3 バイトの要求/応答（RH）SNA ヘッダーが含まれます。

D シリーズと G シリーズの RVU では、これは 32 ビットのカウンターです。H シリーズ以降の RVU では、64 ビットのカウンターです。

G シリーズの RVU では、OUTPUT-BYTES-F フィールドが OUTPUT-BYTES の 64 ビットバージョンです。

カウンタータイプ：累積。

READS

端末プロセスによって実行された（端末からメモリへの）読み取り操作の数。

カウンタータイプ：増分。

REQUESTS

端末プロセスで受信した要求の数。

端末プロセスが読み取るまで要求がキューに入れられていた時間を特定するには、プロセスを測定し、PROCESS RECV-QTIME カウンターを調べます。

カウンタータイプ：増分。

RESPONSE-TIME

端末プロセスが端末の応答で費やした時間。

トランザクションでは、端末からの読み取りの後に端末への書き込みが行われます。端末の応答とは、読み取りが終わってから書き込みが始まるまでの間隔です。

ブロックモードの操作は、ファンクションキーを読み取ってからデータを読み取ります。このカウンターには、2 つの読み取りの間の時間は含まれません。

SNAX 行では、このカウンターは常にゼロです。

カウンタータイプ：応答時間。

TRANSACTIONS

端末プロセスによって実行される端末トランザクション（読み取りの直後に書き込みが続く）の数。

RESPONSE-TIME は、トランザクションで費やした時間を測定します。

SNAX 行では、このカウンターは常にゼロです。

カウンタータイプ：増分。

WRITES

端末プロセスによって実行された（メモリから端末への）書き込み操作の数。

カウンタータイプ：増分。

TERMINAL エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

INPUT-BYTES-F

（G シリーズ、Legacy 形式のみ）INPUT-BYTES と同じですが、より大きい値（32 ビットではなく 64 ビット）に対応します。

カウンタータイプ：累積。

OUTPUT-BYTES-F

（G シリーズ、Legacy 形式のみ）OUTPUT-BYTES と同じですが、より大きい値（32 ではなく 64 ビット）に対応します。

カウンタータイプ：累積。

すべての TERMINAL エンティティの使用上の注意

TERMINAL エンティティを使用して X25AM 通信回線上のサブデバイスを測定する場合、そのサブデバイスも FILE エンティティとして測定できます。

H シリーズ以降の TERMINAL エンティティの使用上の注意

H シリーズ以降の RVU では、すべてのバイト数フィールドで 64 ビットに対応しています。-F で終わるフィールド名は、ZMS 形式レコードではもはや使用されていませんが、Legacy 形式でデータを要求するアプリケーションでは引き続き使用できます。

G シリーズ TERMINAL エンティティの使用上の注意

- ZMSTERM レコードの SUBSYSTEM-VERSION は、各 ZMSLINE レコードによって表された回線（SNAX/XF、X25AM、AM3270 など）の特定の製品サブシステムによって与えられます。
- 64 ビットのバイト数フィールド（-F で終わるフィールド）は、従来の 32 ビットのバイト数フィールドと同じデータを収集します。たとえば、64 ビットフィールドの INPUT-BYTES-F は、32 ビットフィールドの INPUT-BYTES と同じデータを収集します。64 ビットフィールドは、高レベルの I/O アクティビティによって引き起こされるオーバーフローにあまり左右されません。

  32 ビットフィールドは、現在アクティブであり、値を返し続けます フィールドオーバーフローが存在しない場合、32 ビットフィールドと 64 ビットのフィールドは同じ値を返します。32 ビットフィールドがオーバーフローした場合、対応する 64 ビットフィールドは正しい値を返し、32 ビットフィールドは-1 の値を返します。測定対象のエンティティの ERROR フィールドも、オーバーフロー状態を示す-1 を返します。

  64 ビットフィールドを使用するようにアプリケーションを変換します。32 ビットのフィールドは、今後の RVU では使用できなくなる可能性があります。

  MEASCOM コマンドとコマンド（OBEY）ファイルでは、32 ビットのフィールドの名前を使用します。たとえば、LIST TERMINAL BY INPUT-BYTES-F ではなく LIST TERMINAL BY INPUT-BYTES のコマンドを発行します。MEASCOM は、レポートやプロットなどの出力表示で 32 ビットフィールドの名前を使用します。

  Measure G09 以降の PVU では、SWAN コンセントレーターで最高 6 つの CLIP をサポートします。

TMF

TMF エンティティは、TMF トランザクションに関する情報を提供します。

トピック

TMF エンティティのエンティティ指定構文

TMF エンティティの DDL レコード

TMF エンティティの DDL フィールドの定義

TMF エンティティの使用上の注意

TMF エンティティのエンティティ指定構文

TMF エンティティを記述するには、次の構文を使用します。

```
TMF entity-spec [,entity-spec] ...
```

TMF

ローカルシステム上の 1 つ以上の CPU での TMF トランザクションについて情報を収集します。

entity-spec

次のように指定します。

```
{ * | cpu }
```

指定する値は次のとおりです。

*****

すべての CPU での TMF アクティビティを測定します。

cpu

TMF トランザクションを測定する CPU の番号です。複数の CPU を指定するには、カンマ区切りのリストを使用します。

TMF エンティティの DDL レコード（ZMS 形式）

TMF エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

ID フィールドの DDL 定義

```
DEFINITION zmstmf-id.
 02 reserved-1                  type character 8.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmstmf-ctrs.
 02 home-trans                  type binary 64.
 02 home-trans-qtime            type binary 64.
 02 home-net-trans              type binary 64.
 02 home-net-trans-qtime        type binary 64.
 02 remote-trans                type binary 64.
 02 remote-trans-qtime          type binary 64.
 02 aborting-trans              type binary 64.
 02 trans-backout-qtime         type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmstmf. FILE is "zmstmf" ENTRY-SEQUENCED.
 02 hdr                          type zmsheader.
 02 ctr                          type zmstmf-ctrs.
 02 id                           type zmstmf-id.
end
```

TMF エンティティの DDL レコード（Legacy 形式）

TMF エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

BRIEF レポートに含まれるフィールドは、**太字**で示されています。

```
RECORD tmf. FILE is "tmf" ENTRY-SEQUENCED.
                        .
                        .
                        .
error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ) を参照してください)
                        .
                        .
                        .
* entity identification items:
  (none)

* counter value items:
 02 home-trans                   type binary 32 unsigned.
 02 home-trans-qtime             type binary 64.
 02 home-trans-qmax              type binary 16 unsigned.
 02 remote-trans                 type binary 32 unsigned.
 02 remote-trans-qtime           type binary 64.
 02 remote-trans-qmax            type binary 16 unsigned.
 02 home-net-trans               type binary 32 unsigned.
 02 home-net-trans-qtime         type binary 64.
 02 home-net-trans-qmax          type binary 16 unsigned.
 02 aborting-trans               type binary 32 unsigned.
 02 trans-backout-qtime          type binary 64.
 02 trans-backout-qmax           type binary 16 unsigned.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、一般的なエンティティヘッダーフィールド(177 ページ)を参照してください。

TMF エンティティの共通のカウンターフィールドの DDL 定義

これらのカウンターフィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧しています。

ABORTING-TRANS

中断されたトランザクションの数。中断されたトランザクションは、ローカルシステムで開始してそこに残ったか、ローカルシステムで開始してリモートシステムに配信されたか、リモートシステムで開始してローカルシステムに配信されたかに関係なくカウントされます。

このカウンターは、プライマリ$TMP プロセスが含まれている CPU の TMF レコードでのみ加算されます。プライマリ$TMP が PRIMARY コマンドに応答して CPU を切り替えると、新しいプライマリが含まれる CPU の TMF レコードの ABORTING-TRANS カウンターが加算されます。古いプライマリが含まれる CPU の TMF レコードのカウンターは、そのままになります。

$TMP の PRIMARY スイッチによって、少数の中断するトランザクションが 2 回（古いプライマリで 1 回と新しいプライマリで 1 回）カウントされます。複数の PRIMARY によって、中断するトランザクションが 3 回以上カウントされることがあります。

カウンタータイプ：増分。

HOME-NET-TRANS

このシステムで始まり、後で別のシステムに配信されたトランザクション数。

このカウンターは、プライマリ$TMP プロセスが含まれている CPU の TMF レコードでのみ加算されます。プライマリ$TMP が PRIMARY コマンドに応じて CPU を切り替えると、新しいプライマリが含まれる CPU の TMF レコードの HOME-NET-TRANS カウンターが加算されます。古いプライマリが含まれる CPU の TMF レコードのカウンターは、そのままになります。

ローカルシステムが参加しているネットワークトランザクションの合計数は、すべての CPU 内の HOME-NET-TRANS カウンターと REMOTE-NET-TRANS カウンターの合計です。ローカルシステム上で開始され、別のシステムに配信されなかったトランザクションの数は、HOME-TRANS から HOME-NET-TRANS を引いた数です。

カウンタータイプ：増分。

HOME-NET-TRANS-QTIME

ローカルシステム上で開始され、後でリモートシステムに配信されたトランザクションが、リモートシステムで費やした時間。

このカウンターは、プライマリ$TMP プロセスが含まれている CPU の TMF レコードでのみ加算されます。プライマリ$TMP が PRIMARY コマンドに応答して CPU を切り替えた場合、新しい$TMP は、システムの HOME NETWORK トランザクションの数が正確に反映されるまで、CPU の TMF レコード内の HOME-NET-TRANS-QTIME カウンターをインクリメントします。古いプライマリが含まれている CPU の TMF レコード内の HOME-NET-TRANS-QTIME カウンターがゼロになります。

カウンタータイプ：キュー。

HOME-TRANS

測定された CPU で開始されたトランザクションの数。

このカウンターには、ローカルシステムに残っているかリモートシステムに配信されたかに関係なく、トランザクションが含まれています。このシステムが参加しているトランザクションの合計数は、システム内のすべての CPU についての HOME-TRANS カウンターと REMOTE-TRANS カウンターの合計です。

カウンタータイプ：増分。

HOME-TRANS-QTIME

測定された CPU から派生したトランザクションの有効期間の合計。このカウンターには、ローカルシステムに残っているかリモートシステムに配信されたかに関係なく、トランザクションが含まれています。

カウンタータイプ：キュー。

REMOTE-TRANS

リモートシステムで始まり、測定対象のシステムに分散されたトランザクションの数。

このカウンターは、プライマリ$TMP プロセスが含まれている CPU の TMF レコードでのみ加算されます。プライマリ$TMP が PRIMARY コマンドに応答して CPU を切り替えると、新しいプライマリが含まれる CPU の TMF レコードの REMOTE-TRANS カウンターが加算されます。古いプライマリが含まれる CPU の TMF レコードのカウンターは変わりません。

このシステムが参加しているトランザクションの合計数は、システム内のすべての CPU についての HOME-TRANS カウンターと REMOTE-TRANS カウンターの合計です。

カウンタータイプ：増分。

REMOTE-TRANS-QTIME

リモートシステム上で開始され、ローカルシステムに分散されたトランザクションが、ローカルシステムで費やした時間。

このカウンターは、プライマリ$TMP プロセスが含まれている CPU の TMF レコードでのみ加算されます。プライマリ$TMP が PRIMARY コマンドに応答して CPU を切り替えた場合、新しい$TMP は、システム上のリモートネットワークトランザクションの数が正確に反映されるまで、CPU の TMF レコード内の REMOTE-TRANS-QTIME カウンターの値をインクリメントします。古いプライマリが含まれている CPU の TMF レコード内の REMOTE-TRANS-QTIME カウンターがゼロになります。

カウンタータイプ：キュー。

TRANS-BACKOUT-QTIME

トランザクションが BACKOUT の待機と BACKOUT によるサービス中に費やした時間。トランザクションが中断として指定され、その監査がディスク上に置かれた後、そのトランザクションのバックアウトサービスがスケジュールされ、このカウンターが加算されます。バックアウトがトランザクションの取り消しを完了すると、このカウンターはデクリメントされます。

このカウンターは、プライマリ$TMP プロセスが含まれている CPU の TMF レコードでのみ加算されます。プライマリ$TMP が PRIMARY コマンドに応答して CPU を切り替えた場合、新しい$TMP は、BACKOUT を待機しているローカルシステムのトランザクションの数が正確に反映されるまで、CPU の TMF レコード内の TRANS-BACKOUT-QTIME カウンターをインクリメントします。古いプライマリが含まれている CPU の TMF レコード内の TRANS-BACKOUT-QTIME カウンターがゼロになります。

カウンタータイプ：キュー。

TMF エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

HOME-NET-TRANS-QMAX

（Legacy 形式のみ）HOME-NET-TRANS-QTIME カウンターで記述されるキューの項目の最大数。

カウンタータイプ：最大キュー。

HOME-TRANS-QMAX

（Legacy 形式のみ）HOME-TRANS-QTIME カウンターで記述されるキューの項目の最大数。

カウンタータイプ：最大キュー。

REMOTE-TRANS-QMAX

（Legacy 形式のみ）REMOTE-TRANS-QTIME カウンターで記述されるキューの項目の最大数。

カウンタータイプ：最大キュー。

TRANS-BACKOUT-QMAX

（Legacy 形式のみ）TRANS-BACKOUT-QTIME カウンターで記述されるキューの項目の最大数。

カウンタータイプ：最大キュー。

すべての TMF エンティティの使用上の注意

- ZMSTMF レコードの SUBSYSTEM-VERSION は、TMF サブシステムによって提供されます。

TMF について詳しくは、TMF の一連のマニュアルを参照してください。

- TMF レポートには、TMF カウンターから計算されるフィールドが含まれます。これらのフィールドは、TMF DDL レコードでは定義されていません。

HOME-TRANS-ART

（ZMS 形式のみ）各ホームトランザクションが待機または TMF による処理で費やした平均時間。HOME-TRANS-QTIME は HOME-TRANS によって分割されます。

カウンタータイプ：応答時間。

HOME-NET-TRANS-ART

（ZMS 形式のみ）各要求が待機または TMF による処理で費やした時間。HOME-NET-TRANS-QTIME は、HOME-NET-TRANS で分割されます。

カウンタータイプ：応答時間。

REMOTE-TRANS-ART

（ZMS 形式のみ）各リモートホームネットトランザクションが待機または TMF による処理で費やした平均時間。REMOTE-TRANS-QTIME は REMOTE-TRANS で分割されます。

カウンタータイプ：応答時間。

TRANS-BACKOUT-ART

（ZMS 形式のみ）中断中の各トランザクションが待機または TMF による処理で費やした平均時間。TRANS-BACKOUT-QTIME は、ABORTING-TRANS で分割されます。

カウンタータイプ：応答時間。

USERDEF

USERDEF エンティティは、ユーザー定義のカウンターに関する情報を提供します。

Measure G09 以降の PVU では、USERDEF エンティティタイプは OSS ファイルパス名を扱います。

ZMS 形式のインターフェイス（Measure G11 以降の PVU）では、USERDEF レコードのレコードテンプレート、レポート、および構造化データファイル名は ZMSUDEF です。

トピック

USERDEF エンティティのエンティティ指定構文

USERDEF エンティティの DDL レコード

USERDEF エンティティの DDL フィールドの定義

USERDEF エンティティの使用上の注意

USERDEF エンティティのエンティティ指定構文

USERDEF エンティティを記述するには、次の構文を使用します。

```
USERDEF entity-spec [,entity-spec] ...
```

USERDEF

USERDEF プロセスによって変更された 1 つまたは複数のユーザー定義カウンターに関する情報を収集します。

entity-spec

次のように指定します。

```
{ *                                                    }
{ pid                                                  }
{ $process-name [ (pid) ]                             }
{ [[$device.]subvolume.]filename[:crvsn][ (pid) ]  }
{ "pname" [ (pid) ]                                    }
```

指定する値は次のとおりです。

*****

ローカルシステム上のすべてのユーザー定義のプロセスを測定します。

pid

測定対象のプロセスのプロセス ID です。プロセス ID は、プロセスの所有者を識別します。*pid* は次のように指定します。

```
{ cpu,pin           }
{ SYSTEM-PROCESSES }
```

cpu

測定対象のプロセスが実行されている CPU の番号です。

pin

測定対象のプロセスのプロセス識別番号です。すべてのプロセスを指定するには、アスタリスク（*）を使用します。デフォルトは、すべてのプロセスです。

SYSTEM-PROCESSES

SYSGEN プログラムを使用してインストールしたすべてのプロセスを測定します。

$*process-name*

測定対象のプロセスの名前です。すべてのプロセスを指定するには、アスタリスク（*）を使用します。

$*device*

測定対象のプロセスが含まれるボリューム（デバイス）の名前です。すべてのデバイスを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのデバイスです。

subvolume

測定対象のプロセスが含まれるサブボリュームの名前です。すべてのサブボリュームを指定するには、アスタリスク（*）を使用します。デフォルトは、現在のデフォルトのサブボリュームです。

filename

プロセスのオブジェクトファイルの名前です。*filename* は、実行中のプロセスのオブジェクトファイルである必要があります。すべてのファイル（一時ファイルを除く）を指定するには、アスタリスク（*）を使用します。

:*crvsn*

（Measure G09 以降の PVU）一意のファイル名を形成するために必要な、タイムスタンプ、作成バージョンのシリアル番号（CRVSN）、またはファイル名拡張子です。このオプションを使用して、ファイル名の一意性を保証します。CRVSN は Measure レポートおよび LISTGNAME コマンドから使用できます。

"*pname*"

（Measure G09 以降の PVU）完全修飾または部分修飾のいずれかの OSS ファイルパス名です。スラッシュ（/）で始まらない OSS ファイルパス名は、部分的なパス名とみなされ、OSSPATH の現在の設定を前に指定することで拡張されます。/G で始まる OSS ファイルパス名はサポートされていないため、代わりに Guardian ファイル名を使用してください。

注記:

OSS ファイルパス名は大文字と小文字が区別され、二重引用符（""）で囲んで指定する必要があります。有効な OSS ファイルパス名は、特定のファイル、または特定のディレクトリ内のファイルのセットを参照できます。ディレクトリが指定されている場合は、そのディレクトリ内のファイルのみが含まれます。指定したディレクトリの下位のディレクトリ内のファイルは含まれません。

USERDEF エンティティの DDL レコード（ZMS 形式）

USERDEF エンティティの ZMS 形式の DDL レコードは、Measure G11 以降の PVU でサポートされています。

ID フィールドの DDL 定義

```
DEFINITION zmsudef-id.
 02 pin                          type binary 16 unsigned.
 02 priority                     type binary 16 unsigned.
 02 userid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 creatorid.
    03 group                     type binary 8 unsigned.
    03 user                      type binary 8 unsigned.
 02 process-name                 type character 8.
 02 program-file-name.
    03 volume                    type character 8.
    03 subvol                    type character 8.
    03 filename                  type character 8.
 02 osspid                       type binary 32 unsigned.
 02 ancestor-cpu                 type binary 16 unsigned.
 02 ancestor-pin                 type binary 16 unsigned.
 02 ancestor-sysname             type character 8.
 02 ancestor-process-name        type character 8.
 02 device-name                  type character 8.
 02 program-file-name-MID.
    03 PATHID                    type character 24.
    03 CRVSN                     type character 6.
 02 reserved-1                   type character 2.
 02 GMOM.
    03 GMOM-node                 type binary 16 unsigned.
    03 GMOM-cpu                  type binary 16 unsigned.
    03 GMOM-pin                  type binary 16 unsigned.
    03 GMOM-jobid                type binary 16 unsigned.
 02 GMOM-full-id                 type binary 64
                                 redefines GMOM.
 02 GMOM-sysname                 type character 8.
 02 GMOM-process-name            type character 8.
 02 type                         type binary 16 unsigned.
 02 index                        type binary 16 unsigned.
```

```
 02 reserved-2                    type character 4.
 02 name                          type character 16.
end
```

カウンターフィールドの DDL 定義

```
DEFINITION zmsudef-ctrs.
 02 value                         type binary 64.
end
```

DDL レコード記述フィールド

```
RECORD zmsudef. FILE is "zmsudef" ENTRY-SEQUENCED.
 02 hdr                           type zmsheader.
 02 id                            type zmsudef-id.
 02 ctr                           type zmsudef-ctrs.
end
```

USERDEF エンティティの DDL レコード（Legacy 形式）

USERDEF エンティティの Legacy 形式の DDL レコードは、Measure G10 PVU の後も変更されません。

```
RECORD userdef. FILE is "userdef" ENTRY-SEQUENCED.
                      .
                      .
                      .
```

error、time items、および measurement identification items については、
一般的なエンティティヘッダーフィールド(177 ページ)を参照してください)

```
                      .
                      .
                      .
* entity identification items:
 02 pin                           type binary 16 unsigned.
 02 process-name                  type character 8.
 02 program-file-name.
    03 volume                     type character 8.
    03 subvol                     type character 8.
    03 filename                   type character 8.
 02 priority                      type binary 16 unsigned.
* counter identifiers and value:
 02 name                          type character 16.
 02 type                          type binary 16 unsigned.
 02 index                         type binary 16 unsigned.
 02 max-value                     type binary 16 unsigned.
 02 time-value                    type binary 64.
 02 counts                        redefines time-value.
   03 FILLER                      type binary 32 unsigned.
   03 count-value                 type binary 32 unsigned.
* field for entity identification:
 02 userid.
    03 group                      type binary 8 unsigned.
    03 user                       type binary 8 unsigned.
 02 creatorid.
    03 group                      type binary 8 unsigned.
    03 user                       type binary 8 unsigned.
* entity identification items for D10:
 02 ancestor-cpu                  type binary 16 unsigned.
```

```
 02 ancestor-pin                  type binary 16 unsigned.
 02 ancestor-sysname              type character 8.
 02 ancestor-process-name         type character 8.
* SMS changes:
 02 device-name                    type character 8.
* Identifiers for OSS file pathname support:
 02 osspid                         type binary 32 unsigned.
 02 program-file-name-mid.
    03 pathid                      type binary 64.
    03 crvsn                       type binary 64.
* New identifiers for NetBatch Job Control:
 02 GMOM.
    03 GMOM-node                   type binary 16 unsigned.
    03 GMOM-cpu                    type binary 16 unsigned.
    03 GMOM-pin                    type binary 16 unsigned.
    03 GMOM-jobid                  type binary 16 unsigned.
 02 GMOM-full-id                   type binary 64
                                   redefines GMOM.
 02 GMOM-sysname                   type character 8.
 02 GMOM-process-name              type character 8.
end
```

すべてのエンティティで使用されるヘッダーフィールドの説明については、**一般的なエンティティヘッダーフィールド**(177 ページ)を参照してください。

USERDEF エンティティの共通の ID フィールドの DDL 定義

これらの ID フィールドは、両方の形式の DDL レコードで使用されます。

フィールドはアルファベット順に一覧にしています。サブフィールドは、フィールドに指定する順に一覧にしています。

ANCESTOR-CPU

親プロセスが存在する CPU の番号。

ANCESTOR-PIN

親プロセスのプロセス識別番号。

ANCESTOR-PROCESS-NAME

親プロセスの名前。

ANCESTOR-SYSNAME

親プロセスが存在するシステムの名前。

CREATORID

作成者アクセス ID（CAID）。このフィールドはプロセスの作成者を識別し、GROUP と USER の 2 つのサブフィールドに分かれています。

DEVICE-NAME

プログラムファイルが配置されているディスクデバイス。SMF ファイルの場合、このフィールドには、論理ファイル名に対応する物理的な場所が表示されます。NonStop ファイルの場合、このフィールドは、FILE-NAME VOLUME サブフィールドと同じです。

GMOM

GMOM 構造は以下の 4 つのサブフィールドで構成されます。

GMOM-NODE	GMOM プロセスが NetBatch ジョブの一部であり、GMOM プロセスがリモートである場合、GMOM プロセスの Expand ノード番号。そうでない場合はゼロ。
GMOM-CPU	GMOM プロセスが NetBatch ジョブの一部である場合、GMOM プロセスのプロセッサー番号。そうでない場合はゼロ。
GMOM-PIN	GMOM プロセスが NetBatch ジョブの一部である場合、GMOM プロセスの PIN。そうでない場合はゼロ。
GMOM-JOBID	指定された GMOM によって割り当てられる、このプロセスを起動した NetBatch ジョブのジョブ ID。そうでない場合はゼロ。

GMOM-FULL-ID

個々の GMOM フィールドの単一値としての 64 ビットの再定義。ゼロの値は、プロセスが NETBATCH ジョブストリームの一部でないことを示します。

GMOM-PROCESS-NAME

このプロセスを同様に開始した、NETBATCH ジョブを開始した GMOM プロセスの名前。そうでない場合は空白。

GMOM-SYSNAME

GMOM プロセスがリモートの場合、GMOM ノードの Expand システム名。そうでない場合は空白。

INDEX

カウンターのインデックス値。単一カウンター値のインデックス値は 0 です。配列のインデックス値の範囲は 0〜127 です。

注記:

H07、J05、L シリーズ以降の PVU の配列のインデックス値は 0〜511 の範囲で指定できます。

NAME

ユーザー定義カウンターの名前。

OSSPID

このプロセスが OSS 環境で実行される場合、このフィールドには OSS プロセス ID が含まれます。別のプロセスである場合、このフィールドにはゼロが含まれます。

PIN

測定対象のプロセスのプロセス識別番号。

PRIORITY

測定対象のプロセスの優先順位。

PROCESS-NAME

測定対象のプロセスの名前。

PROGRAM-FILE-NAME

プロセスが実行しているオブジェクトファイルの名前。この名前は、NonStop ファイルか SMF ファイルのいずれかに適用できます。PROGRAM-FILE-NAME には次の 3 つのサブフィールドがあります。

VOLUME	NonStop ファイルでは、物理ファイルのデバイス位置です。 SMF ファイルでは、DEVICE-NAME フィールドに格納されている論理ファイル名に対応する物理ファイルのデバイス位置です。
SUBVOLUME	NonStop ファイルでは、プログラムファイルが置かれている物理ボリュームのデバイス名です。
FILENAME	NonStop ファイルでは、プロセスが実行されたときに指定された物理ファイル名を表します。 SMF ファイルでは、プロセスが実行されたときに使用された、位置に依存しない論理ファイル名を表します。

PROGRAM-FILE-NAME-MID

OSS ファイルパス名の内部形式表現。PROGRAM-FILE-NAME-MID には次の 2 つのサブフィールドがあります。

PATHID	OSS ファイルパス名の内部形式表現です。その他のファイルでは、このフィールドにはゼロが含まれます。
CRVSN	OSS ファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。その他のファイルでは、このフィールドにはゼロが含まれます。

TYPE

ユーザー定義カウンターのタイプ。

1	累積カウンター、32 ビット（ACCUM）
2	ビジーカウンター（BUSY）
3	キューカウンター（QUEUE）
4	累積カウンター、64 ビット（FACCUM）
5	キュービジーカウンター（QBUSY）
6	ビジーカウンター、タイマーセルでインクリメント（TCELLBUSY）
7	キューカウンター、タイマーセルでインクリメント（TCELLQUEUE）
8	キュービジーカウンター、タイマーセルでインクリメント（TCELLQBUSY）

USERID

プロセスアクセサー ID（PAID）。このフィールドは、プロセスの所有者の識別に役立ちます。

USERDEF エンティティの ZMS 形式のみのカウンターフィールドの DDL 定義

このカウンターフィールドは、ZMS 形式の DDL レコードでのみ使用されます。

VALUE

（ZMS 形式のみ）ユーザー定義のカウンターの値。

USERDEF エンティティの Legacy 形式のみのカウンターフィールドの DDL 定義

これらのカウンターフィールドは、Legacy 形式の DDL レコードでのみ使用されます。

フィールドはアルファベット順に一覧しています。

COUNTS

（Legacy 形式のみ）ユーザー定義の累積カウンターの値です。

MAX-VALUE

（Legacy 形式のみ）TIME‑VALUE カウンターで記述されるキューの項目の最大数。

TIME-VALUE

（Legacy 形式のみ）再定義されました。COUNTS を参照してください。

すべての USERDEF エンティティの使用上の注意

- カウンターをアプリケーションで定義するには、適切な時点で MEASCOUNTERBUMPINIT および MEASCOUNTERBUMP プロシージャーを呼び出すようにソースコードを変更します。詳しくは、**MEASCOUNTERBUMP**(473 ページ)および **MEASCOUNTERBUMPINIT**(474 ページ)を参照してください。
- ユーザー定義カウンターから情報を収集するには、ADD USERDEF コマンドを使用して、ユーザー定義のカウンターを含むプロセスを指定し、ADD COUNTER コマンドを使用して、カウンターの名前を指定します。詳しくは、**ADD entity-type**(43 ページ)の ADD USERDEF と ADD COUNTER を参照してください。

L シリーズ USERDEF エンティティの使用上の注意

L シリーズの RVU では、タイマーセル USERDEF カウンタータイプは、このプラットフォームではサポートされていない機能であるタイマーセルの代わりに通常の TOD インターフェイスを使用します。タイマーセルカウンタータイプ（TCELLBUSY、TCELLQUEUE、および TCELLQBUSY）は、下位互換性を保つために引き続きサポートされますが、内部的にこれらは現在、BUSY、QUEUE、および QBUSY USERDEF のカウンターと同じになります。

MEASCOM INFO コマンドは、CONTAB 構成テーブル内にある入力カウンタータイプを表示します。ただし、MEASCOM LIST コマンドは、COUNTER レコード内にある内部的に使用されるカウンタータイプを表示します。たとえば、カウンターが TCELLBUSY のタイプである場合、MEASCOM INFO コマンドは TCELLBUSY カウンタータイプを表示しますが、MEASCOM LIST コマンドは BUSY カウンタータイプを表示します。

H シリーズ以降の USERDEF エンティティの使用上の注意

- H シリーズ以降の RVU に、複数の新しいカウンタータイプが定義されています。

- キュービジー。このカウンタータイプは、リソースのキューがビジー状態である時間、つまり項目がキューで待機していた時間を測定します。
- ビジー。タイマーセルにより実装されます。このカウンターでは、セマンティクスは従来のビジーカウンターと同じですが、タイマーセルメカニズムを使用して細分性の高いタイマーを提供します。
- キュー。タイマーセルにより実装されます。このカウンターでは、セマンティクスは従来のキューカウンターと同じですが、タイマーセルメカニズムを使用して細分性の高いタイマーを提供します。
- キュービジー。タイマーセルにより実装されます。このカウンターでは、セマンティクスは従来のキュービジーカウンターと同じですが、タイマーセルメカニズムを使用して細分性の高いタイマーを提供します。

- 新しいタイプのカウンターを定義する場合、Measure は暗黙的にタイマーセルを使用してカウンターを維持します。タイマーセルメカニズムの使用法について詳しくは、**Measure 呼び出し可能プロシージャー**(458 ページ)を参照してください。

G シリーズ USERDEF エンティティの使用上の注意

- Measure G07 以降の PVU では、USERDEF エンティティは 64 ビットのアキュムレータカウンタータイプです。この新しいカウンタータイプを追加するには、キーワード FACCUM とともに ADD COUNTER コマンドを使用します。
- Measure G11 以降の PVU：
  - カスタマーアプリケーションは、MEASCOUNTERBUMPINIT()プロシージャーの SUBSYSTEM-VERSION 拡張を使用する場合、ZMSUSER レコードに SUBSYSTEM-VERSION を提供できます。
  - USERDEF レポートでは、ACCUM カウンター（32 ビット）と FACCUM カウンター（64 ビット）を区別します。ZMSUSER レポートでは、すべてのカウンターが 64 ビットフィールドからフォーマットされ、タイプ ACCUM として報告されます。

Measure 呼び出し可能プロシージャー

この章では、アプリケーションプロセスが Measure のパフォーマンス監視を制御し測定データにアクセスするようにする、Measure 呼び出し可能プロシージャーについて説明します。

Measure プロシージャーについては、**Measure プロシージャーの概要**(460 ページ)を参照してください。

Measure プロシージャーの一覧

次の表では、Measure プロシージャーを機能別に一覧表示しています。

表 5: Measure 呼び出し可能プロシージャー

機能	プロシージャー	説明
データの収集	**MEASCLOSE**(468 ページ)	測定データファイルへのアクセスを削除します
	MEASCONFIGURE(469 ページ)	測定の構成を定義します
	MEASCONTROL(471 ページ)	測定を開始および停止します
	MEASOPEN(503 ページ)	測定データファイルにアクセスします
データの確認	**MEASGETVERSION**(478 ページ)	Measure 製品バージョンか、データファイルの外部エンティティレコード長、あるいはその両方を返します
	MEASINFO(481 ページ)	開かれていない測定データファイルから構成情報を返します
	MEASREAD(507 ページ)	測定データファイルからデータを読み込み、呼び出し元に結果を返します
	MEASREADACTIVE(515 ページ)	アクティブなカウンターレコードからデータを読み込みます
	MEAS_READACTIVE_(518 ページ)	アクティブなカウンターレコードからデータを読み込みます
	MEAS_READACTIVE_MANY_(521 ページ)	同一の呼び出しで複数のアクティブなカウンターレコードからデータを読み込みます
	MEASREADCONF(524 ページ)	測定データファイルから構成情報を読み取ります
	MEASREAD_DIFF_(509 ページ)	測定データファイルからデータを読み込み、呼び出し元に結果を返します
	MEASSTATUS(535 ページ)	アクティブな測定に関する情報を返します
	MEASWRITE_DIFF_(538 ページ)	測定データファイルからデータを読み込み、結果をファイルに格納します

表は続く

機能	プロシージャー	説明
ユーザー定義のカウンターのバンプ	**MEASCOUNTERBUMP**(473 ページ)	ユーザー定義のカウンターをバンプします
	MEASCOUNTERBUMPINIT(474 ページ)	ユーザー定義のカウンターのオフセットを返します
Measure サブシステムの制御	**MEASMONCONTROL**(499 ページ)	サブシステムを開始および停止します
	MEASMONSTATUS(501 ページ)	現在アクティブな測定の名前と数を返します
システム構成情報の取得	**MEASLISTCONFIG**(485 ページ)	MEASCTL プロセスによって提供されるシステム構成情報を返します
ファイル名の変換	**MEAS_CODERANGENAME_DEMANGLE_**(468 ページ)	プロシージャー名の内部（マングル）表現を、プログラマが指定した元の名前に戻します
	MEAS_GETDESCINFO_(475 ページ)	ANSI SQL 名を、その対応するエンティティ記述子コンポーネントに変換します
	MEASLISTENAME(487 ページ)	Guardian ファイル名または MID を、対応する外部形式の ANSI SQL 名または OSS パス名に変換します
	MEASLISTEXTNAMES(491 ページ)	Measure リスト構造化 OSS および ANSI SQL 名前情報を、EXTNAMES ファイルに書き込みます
	MEASLISTGNAME(492 ページ)	OSS ファイルパス名を Guardian ファイル名に変換し、エンティティ記述子構成の MID（PATHID および CRVSN）コンテンツを返し、OSS ディレクトリのコンテンツを返します
	MEASLISTOSSNAMES(496 ページ)	構造化 OSS ファイルパス名の変換情報を、ファイル OSSNAMES に返します
	MEASLISTPNAME(497 ページ)	Guardian ファイル名または OSS PathID をその OSS ファイルパス名の同等表現に変換します
ANSI SQL 名の使用	**MEAS_SQL_MAP_INIT_**(529 ページ)	SQL/MX マッピングセッションを開始します
	MEAS_SQL_MAP_STOP_(530 ページ)	SQL/MX マッピングセッションを停止します
	MEAS_SQLNAME_COMPARE_(531 ページ)	外部形式で 2 つの完全修飾 ANSI SQL 名を比較します
	MEAS_SQLNAME_RESOLVE_(532 ページ)	ANSI SQL 名の各部を組み合わせて、正規化された外部形式で完全修飾名を作成します

表は続く

機能	プロシージャー	説明
	MEAS_SQLNAME_SCAN_(534 ページ)	外部形式の完全修飾された（おそらくワイルドカードが使用された）ANSI SQL 名を解析します
レコード形式の調整	**MEAS_ADJUSTZMSRECORD_**(465 ページ)	ZMS 形式の構造体レコードを、アプリケーションをコンパイルしたときの MEASDDLS 形式に調整します
細分性の高いタイマーの保守	**MEAS_ALLOCATE_TIMERCELLS_**(466 ページ)	TCELLBUSY、TCELLQUEUE、および TCELLQBUSY タイマーを保守するためのタイマーセルを割り当てます
	MEAS_BUMP_TIMERCELL_(467 ページ)	TCELLBUSY、TCELLQUEUE、または TCELLQBUSY タイマーを設定、リセット、インクリメント、またはデクリメントします
	MEAS_DEALLOCATE_TIMERCELLS_(475 ページ)	タイマーセルの割り当てを解除します
	MEAS_RETRIEVE_TIMERCELLS_(528 ページ)	1 つまたは複数の TCELLBUSY、TCELLQUEUE、および TCELLQBUSY タイマーのタイマー値を取得します

Measure プロシージャーの概要

Measure レコード（DDL）での読み取り

Measure DDL レコードは PVU ごとに異なります。新しいカウンターは常に、既存の Measure DDL カウンターの末尾に追加され、既存のカウンターを使用しているアプリケーションが、新しいカウンターの追加の影響を受けないようにします。

いくつかの単純なコーディング方法が、今後の PVU との互換性の問題の回避に役立ちます。プログラムインターフェイスを通じてレコードを要求するユーザーアプリケーションは、（新しいフィールドが重要でない場合でも）より長い新しいレコードに対応できるだけ十分に大きなバッファーを使用する必要があります。Measure は部分的なレコードを返さないので、数千バイトのバッファーを使用するように作成されたアプリケーションでの DDL レコードの読み取り時に、互換性の問題を回避します。

Legacy 形式と ZMS 形式のレコード

H01 以降の PVU では、Measure データの内部表現は ZMS 形式のレコードで、今後の機能拡張は ZMS 形式でのみ行われます。ただし、Measure API によって返されるレコードのデフォルト形式は、依然として Legacy 形式です。

将来の拡張機能を利用してアプリケーションのパフォーマンスを改善するには、ZMS 形式を使用するように既存のアプリケーションを移植することをお勧めします。この効果を得るには、*templateversion* パラメーター（各種プロシージャーコール内に存在する）を使用します。

アプリケーションが引き続き Legacy 形式のレコードを要求する場合、Measure は内部表現から Legacy 形式にデータを変換します。この変換は、レガシーのカウンターフィールドにいくつかの方法で影響を及ぼします。

- レガシーレコードに同じカウンターの 2 つのバージョンがある場合、つまり 1 つが 32 ビットバージョンで 1 つが 64 ビットバージョン（通常は-F で終わる）の場合、64 ビットのフィールドには必ず値が含まれます。32 ビットフィールドに値が含まれるのは、ZMS 形式の内部レコードの値が 32 ビットに

収まる場合に限られます。内部の値が 32 ビットに収まらない場合、アプリケーションに返される 32 ビットのカウンターの値は-1 になります。
- MAX-QLEN カウンターは保持されなくなりました。これらのフィールドは、値 1 が Legacy 形式のレコードに返されます。
- Measure G12 PVU から ZMS 形式のレコードに追加されたフィールドは、Legacy 形式のレコードを要求するアプリケーションでは使用できません。

宣言ファイルでの読み取り

TAL コンパイラの?SOURCE コマンドを使用して、以下の宣言ファイルを、ソースコードのグローバル宣言に読み込みます。

- $SYSTEM.SYSTEM.EXTDECS0 には、Measure プロシージャーの外部宣言が含まれます。プログラムで使用する各 Measure プロシージャーは、?SOURCE コマンドで指定する必要があります。
- $SYSTEM.SYS*nn*.MEASDECS には、Measure 制御ブロック、構成テーブル、およびエンティティ記述子に対するテンプレート構造体が含まれます。また、プロシージャーから返されるエラーメッセージコードのリテラル識別子と、構成テーブルおよびエンティティ記述子で使用される定数のリテラル識別子を宣言します。

Measure 制御ブロックへの領域の割り当て

プログラムで MEASCLOSE、MEASOPEN、MEASREAD、および MEASREADCONF プロシージャーのみを使用する場合以外は、プログラムのグローバルデータ領域に Measure 制御ブロック用の領域を割り当てる必要があります。プロセスで行われる測定の数に関係なく、1 プロセスあたりに 1 つのみの Measure 制御ブロックが必要です。

MEASDECS ファイルの MEASCB^DEF は、制御ブロックのためのテンプレート構造体です。この参照構造体は、制御ブロックのための領域を割り当てます。

```
STRUCT .MEASCB(MEASCB^DEF)
```

Measure サブシステムは、この制御ブロックを使用してプロシージャーコールのデータを保存します。ブロックの内容は、プロシージャーに渡した後は変更しないでください。

制御ブロックを使用する最初のプロシージャーを呼び出す前に、ブロック内の各要素を-1 に初期化する必要があります。

```
$FILL16(MEASCB, $LEN(MEASCB)/2, -1);
```

エンティティ記述子の指定

コマンドインターフェイスで使用されるエンティティの指定と同様に、エンティティ記述子はエンティティまたはエンティティのセットを記述します。エンティティ記述子を使用して、測定するエンティティを定義したり、エンティティごとに維持するカウンターレコードを指定したりします。エンティティ記述子は、測定を開始する前に MEASCONFIGURE プロシージャーに渡す構成テーブルの一部です。

MEASDECS ファイルは、エンティティ記述子のテンプレート構造体を宣言します。たとえば、CPU^DESC テンプレートには、CPU エンティティを定義するために使用するフィールドが含まれます。各エンティティ記述子には次の要素が含まれています。

- *Type*。定義されているエンティティのタイプを識別する数値識別子またはリテラル。MEASDECS ファイルは、エンティティタイプごとにリテラルを宣言します。
- *Len*。記述子の長さ（バイト単位）。
- *Cpu^number*。エンティティが存在する CPU の番号 （一部の記述子は、このフィールドに別の名前を使用します）。

記述子の残りのフィールドは、測定対象の 1 つまたは複数のエンティティを識別します。

エンティティ記述子について詳しくは、**エンティティ記述子**(545 ページ)を参照してください。

エンティティのセットの測定

多くの記述子フィールドで、特定のタイプのすべてのエンティティを示すワイルドカード値を使用できます。次に例を示します。

- CPU エンティティ記述子で、*cpu^number* フィールドのリテラル-1 は、システム上のすべての CPU を指定します。
- PIN、チャネル、ctl、単位、またはシステム番号を表すフィールドで、リテラル-1 はすべてを指定します。
- 名前フィールドで、アスタリスク（*）はすべてを指定します。このようなフィールドは左寄せする必要があり、ブランクまたは null で埋める必要があります。左端の文字として任意の適切な先頭文字を含めます。たとえばノードの場合はバックスラッシュ（\）、ボリュームの場合はドル記号（$）、サブデバイスの場合はシャープ記号（#）です。
- アダプター名、config 名、SAC 名、仮想チャネル名、およびエンドポイント名が含まれている名前フィールドには（必要な場合に）null で埋める必要があります。他のすべての名前フィールドはブランクで埋める必要があります。
- すべてのシステムプロセスを指定するには、PIN に-2 を設定します。

ワイルドカードは常に有効というわけではありません。たとえば、Measure がアクティブなカウンターから読み取るとき、MEASREADACTIVE に渡すエンティティ記述子は単一エンティティを指定する必要があります。

セットからのエンティティの除外

測定するエンティティのセットを指定した後、個々のエンティティを測定から除外できます。エンティティを除外するには、そのエンティティの記述子を構成テーブルに追加し、*type* フィールドを負の値に設定します。*type* 文字列リテラルまたは数値識別子に負の値を指定できます。

文字列リテラルによるセットからのエンティティの除外

CPU 6 を除くすべての CPU を測定するには、以下のようにします。

手順

1. *type* が CPU^T に設定されて、*cpu^number* が ALL に設定されている CPU 記述子を追加します。
2. *type* が-CPU^T に設定され（CPU^T に対応する負リテラル）、*cpu^number* が 6 に設定された 2 番目の CPU 記述子を追加します。

数字識別子によるセットからのエンティティの除外

CPU 6 を除くすべての CPU を測定するには、以下のようにします。

手順

1. *type* が 1（CPU の数値識別子）に設定され、*cpu^number* が ALL（または同等の数値リテラル、-1）に設定された記述子を追加します。
2. *type* フィールドのビット 0 が 1 に設定された 2 番目の記述子を追加します。このビットは除外フラグを設定し、負の値を設定するのと同じです。
3. *type* フィールドのビット 1 からビット 15 までを 1 に設定します（CPU の数値タイプ識別子）。
4. *cpu^number* フィールドを 6 に設定します。

ファイルおよびデバイス名の指定

すべてのディスクファイル、システム、デバイス、およびプロセス名は、ローカル内部名形式で指定する必要があります。内部ファイル名形式の説明については、*Guardian Procedure Calls Reference Manual* を参照してください。名前フィールドまたはサブフィールドでアスタリスクを使用する場合、適切な先頭文字（\、$、または#）を含め、フィールドを空白で埋める必要があります。

ANSI SQL 名の指定

Measure G11 および H01 の PVU では、ANSI SQL 名をサポートするために、一部のエンティティ記述子に追加フィールドが設けられました。

ANSI SQL 名をプロシージャーに引数として渡すときには、一重引用符で囲まないでください。

詳しくは、**ANSI SQL 名の処理**(183 ページ)を参照してください。

構成テーブルの作成

構成テーブル（*contab*）は、測定するエンティティを定義します。測定を開始する前に、構成テーブルを MEASCONFIGURE に渡します。測定を構成した後、MEASREADCONF または MEASINFO を呼び出すことによって構成テーブルを読み取ることができます。構成テーブルは次のもので構成されます。

- ヘッダーレコード
- エンティティ記述子セクション
- トレーラーレコード

ヘッダーレコードとトレーラーレコードは両方とも固定長です。エンティティ記述子セクションの長さは可変のため、テーブル全体としては可変長です。

ヘッダーレコード

テンプレート構造 CONTAB^HDR を使用して、以下の情報を含むヘッダーレコードを定義します。

変数	説明
Type	数値 50 またはリテラル CONTAB^T（MEASDECS ファイルで宣言される）。このフィールドは、このレコードをヘッダーレコードとして識別します。
Len	構成テーブル全体の長さ（バイト単位）。
Sections	テーブル内の各エンティティタイプの記述子セクションを指す、バイト単位のオフセットの配列。配列の最初の単語は常にゼロです。以下のオフセットはエンティティタイプ番号順に表示され、構成テーブルの先頭に対してインデックスが作成されます。（エンティティタイプ番号の説明については、**エンティティ記述子**(545 ページ)を参照してください。）エンティティタイプが測定されていない場合、オフセットはゼロです。
Maxents	ヘッダーフォーマット。*Maxents* はセクション配列の最初の要素です（つまり、SECTIONS[0]）。

注記:

このヘッダーフォーマットを使用していない古いカスタムプログラムがある場合は、これらのプログラムを再コンパイルします。

エンティティ記述子セクション

構成テーブルの本体はエンティティ記述子の配列です。エンティティ記述子は、エンティティタイプに従ってセクションに分類され、セクションは数値識別子によって順序付けられます。最初にすべての CPU 記述子（タイプ番号 1）が置かれ、次にすべての PROCESS 記述子（タイプ番号 2）、その次にすべて PROCESSH 記述子（タイプ番号 3）、以下同様となります。

エンティティタイプが測定対象でない場合、そのタイプのセクションは含めないでください。

DELETE 記述子は ADD 記述子より優先されます。ADD 記述子の *contab* に一致が見つかった場合、戻り値が暫定的に true に設定されます。ただし、その後で DELETE 記述子で一致が見つかった場合は、すぐに false の戻り値が返されます。

トレーラーレコード

トレーラーレコードを定義するには、テンプレート構造体 CONTAB^TRAILER を使用します。内容は、次のとおりです。

変数	説明
Type	数値 51 または文字列リテラル CONTAB^TRAILER^T（MEASDECS ファイルで宣言される）。このフィールドは、このレコードをトレーラーレコードとして識別します。
Len	トレーラーレコードの長さ（バイト単位）。

テンプレート構造体 CONTAB^TRAILER を使用して、トレーラーレコードを定義し、リテラル CONTAB^TRAILER^T を *type* フィールドに割り当てます。

Measure プロシージャーのエラーコードの解釈

Measure プロシージャーは、エラーワードを返す整数関数プロシージャーです。エラーが発生しなかった場合、プロシージャーはエラーワードに 0（ゼロ）を返します。エラーが発生した場合、プロシージャーはエラーコードを返します。

Measure プロシージャーに固有のエラーは 3200 から 3499 までの間のエラーコードを持ち、**エラーコード**(627 ページ)で説明されています。MEASDECS ファイルには、エラーコードのリテラル識別子が宣言されています。

Measure プロシージャーは、実行するプログラムデータスタック上のスペースを使用します。（Measure プロシージャーによって使用される最大スペースは約 3 ページです。）Measure プロシージャーの実行中に無効なアドレス参照トラップ（トラップ 0）またはスタックオーバーフロートラップ（トラップ 3）を受け取った場合、TAL コンパイラの?DATAPAGES コマンドまたは RUN コマンドの MEM パラメーターを使用して、データページの割り当てを増やしてみてください。

MEASDDLS と MEASCHMA での新しい構造との互換性の維持

いくつかの状況では、C および C++アプリケーションの再コンパイルが必要です。たとえば、G08 Measure は SERVERNET エンティティ記述子の MODULE フィールドと SUBDEVICE フィールドを、ServerNet クラスターの NODE 番号とリモートシステムのプロセッサーの CPU 番号として再定義します。C または C++アプリケーションでは、MEASDDLS と MEASCHMA によって提供される新しい構造を使用する ServerNet 測定を設定するための Measure 呼び出し可能プロシージャーを使用する場合、再コンパイルが必要です。これらの再定義されたフィールドは、ユニオンのメンバーとしてアクセスしなければなりません。

タイマーセルの使用

H シリーズの Measure ではタイマーセル機能が導入されています。この機能は、同じプロセッサー内の 2 つのプロセス間でのディスパッチまたは単純なメッセージ交換のコンテキスト内で完了するイベントのタイマー計算の精度を改善します。このメカニズムを使用すれば、すべてのタイマー更新時にプロセッサー同期の解除を招くことなく細分性の高いタイマーを維持できます。タイマーセル値が同期されるのは、間隔を置いてコピーされるときや、要求側のアプリケーションに報告されるときだけです。

タイマーセルのメカニズムに関連付けられているカウンタータイプは、次のとおりです。

- TCELLBUSY。このカウンタータイプは、従来の BUSY カウンターと同等ですが、タイマーセルで実装されます。
- TCELLQUEUE。このカウンタータイプは、従来の QUEUE カウンターと同等ですが、タイマーセルで実装されます。
- TCELLQBUSY。このカウンタータイプは、従来の QBUSY カウンターと同等ですが、タイマーセルで実装されます。

USERDEF カウンターの場合、新しいタイプのカウンターを定義することによってタイマーセルのメカニズムを活用できます。たとえば、アプリケーションを G シリーズの RVU で実行しており、次にそれを H シリーズまたは J シリーズの RVU で実行する予定の場合、測定するイベントが適切な間隔を置いて行われるときには、ビジー、キュー、およびキュービジー状態のタイマーを再定義することを検討してください。このような場合、プロセッサー同期の解除が与える影響はほとんどない、またはまったくないため、プロセッサー間通信または I/O を含むイベントは再定義する必要はありません。

タイマーセルを使用する USERDEF カウンターの再定義では、アプリケーションの変更は必要ありません。新しい呼び出し可能なプロシージャーはタイマーセルメカニズムをサポートしますが、Measure はこれらのプロシージャーを暗黙的に呼び出して、USERDEF エンティティに対して定義するタイマーセルカウンターを維持します。USERDEF エンティティの構成でのみ、変更が必要です。ここでは、カウンターがいずれかの新しいタイプのものであることを指定します。

アプリケーションが Measure の外部でカウンターを維持する場合にのみ、タイマーセル呼び出しを明示的に使用します。詳しくは、MEAS_ALLOCATE_TIMERCELLS_、MEAS_DEALLOCATE_TIMERCELLS_、MEAS_BUMP_TIMERCELL_、および MEAS_RETRIEVE_TIMERCELLS_の説明を参照してください。

L シリーズの RVU ではタイマーセル機能は無効で、**ビジーカウンター**(170 ページ)、**キューカウンター**(172 ページ)、および**キュービジーカウンター**(172 ページ)に置き換えられています。

MEAS_ADJUSTZMSRECORD_

ZMS 形式の構造レコードをいずれかの公開されている形式で取得し、アプリケーションがコンパイルされる MEASDDLS 形式にそれらを調整します。

```
error := MEAS_ADJUSTZMSRECORD_ ( in^record              ! i
                               ,out^record              ! o
                               ,templateversion );      ! i
```

in^record

入力

INT .EXT：参照：*n*

目的の新しい形式にパディングまたは短縮される ZMS 形式の外部レコードを含むバッファーです。*in^record* の長さは、*in^record* によって指定されるヘッダーレコードのテンプレートバージョンにエンコードされた情報から取得されます。

out^record

出力

INT .EXT：参照：*n*

templateversion パラメーターによって指定された形式にパディングまたは短縮される ZMS 形式の外部レコードを返すバッファーです。*out^record* の長さは *templateversion* パラメーターにエンコードされた情報から取得されます。

templateversion

入力

FIXED：値

要求されているレコードの形式のレコードテンプレートバージョンです。これは、リクエストするアプリケーションがコンパイルされたときに使用した MEASDDLS から取得されます。

-1F として渡された場合、H06.15/J06.04 RVU 以降では、現在のリリースの *templateversion* が使用されます。そのような場合、返される外部レコードは、アプリケーションがコンパイルされたときに使用したカウンターレコード定義と一致しないことがあります。

使用上の注意

さらに小さいまたは以前のレコード形式に合わせて調整する場合、2 つのレコードバージョン間で追加または増加されたフィールドは除去または切り捨てられます。

さらに大きいまたは新しいレコード形式に合わせて調整する場合、さらに最新の形式での新しいまたは大きなフィールドの位置がパディングされます。

すべてのパディングでは 0 の値を使用しますが、識別子セクションの最後にある長い可変長の識別子は除きます。これらによってサイズが増加する場合は、スペースでパディングされます。

アプリケーションがコンパイルされたときに使用されたもの以外のテンプレートバージョンのレコードを含む構造化データファイル（ファイルコード 170）を読み取らせるには、次の手順に従います。

手順

1. ファイル構造化データファイルを開きます。
2. FILE_GETINFO_を使用して、ファイル内のデータのレコード長を取得します。
3. レコードをすべて保持できる十分な長さのバッファーにファイルの各レコードを読み込みます。
4. MEASDDLS テンプレートに基づいて、再フォーマットされたレコードのバッファー領域を宣言します。
5. ファイルからのレコード、再フォーマットされたレコードの宣言されたバッファー領域、MEASDDLS からの正しい zms *entity-template-version* リテラルを渡す MEAS_ADJUSTZMSRECORD_を呼び出します。

MEAS_ALLOCATE_TIMERCELLS_

指定した数のタイマーセルを割り当て、それらのタイマーにインデックスの配列を返します。このプロシージャーは、Measure の外部のカウンターを維持するためにアプリケーションで細分性の高いタイマーを必要とする場合に使用します。Measure は、USERDEF エンティティのタイマーセルカウンターを定義するときにこのプロシージャーを暗黙的に呼び出します。

L シリーズ RVU ではタイマーセル機能は無効です。L シリーズで呼び出されると、この API は ERR^NOT^SUPPORTED エラーを返します。この機能は、**ビジーカウンター**(170 ページ)、**キューカウンター**(172 ページ)、および**キュービジーカウンター**(172 ページ)に置き換えられています。

```
error := MEAS_ALLOCATE_TIMERCELLS_ ( count          ! i
                                   ,types          ! i
                                   ,indexes);      ! i,o
```

count

入力

INT：値

割り当てるタイマーセルの数です。維持しようとするカウンターごとに 1 つのタイマーセルを割り当てます。プロセスごとのタイマーセルの最大数は 10,000 です。

types

入力

INT .EXT：参照：*count*

カウンタータイプの配列です。配列サイズは *count* によって指定されます。許容されるカウンタータイプは、TCELLBUSY (6)、TCELLQUEUE (7)、および TCELLQBUSY (8)です。この配列は、項目ごとにインデックスの配列にマップします。

indexes

入出力

INT .EXT：参照」*count*

タイマーセルへのインデックスの配列です。配列サイズは *count* によって指定されます。入力時は配列は空です。出力時には、後続の呼び出しで使用するインデックスが含まれます。この配列は、項目ごとにタイプの配列にマップします。

MEAS_BUMP_TIMERCELL_

タイマーセル内の値を増加、減少、設定、またはリセットすることによってカウンターの値を修正します。このプロシージャーは、Measure の外部のカウンターを維持するためにアプリケーションで細分性の高いタイマーを必要とする場合に使用します。Measure は、USERDEF エンティティのタイマーセルカウンターを定義するときにこのプロシージャーを暗黙的に呼び出します。

L シリーズ RVU ではタイマーセル機能は無効です。L シリーズで呼び出されると、この API は ERR^NOT^SUPPORTED エラーを返します。この機能は、**ビジーカウンター**(170 ページ)、**キューカウンター**(172 ページ)、および**キュービジーカウンター**(172 ページ)に置き換えられています。

```
error := MEAS_BUMP_TIMERCELL_ ( index            ! i
                              ,op                ! i
                              ,[ count ] );      ! i
```

index

入力

INT：値

タイマーセルのインデックスです。

op

入力

INT：値

タイマーセル上で実行される操作を示します。指定可能な値は、以下のとおりです。

- TCELLBUSY カウンターの値を設定するには SETBUSY (3)を指定します。この値は、カウンターによって測定されるリソースの使用を開始するときに使用します。
- TCELLBUSY カウンターの値をリセットするには RESETBUSY (4)を指定します。この値は、カウンターによって測定されるリソースの使用を停止するときに使用します。
- TCELLQUEUE カウンターの値を増やすには INCQUEUE (5)を指定します。この値は、項目をキューに配置する場合に使用します。
- TCELLQUEUE カウンターの値を減らすには DECQUEUE (6)を指定します。この値は、項目をキューから除去する場合に使用します。
- TCELLQBUSY カウンターの値を増やすには INCQBUSY (7)を指定します。この値は、キューが空になった後でキューに最初の項目を格納する場合に使用します。
- TCELLQBUSY カウンターの値を減らすには DECQBUSY (8)を指定します。この値は、キューから最後の項目を除去してキューを空にする場合に使用します。

count

入力

INT：値

同じカウンターを同じ呼び出しで複数回増減するために使用するオプションのパラメーターです。SETBUSY および RESETBUSY の場合、有効な値は 1 のみです。

MEASCLOSE

測定データファイルを閉じ、データファイルにアクセスするために Measure サブシステムが作成したファイル処理プロセス（MEASFH）を停止します。測定データファイルを開くには、MEASOPEN プロシージャーを使用します。

```
error := MEASCLOSE ( dfnum );      ! i
```

dfnum

入力

INT：値

データファイルアクセス番号です。MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。

MEAS_CODERANGENAME_DEMANGLE_

プログラマによって提供された元のプロシージャー名を返します。

```
error := MEAS_CODERANGENAME_DEMANGLE_ ( InputName            ! i
                                      ,InputNameLen          ! i
                                      ,OutputName            ! o
                                      ,OutputNameMaxLen      ! i
                                      ,OutputNameLen         ! o
                                      ,NameType );           ! o
```

InputName

入力

STRING .EXT：参照：*

変換される名前文字列を含む最大 1024 バイトのバッファーです。終端ヌルは予期されません。

InputNameLen

入力

INT：値

入力される名前文字列内のバイト数です。

OutputName

出力

STRING .EXT：参照：*

変換される名前を含む最大 1024 バイトのバッファーです。入力される名前が C++マングル名でない場合は、入力文字列が変更されずに出力バッファーにコピーされます。

OutputNameMaxLen

入力

INT：値

入力バッファーのサイズです。

OutputNameLen

出力

INT .EXT：参照：*

出力バッファー内のバイト数です。

NameType

出力

INT .EXT：参照：1

C++以外の名前の場合は 1、C++の名前の場合は 2 です。

MEASCONFIGURE

測定を設定します。

測定アプリケーションは MEASCONFIGURE を呼び出す前に、（Measure データファイルを初期化するための）MEASOPEN プロシージャーを呼び出す必要があります。

L シリーズ RVU では、このプロシージャーは FABRIC 記述子をサポートし、ENDPOINT エンティティ記述子の GMS、ポート、およびファイバーフィールドをサポートしなくなりました。

```
error := MEASCONFIGURE ( meascb                ! i,o
                        ,dfnum                 ! i
                        ,measnum               ! o
                        ,contab                ! i
                        ,[ errDetail ] );      ! o
```

meascb

入出力

INT：参照：$LEN (MEASCB^DEF) / 2

Measure サブシステムが後のプロシージャーコールのためにデータを格納する制御ブロックです。*meascb* を使用する最初の Measure プロシージャーを呼び出す前に、グローバルデータ領域内に制御ブロック用のスペースを割り当て、制御ブロックの各要素を-1 に初期化する必要があります。*meascb* を Measure プロシージャーに渡した後は、内容を変更しないでください。

MEASDECS ファイルには、制御ブロック記述子の構造定義（MEASCB^DEF）が含まれています。

dfnum

入力

INT：値

データファイルアクセス番号です。MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。

measnum

出力

INT：参照：1

測定番号です。後続のプロシージャーコールでこの番号を使用して、測定を識別します。

contab

入力

INT .EXT：参照：*

測定の構成を定義する配列です。*contab* 配列には、ヘッダーセクション、エンティティタイプごとのエンティティ記述子セクション、およびトレーラーセクションが含まれています。

```
CONTAB^HDR
  ...
  maxents
```

```
entitydesc_1
[ entitydesc_2
   ...
  entitydesc_n ]

CONTAB^TRAILER
  ...
  len
```

contab 配列で使用される構造定義は MEASDECS ファイルで定義されます。MEASDECS は、各エンティティタイプおよびヘッダーとトレーラーのレコードについてリテラル識別子および数値識別子の両方を宣言します。

CONTAB^HDR

contab 配列の内容を記述します。

```
Struct Contab^hdr (*)
  Begin
  Int    Type;                    ! Set to CONTAB^T
  Int    Len;                     ! Byte length of entire contab
  Int    Sections[0:MAX^T];       ! Byte offset from Contab^hdr.Type to
                                  ! start of descriptor section
  Int    Maxents = Sections;      ! Maximum entity types known to the
                                  ! user; this is used to identify the
                                  ! version of the contab.
  Filler 8;                       ! Reserved for future use
  End;
```

type

INT：値：1

contab 配列のヘッダーセクションを識別します。*type* は、リテラルの CONTAB^T または数値識別子の 50 に指定します。

len

INT：値：1

contab 配列全体の長さ（バイト単位）です。最大長は、ヘッダーとトレーラーを含めて 32,000 バイトです。

sections

INT：値：MAX^T+1

contab 配列の各セクションへのバイトオフセットの配列です。配列要素は**エンティティおよびカウンター**に示すようにエンティティタイプ順です。

最初のバイトオフセット（配列要素 0）を 0 として指定します。*contab* 配列のセクションを省略する場合、そのセクションのバイトオフセットを 0 として指定します。

maxents

INT：値：1

ヘッダーフォーマットを識別します。このフィールドは SECTIONS[0]と同等であるため、CONTAB^HDR の長さに追加されません。*maxents* フィールドは、ユーザーが入力する必要はありません。このフィールドは、ユーザーが提供する記述子に基づいて、MEASCONFIGURE が入力します。

***entitydesc_1* [*entitydesc_2 ... entitydesc_n*]**

測定のエンティティを定義します。**エンティティ記述子**(545 ページ)に記載されているエンティティ記述子の 1 つとして、それぞれの *entitydesc* を指定します。

Measure H01 以降の PVU では、MEASCONFIGURE は ANSI SQL オブジェクトまたはパーティションの DISCOPEN、DISKFILE、FILE、または SQLSTMT エンティティ記述子を受け入れます。

Measure H07 以降の PVU では、MEASCONFIGURE は FABRIC エンティティ記述子を受け入れます。

CONTAB^TRAILER

contab 配列の終了のマークを付けます。

```
Struct Contab^trailer (*)
  Begin
  Int    Type;      ! Set to CONTAB^TRAILER^T
  Int    Len;       ! Byte length of this record
  End;
```

type

INT：参照：1

contab 配列のトレーラーセクションを識別します。*type* は、リテラルの CONTAB^TRAILER^T または数値識別子の 51 に指定します。

len

INT：値：1

CONTAB^TRAILER 記述子のバイト単位の長さ（通常は 4）。

errDetail

出力

FIXED .EXT：参照：1

contab バッファー内で最初に発生したエラーに関する情報を格納します。

`errDetail.32:47`	`> 0 =`	エンティティタイプ内の不正な記述子の序数で、1 から始まります。
`errDetail.48:63`	`-1 =`	この引数は、MEASFH の使用されているバージョンでサポートされていません
	`0 =`	contab に何らかの異常がありますが、必ずしもいずれかの記述子にあるわけではありません。
	`> 0 =`	エンティティタイプ。

errDetail パラメーターは、Measure H03-AFV/J01-AFW 以降のリリースでのみ使用できます。このパラメーターは、エラー `ERR^BADDESC` が返された場合のみ有効です。

関連プロシージャー

計測を開始するには、MEASCONTROL プロシージャーを使用します。

MEASCONTROL

測定を起動および停止し、オプションとして測定間隔を含めます。

MEASCONTROL を呼び出す前に、プログラムは MEASCONFIGURE を呼び出して測定を構成する必要があります。

```
error := MEASCONTROL ( meascb                ! i,o
                      ,measnum               ! i
                      ,[ starttime ]         ! i
                      ,[ stoptime  ]         ! i
                      ,[ interval ] );       ! i
```

meascb

入出力

INT：参照：$LEN(MEASCB^DEF) / 2

Measure サブシステムが後のプロシージャーコールのためにデータを格納する制御ブロックです。*meascb* を使用する最初の Measure プロシージャーを呼び出す前に、グローバルデータ領域内に制御ブロック用のスペースを割り当て、制御ブロックの各要素を-1 に初期化する必要があります。*meascb* を Measure プロシージャーに渡した後は、内容を変更しないでください。

$SYSTEM.SYS*nn*.MEASDECS ファイルには、制御ブロック記述子の構造定義（MEASCB^DEF）が含まれています。

measnum

入力

INT：値

測定番号です。MEASCONFIGURE プロシージャーによって返される *measnum* 値を使用します。

starttime

入力

FIXED：値

測定の開始時間です。省略するか、-1 として指定すると、測定は構成されますが開始されません。*starttime* および *stoptime* を指定した場合、*starttime* は *stoptime* より小さくする必要があります。*starttime* は、システムプロシージャー JULIANTIMESTAMP によって提供された 4 ワード/1 マイクロ秒形式の地方平均時（LCT）に基づくユリウス日として指定します。*Guardian プロシージャーコールリファレンスマニュアル*を参照してください。

stoptime

入力

FIXED：値

測定の停止時間です。省略するか、-1 として指定すると、後続の呼び出しで停止時間が入力されるまで測定が実行されます。*starttime* および *stoptime* を指定した場合、*starttime* は *stoptime* より小さくする必要があります。*stoptime* は、システムプロシージャー JULIANTIMESTAMP によって提供された 4 ワード/1 マイクロ秒形式の LCT に基づくユリウス日として指定します。*Guardian プロシージャーコールリファレンスマニュアル*を参照してください。

interval

入力

FIXED：値

収集間隔です。カウンターの値は、一時的なエンティティの起動時、または一時的なエンティティの停止時に指定された時間間隔でデータファイルに書き込まれます。*interval* を省略した場合、カウンターの値は、測定の開始時に 1 回、終了時に 1 回の 2 回データファイルに書き込まれます。*interval* はマイクロ秒で指定します。測定の開始後は収集間隔を変更できません。

MEASCOUNTERBUMP

以下の場合、*offset* によって識別されるカウンターを変更します。

- カウンターが現在アクティブな測定の一部である。カウンターが現在測定されていない場合、MEASCOUNTERBUMP は ERR^UDCNOTPRESENT を返します。
- カウンタータイプが以下のものと互換性を持つ。

  bumptype

  カウンターを測定の構成に追加するとき、カウンタータイプを指定します。

ユーザー定義のカウンターについて詳しくは、**MEASCOUNTERBUMPINIT**(474 ページ)を参照してください。

```
error := MEASCOUNTERBUMP ( offset              ! i
                         ,bumptype             ! i
                         ,[ addvalue ]         ! i
                         ,[ index ]            ! i
                         ,[ doubleadd ]        ! i
                         ,[ quadadd ] );       ! i
```

offset

入力

INT：値

内部カウンターレコード内のカウンター値のオフセットです。MEASCOUNTERBUMPINIT プロシージャーによって返される *offset* 値を使用します。

bumptype

入力

INT：値

カウンターバンプアクションを示すリテラル値です。*bumptype* はカウンタータイプと互換性を持つ必要があります。

値	アクション	カウンタータイプ
INC	カウンターをインクリメントします	累積
ADD	カウンターに *addvalue* を追加します	累積
SETBUSY	カウンターをビジーに設定します	ビジーまたは TCELLBUSY
RESETBUSY	カウンターを非ビジーに設定します	ビジーまたは TCELLBUSY
INCQUEUE	（キュー）カウンターを増やします	キューまたは TCELLQUEUE
DECQUEUE	（キュー）カウンターを減らします	キューまたは TCELLQUEUE
INCQBUSY	（キュービジー）カウンターを増やします	キュービジー（QBUSY または TCELLQBUSY）
DECQBUSY	（キュービジー）カウンターを減らします	キュービジー（QBUSY または TCELLQBUSY）

addvalue

入力

INT：値

カウンターに追加する値（正または負）です。*bumptype* を ADD に指定し、カウンタータイプが ACCUM または FACCUM である場合にのみ適用されます。これは、*doubleadd* パラメーターも指定されている場合は指定できません。

index

入力

INT：値

各カウンターの配列のインデックスです。最大のインデックス値は 127 です。デフォルトでは、カウンター名は 1 つのカウンター値（インデックス値 0）に関連付けられています。また、カウンター名をカウンター値の配列に関連付けることができます。

注記:

H07、J05、L シリーズ以降の PVU についてのインデックスの最大値は 511 です。

doubleadd

入力

INT(32)：値

カウンターに追加する値（正または負）です。*bumptype* を ADD に指定し、カウンタータイプが FACCUM である場合にのみ適用されます。*doubleadd* は、*addvalue* も指定されている場合は指定できません。

quadadd

入力

FIXED：値

64 ビットカウンターに追加する値（正または負）です。*bumptype* を ADD に指定し、カウンタータイプが FACCUM である場合にのみ適用されます。*quadadd* は、*addvalue* も指定されている場合は指定できません。*quadadd* パラメーターは Measure H01 以降の PVU でのみサポートされます。

MEASCOUNTERBUMPINIT

ユーザー定義のカウンター *name* が現在アクティブな測定の一部かどうかを判別します。そうである場合、MEASCOUNTERBUMPINIT はバンプされるカウンター値の場所を *offset* に返します。そうでない場合、MEASCOUNTERBUMPINIT は *offset* に-1 を返し、*error* に ERR^UDCNOTPRESENT を返します。

アプリケーションでカウンターを定義するには、ソースコードを変更して、MEASCOUNTERBUMPINIT および MEASCOUNTERBUMP（カウンターをバンプする）プロシージャーを適切な時点で呼び出します。ユーザー定義カウンターから情報を収集するには、以下のようにします。

手順

1. MEASCOM コマンド ADD USERDEF または MEASCONFIGURE プロシージャーを使用して、カウンターを変更するプロセスを構成に追加します。
2. MEASCOM コマンド ADD COUNTER または MEASCONFIGURE プロシージャーを使用して、カウンターを構成に追加します。

```
error := MEASCOUNTERBUMPINIT ( name                ! i
                              ,offset              ! o
                              ,[ version ] );      ! i
```

name

入力

INT：参照：8

ユーザー定義カウンター名を含む配列です。名前は 1～16 文字の英数字、ハイフン、またはアンダースコアである必要があり、名前の先頭は文字である必要があります。カウンター名は、大文字と小文字が区別されません。それらは大文字で表示されます。

offset

出力

INT：参照：1

内部カウンターレコード内のカウンター値のオフセットです。指定されたカウンターがアクティブな測定の一部でない場合、MEASCOUNTERBUMPINIT は *offset* として-1 を返します。

version

入力

INT：値

（Measure G11 以降の PVU）Measure データの SUBSYSTEM-VERSION として報告される増分バージョンです。Hewlett Packard Enterprise では、バージョンは 1 から始まり、アプリケーションの USERDEF インストルメンテーションロジックが変更されるたびに増やすようにすることをお勧めします。指定しない場合、SUBSYSTEM-VERSION は 0 として報告されます。

MEAS_DEALLOCATE_TIMERCELLS_

アプリケーションによって以前割り当てられていたタイマーセルの割り当てを解除します。このプロシージャーは、Measure の外部のカウンターを維持するためにアプリケーションで細分性の高いタイマーを必要とする場合に使用します。Measure は、USERDEF エンティティに定義されるタイマーセルカウンターに対してこのプロシージャーを暗黙的に呼び出します。Measure は、プロシージャーを呼び出さずにプログラムを終了する場合も、このプロシージャーを呼び出します。

L シリーズ RVU ではタイマーセル機能は無効です。L シリーズで呼び出されると、この API は ERR^NOT^SUPPORTED エラーを返します。この機能は、**ビジーカウンター**(170 ページ)、**キューカウンター**(172 ページ)、および**キュービジーカウンター**(172 ページ)に置き換えられています。

```
error := MEAS_DEALLOCATE_TIMERCELLS_ ( count          ! i
                                     ,indexes );     ! i,o
```

count

入力

INT：値

割り当て解除されるタイマーセルの数です。

indexes

入出力

INT .EXT：参照：*count*

割り当てを解除するタイマーセルのインデックスの配列です。出力では、セルの割り当てが解除される場合は値-1 が配列に含まれます。

MEAS_GETDESCINFO_

Measure H01 以降の PVU では、MEAS_GETDESCINFO_は、ワイルドカードで表される可能性がある完全修飾された ANSI SQL 名を、それに対応するエンティティ記述子コンポーネントに変換します。

MEAS_GETDESCINFO_は、エンティティ記述子構造のための MID（PATHID および CRVSN）と UID の内容を提供します。

```
error := MEAS_GETDESCINFO_ ( [ dfnum ]                  ! i
                           ,[ name ]                    ! i
                           ,[ name_len ]                ! i
                           ,pathid                      ! o
                           ,crvsn                       ! o
                           ,uid                         ! o
                           ,[ index ]                   ! i,o
                           ,[ connectionInfo ] );       ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3238	err^unknownsqlname	指定された ANSI SQL 名を変換できませんでした。
3239	err^badformatsqlname	指定された ANSI SQL 名に構文エラーが含まれていたか、完全修飾されていませんでした。
3241	err^missing^sqljournal	SQL/MX ジャーナルセグメントが必要でしたが、存在しませんでした。
3295	err^sql^api^internal	指定された ANSI SQL 名を構文解析できませんでした。
3296	err^sqlmx^map^process	*connectionInfo* が渡されなかったか、おそらく MEAS_SQL_MAP_INIT への呼び出しによって入力されなかったために *connectionInfo* の内容が正しくありませんでした。

dfnum

入力

INT：値

データファイルのアクセス番号か、デフォルト値でもある-1 です。ジャーナルセグメントを含むオープン状態の測定データファイルのジャーナルセグメントにアクセスするには、MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。-1 を指定した場合、または指定されたデータファイルからジャーナルセグメントを利用できないが、ファイルが現在のシステムにある場合、SQL/MX サブシステムから解決策が得られます。アクティブな測定について記述子を使用する場合、-1 を渡す必要があります。

name

入力

STRING .EXT：参照：*

変換される外部形式の、ワイルドカードで表される可能性がある完全修飾された ANSI SQL オブジェクト名または ANSI SQL パーティション名を含むバッファーです。名前が渡されない場合、出力フィールドにワイルドカードインジケーターが入力されます。

name_len

入力

INT：値

入力される名前のサイズ（バイト単位）です。*name* が入力される場合は存在する必要があります。

pathid

出力

INT .EXT：参照：12

パーティション名が渡された場合は MID の *pathid* コンポーネントです。それ以外の場合、ワイルドカード *pathid* 値が返されます。この値はエンティティ記述子の File^name^MID.Pathid フィールドに対応します。

crvsn

出力

INT .EXT：参照：3

パーティション名が渡された場合は MID の CRVSN コンポーネントです。それ以外の場合、ワイルドカード CRVSN 値が返されます。この値はエンティティ記述子の File^name^MID.crvsn フィールドに対応します。

uid

出力

INT .EXT：参照：12

オブジェクト名の UID コンポーネント（カタログ、スキーマ、オブジェクト）です。それ以外の場合、パーティション名のワイルドカード値が返されます。この値はエンティティ記述子の SQLMX^Obj^Desc フィールドに対応します。

index

入出力

FIXED .EXT：参照：1

反復呼び出しのコンテキスト値です。MEAS_GETDESCINFO_の最初の呼び出しで、*index* を-1F に設定する必要があります。オブジェクトのインスタンスがこれ以上見つからない場合、*index* は-1F として返されます。渡されない場合、-1F だとみなされます。

connectionInfo

入力

INT(32) .EXT：参照：25

このフィールドは MEAS_SQL_MAP_INIT_への呼び出しによって入力され、呼び出し元によって少なくとも 100 バイトになるように割り当てられます。この値は、SQL ジャーナルが使用されない場合にのみ必要です。

使用上の注意

- SQL ジャーナルと組み合わせて使用した場合、各 ANSI SQL 名は、同じオブジェクトの複数のインスタンスに対応できます。
- 0 の *name_len* でこの API が呼び出された場合、*pathid*、*crvsn*、および *uid* のフィールドはワイルドカード値で入力されます。

例

これは、MEAS_SQLNAME_RESOLVE_と MEAS_GETDESCINFO_を使用する方法を示す、pTAL 疑似コードの簡単な例です。

```
STRING .EXT defcat[0:MAXCATLEN]  :=["cat_12"];
STRING .EXT inbuf[0:MAXANSINAME] :=["table sch.t"];
STRING .EXT namebuf[0:MAXANSINAME];
INT         namelen;
INT         inlen;
FIXED       index;
INT    .EXT dfile^desc(diskfile^ANSI^desc);

! resolve defaults and syntax check
! calculate inlen and catlen
if (error := MEAS_SQLNAME_RESOLVE_ (namebuf,
                                    MAXANSINAME + 1,
                                    namelen,
                                    inbuf,
                                    inlen,
                                    defcat,
                                    catlen ))
  ! handle error

! we have a syntactically correct fully qualified name
! initialize non SQL fields of the descriptor
index := -1f;
do
begin
  if (error := MEAS_GETDESCINFO_ (dfnum,
                                  namebuf,
                                  namelen,
                                  dfile^desc.File^name^MID.pathid,
                                  dfile^desc.File^name^MID.crvsn,
                                  dfile^desc.SQLMX^Obj^Desc,
                                  index )) then
    ! handle error
  else
    ! use the descriptor, e.g. call measread_diff_
  end;
end
until (index < 0f);
```

MEASGETVERSION

指定したデータファイルの Measure バージョンを返します。呼び出しでバッファーが提供される場合、プロシージャーはエンティティタイプでインデックス付けされた、バージョンの外部エンティティレコード長の配列も返します。MEASGETVERSION は次の 2 つの目的で使用できます。

- データファイルの Measure バージョンを使用して、使用する MEASFH バージョンを決定できます。
- 返されるレコード長を使用して、他のプロシージャー（MEASREAD など）によって返されるレコードのアドレスを決定できます。

```
error := MEASGETVERSION ( { dfile }              ! i
                        ,{ dfnum }               ! i
                        ,version                 ! o
                        ,[ buffer ]              ! o
                        ,[ buflen ]              ! i
```

```
,[ sysname ]            ! o
,[ release ]            ! o
,[ processor^type ]     ! o
,[ sysidbuf ]           ! o
,[ sysidbuflen ]        ! i
,[ sysidlen ] );        ! o
```

注記:

sysidbuf、*sysidbuflen*、および *sysidlen* のサポートは、Measure H05/J03 PVU から開始されます。

dfile

入力

INT .EXT：参照：12

データファイル名を含む配列です。*dfile* または *dfnum* のどちらかを指定する必要があります。指定しない場合、エラー 3201（ERR^MISSINGPARAM）が返されます。必要に応じて、両方を指定することができます。

dfnum

入力

INT：値

データファイルアクセス番号です。MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。*dfile* または *dfnum* のどちらかを指定する必要があります。指定しない場合、エラー 3201（ERR^MISSINGPARAM）が返されます。必要に応じて、両方を指定することができます。

version

出力

INT：参照：1

データファイルが作成されたときの Measure 製品バージョンです。返される値は 2 つの部分で構成されます。ビット 0:7 にはアルファベット文字（たとえば D）が含まれ、ビット 8:15 には数値（たとえば 30）が含まれています。これらを合わせて製品のバージョンを表します（たとえば D30）。

buffer

出力

INT .EXT：参照：*

エンティティタイプインデックスを使用してアクセスする 16 ビットの整数の配列です。*buffer* は通常、INT.BUFFER[0:MAX^T]として宣言されます。各配列要素は、そのエンティティタイプのレガシー型の外部レコード長を示します。*buffer* パラメーターを渡す場合は *buflen* を指定する必要があります。指定しない場合はエラー 3201（ERR^MISSINGPARAM）が返されます。

buflen

入力

INT：値

buffer 配列のレガシー型のバイト長を示す整数です。この値は通常、(MAX^T+1) * 2 に等しくなります。

sysname

出力

INT .EXT：参照：4

データファイルが作成されたシステムの名前を返す 8 バイトの文字列です。

release

出力

INT .EXT：参照：2

Measure バージョンの識別番号を含む 2 ワードからなる配列です。

```
RELEASE[0] = F40
RELEASE[1] = 1
```

processor^type

出力

INT .EXT：参照：1

Measure が動作しているシステムの CPU タイプを示す整数です。

1	TNS II
2	TXP
3	VLX
4	CLX（CLX 2000 を除く）
5	CYCLONE
6	NSR-L
7	NSR-N、NSR-P、および NSR-K
8	NSR-W
9	NSR-G、NSR-T、NSR-V
10	NSE-*n*、ここで *n* は Itanium プロセッサーのリストを表します（これらのプロセッサーのリストについては *HPE NonStop System Glossary* を参照してください）
11	TNS/X

パラメーターは、呼び出し元のプロセスのデータスタック内にあるとみなされます。

sysidbuf

出力

STRING .EXT：参照：*

（Measure H05/J03 以降の PVU）システムのシリアル番号が返されるバッファー。

sysidbuflen

入力

INT：値

（Measure H05/J03 以降の PVU）*sysidbuf* 配列のバイト長を示す整数です。

sysidlen

出力

INT：参照：1

（Measure H05/J03 以降の PVU）*sysidbuf* 配列で返されるシステムのシリアル番号のバイト長を示す整数です。

MEASINFO

データファイルから測定の構成を返します。

MEASREADCONF プロシージャーとは異なり、このプロシージャーは呼び出し元プロセスが（MEASOPEN を使用して）ファイルをすでに開いている必要がありません。MEASINFO は使用するディスクスペースが MEASREADCONF よりも少ないため、Hewlett Packard Enterprise では実際のデータレコードの取得を必要としないタスクについてはこちらを使用することをお勧めします。これを使用すると、MEASFH を作成する前にデータファイルが必要な情報を含んでいるかどうかを判断するのに役立ちます。

Measure G09 以降の PVU では、MEASINFO はジャーナルセグメント機能の構成を報告する設定を取得するためパラメーターを持ちます。

```
error := MEASINFO ( dfile               ! i
                  ,[ contab ]           ! o
                  ,[ bufsize ]          ! i
                  ,[ bytesret ]         ! o
                  ,[ starttime ]        ! o
                  ,[ stoptime ]         ! o
                  ,[ interval ]         ! o
                  ,[ entities ]         ! o
                  ,[ ctrspace ]         ! o
                  ,[ buflen   ]         ! i
                  ,[ settings ] );      ! o
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3201	err^missingparam	必須パラメーターが欠落していました。
3203	err^badparams	不正なパラメーター値が指定されました
3204	err^buftoosmall	返された外部名がバッファーサイズを超過しました。

dfile

入力

INT .EXT：参照：12

内部（12 ワード）形式でデータファイル名を含む配列です。

contab

出力

INT .EXT：参照：*

測定構成テーブルを保持するバッファーです。*contab* 配列の形式については、**MEASCONFIGURE**(469 ページ)の *contab* パラメーターを参照してください。

bufsize

入力

INT：値

contab バッファーのバイト単位のサイズ。*bufsize* は 32,000 バイトを超過することはできず、超過した場合はエラー 3203（ERR^BADPARAMS）が返されます。

bytesret

出力

INT：参照：1

contab 配列のバイトサイズ（およびターゲットバッファー *contab* に返されるバイト数）。*bytesret* が *bufsize* より大きい場合はエラー 3204（ERR^BUFTOOSMALL）が返されます。

starttime

出力

FIXED：参照：1

測定の開始時間です。

stoptime

出力

FIXED：参照：1

測定の停止時間です。-1 の値は停止時間が指定されていないことを示します。

interval

出力

FIXED：参照：1

測定の収集間隔です。-1 の値は収集間隔が指定されていないことを示します。

entities

出力

INT(32) .EXT：参照：MAX^T+1

同時に測定された各エンティティタイプのエンティティの最大数を含む配列です。この表に示すように、配列要素はエンティティタイプ順です。0 はエンティティタイプが構成に含まれていないことを示します。MEASINFO ではなく MEASSTATUS は、アクティブ測定の *entities* 配列および *ctrspace* 配列を返します。この配列をゼロベースで宣言します。次に例を示します。

```
INT(32).ENTITIES[0:MAX^T];
or
INT(32) .EXT ENTITIES[0:MAX^T];
```

エンティティタイプ	リテラル値	数値識別子
CPU	CPU^T	1
PROCESS	PROCESS^T	2

表は続く

エンティティタイプ	リテラル値	数値識別子
PROCESSH	PROCESSH^T	3
USERDEF	USERDEF^T	4
FILE	FILOP^T	5
DISCOPEN	DFILOP^T	6
DISC	DISC^T	7
DEVICE	IODEV^T	8
LINE	LINE^T	9
NETLINE	NETLINE^T	10
SYSTEM	REMSYS^T	11
CLUSTER	CLUSTER^T	12
TERMINAL	TERM^T	13
TMF	TMF^T	14
SQLPROC	SQLPROC^T	15
SQLSTMT	SQLSTMT^T	16
OPDISK	OPDISK^T	17
CONTROLLER	CTRL^T	18
ENDPOINT	ENDPOINT_T	18
SERVERNET	SVNET^T	18
DISKFILE	DISKFILE^T	19
OSSCPU	OSSCPU^T	20
OSSNS	OSSNS^T	21
FABRIC	FABRIC_T	22
（最大値）	MAX^T	24

ctrspace

出力

INT(32) .EXT：参照：MAX^T+1

各エンティティタイプによって使用されるワード単位の最大カウンタースペースを含む配列です。配列要素は数値識別子順に並んでいます。値 0 はエンティティタイプが構成に含まれていないことを示

します。MEASINFO は、現在アクティブな測定で *entities* 配列および *ctrspace* 配列を返しませんが、MEASSTATUS は配列を返します。この配列をゼロベースで宣言します。次に例を示します。

```
INT(32) .CTRSPACE[0:MAX^T];
or
INT(32) .EXT CTRSPACE[0:MAX^T];
```

buflen

入力

INT：値

entities、*ctrspace*、またはその両方のバイト単位の長さです。*buflen* は通常、[0:MAX^T]ディメンションの *entities* 配列または *ctrspace* 配列を指定する 4￥(MAX^T+1)である必要があります。より小さい量として指定すると、*entities*、*ctrspace* または両方の配列によって返される情報量を制限します（切り捨てます）。

settings

出力

INT：参照：1

（Measure G09 以降の PVU）データファイルによって表現される測定の構成属性を識別するフラグの配列です。

settings.0:8	現在は使用されていません。
settings.9	1 = このファイルではカウンターデータレコードは抑制されます。
settings.10	現在は使用されていません。
settings.11	1 = このファイルの SQL/MX ジャーナルセグメントを作成中です。
settings.12	1 = SQL/MX ジャーナルセグメントはこのファイル内に存在します。
settings.13	現在は使用されていません。
settings.14	1 = このファイルの OSS ジャーナルセグメントを作成中です。
settings.15	1 = OSS ジャーナルセグメントはこのファイル内に存在します。

SQL ジャーナルファイルを作成中の場合、ビット 12 および 13 は両方とも ON として報告されます。

MEASINFO の使用上の注意

- L シリーズの RVU で作成される測定では、このプロシージャーは FABRIC 記述子をサポートします。また、このような測定では、GMS、ポート、およびファイバーの各フィールドは、ENDPOINT/SERVERNET エンティティ記述子で使用されません。
- MEASINFO は ALLOCATESEGMENT を呼び出して、データファイルの読み取りトラバーサルのプライベート拡張データセグメント（セグメント ID 10）を取得します。拡張セグメントサイズは、読み取りバッファーにまたがるレコードを再構築するために、2 つの 30KB（30,000 バイト）の読み取りバッファーと 32KB（32,000 バイト）の一時バッファーに応じられるように設定されています。

  MEASINFO は、データファイル（非構造化、読み取り専用、待ちなし）を開き、長い転送に対して SETMODE 141 を発行し、データファイルを通じて（30KB）READX I/O を実行して、要求された情報を見つけます。

- 長さパラメーター *bufsize* および *buflen* がゼロバイトより大きく、32,001 バイト未満である必要があります。それ以外の場合はエラー 3203（ERR^BADPARAMS）が返されます。
- MEASINFO パラメーターはオプションです。ただし、

  dfile

  プロセススタックセグメント内または拡張データセグメント内に置くことができます。省略すると、エラー 3201（ERR^MISSINGPARAM）が返されます。

  contab*、*bufsize*、*bytesret

  グループ（つまり、他の 2 つなしに 1 つを渡すことはできません。これを行わない場合エラー 3201（ERR^MISSINGPARAM）が返されます）です。*contab* は、プロセススタックセグメント内または拡張データセグメント内に置くことができます。*bufsize* および *bytesret* はプロセススタックセグメントに置く必要があります。

  starttime*、*stoptime*、*interval

  渡された場合は、プロセススタックセグメントに置く必要があります。

  entities*、*ctrspace

  プロセススタックセグメント内または拡張データセグメント内に置くことができます。これらのパラメーターの一方または両方が渡された場合、パラメーター *buflen* も渡す必要があります。これを行わない場合エラー 3201（ERR^MISSINGPARAM）が返されます。

  buflen

  プロセススタックセグメントに置く必要があります。このパラメーターが渡された場合、*entities* または *ctrspace* も渡す必要があります。そうしない場合はエラー 3201（ERR^MISSINGPARAM）が返されます。

  アレイサイズを制限するためにこのパラメーターが使用される場合、ゼロベースのアレイを考慮する必要があります。たとえば、4（ゼロベース）の *buflen* は、*entities* 情報も *ctrspace* 情報も返すことはできません。8 の *buflen* は、CPU エンティティのみに *entities* 情報と *ctrspace* 情報を返すことができます。12 の *buflen* は、CPU およびプロセス情報などを返すことができます。
- 拡張データセグメントでパラメーターが宣言されている場合、それらのパラメーターは同一のセグメントに置く必要があり、MEASINFO を呼び出す前にセグメントが使用されている必要があります。
- 拡張参照としてタイプ付けされたパラメーター（*dfile* および *contab*）は、プロセススタックセグメント内か、同じ拡張データセグメント内のどちらかにあると予想されます。渡されたパラメーターが拡張データセグメントに含まれている場合、それらのパラメーターは同一のセグメントに置く必要があり、MEASINFO を呼び出す前にセグメントが使用されている必要があります。
- 拡張アレイを宣言した場合（つまり、*dfile*、*contab*、*entities*、および *ctrspace* の場合）、pTAL は自動的にセグメント管理を処理します。ただし、複数の拡張データセグメントを割り当て、管理している場合は、MEASINFO を呼び出す前に正しい拡張データセグメントが使用されていることを確認する必要があります（また、MEASINFO によるどのリターンも、拡張データセグメント 1 つだのみに行えます）。
- OSS ジャーナルファイルが作成中の場合、設定フィールドのビット 14 および 15 が両方とも報告されます。

MEASLISTCONFIG

指定された CPU の MEASCTL プロセスからシステム構成情報を取得します。各 MEASLISTCONFIG 呼び出しでは、1 つの CPU に関連付けられている 1 つのタイプのデバイスの構成情報を取得します。

```
error := MEASLISTCONFIG ( cpunum          ! i
                        ,entity           ! i
                        ,buf              ! o
                        ,bufsize          ! i
                        ,bytesret         ! o
                        ,firstcall );     ! i,o
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3204	err^buftoosmall	返された記述子がバッファーサイズを超過しました。

cpunum

入力

INT：値

構成情報の取得元となる CPU の数です。

entity

入力

INT：値：1

構成情報を必要とするエンティティタイプです。*entity* は次のいずれかである必要があります。

DISK^T	ENDPOINT_T	IODEV^T	LINE^T	NETLINE^T	SERVERNET^T	FABRIC_T

buf

出力

INT .EXT：参照：*

エンティティ記述子が返されるバッファーです。MEASLISTCONFIG は、バッファー内に収められる、できるだけ多くの記述子を返します。バッファーサイズを設定するには *bufsize* を使用します。

bufsize

入力

INT：値

バッファーサイズです。*bufsize* は、要求されたタイプの少なくとも 1 つ以上の記述子を保持するために、バッファーに対して十分な大きさでなければなりません。

バッファーが 1 つ以上の記述子を保持できない場合、エラー 3204（ERR^BUFTOOSMALL）が返されます。

bytesret

出力

INT：参照：1

バッファーに返されるバイト数です。

エラー 3204（ERR^BUFTOOSMALL）が返された場合、*bytesret* は、指定された記述子タイプに必要な最小のバッファー長さを返します。

firstcall

入出力

FIXED：参照：1

コンテキスト値です。*entity* を MEASLISTCONFIG に初めて渡す場合、*firstcall* を 0 に指定します。MEASLISTCONFIG は、*firstcall* を変更して返します。-1 以外の *firstcall* 値は、他の記述子が使用可能であることを示します。同じ *entity* 値と、返された *firstcall* 値を使用して、MEASLISTCONFIG を再び呼び出してください。

記述子がこれ以上使用できない場合、MEASLISTCONFIG は *firstcall* を-1 に設定し、エラー 3022（WARN^NO^MORE^DATA）を返します。別のエンティティの情報を取得するには、*firstcall* を 0 に再初期化し、新しい *entity* 値を指定します。

使用上の注意

- L シリーズの RVU では、このプロシージャーは、FABRIC 記述子をサポートし、返された ENDPOINT/SERVERNET エンティティ記述子の GMS、ポート、およびファイバーの各フィールドを入力しないようになりました。
- デバイスが（たとえば SCF ADD または SCF DELETE コマンドで）動的に追加または削除されたときと同時に MEASLISTCONFIG が呼び出された場合、返された構成データに重複したエントリが含まれたり、エントリが欠落することがあります。

MEASLISTENAME

Guardian ファイル名、MID、または ANS UID を、対応する外部形式の ANSI SQL 名または OSS パス名に変換します

特定の ANSI SQL 名または OSS パス名が、呼び出し時に有効（変換可能）でなければなりません。

CRVSN が指定されている場合、指定された CRVSN に一致する変換のみが名前を返します。

```
error := MEASLISTENAME ( dfnum                      ! i
                        ,fname-pathid-uid           ! i
                        ,[ sysname ]                ! i
                        ,[ crvsn ]                  ! i
                        ,[ pathid ]                 ! o
                        ,[ extname ]                ! o
                        ,[ extname_max ]            ! i
                        ,[ extname_len ]            ! o
                        ,[ index ]                  ! i,o
                        ,[ extname_type ]           ! o
                        ,[ context_crvsn ]          ! o
                        ,[ extname_format ]         ! i
                        ,[ input_type ]             ! i
                        ,[ connectionInfo ] );      ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3204	err^buftoosmall	返された外部名がバッファーサイズを超過しました。

表は続く

エラーコード		説明
3233	err^invalidosspath	指定されたパス名を解決できませんでした。
3234	err^crvsnnotspecified	CRVSN が必要でしたが指定されていません。
3236	err^missingossjournal	OSS ジャーナルセグメントが必要でしたが、存在しませんでした。
3238	err^unknownsqlname	指定された ANSI SQL 名を変換できませんでした。
3239	err^badformatsqlname	指定された ANSI SQL 名に構文エラーが含まれていたか、完全修飾されていませんでした。
3241	err^missing^sqljournal	SQL ジャーナルセグメントが必要でしたが、存在しませんでした。
3295	err^sql^api^internal	指定された ANSI SQL 名を構文解析できませんでした。
3296	err^sqlmx^map^process	connectionInfo が渡されなかったか、MEAS_SQL_MAP_INIT への呼び出しによって入力されなかった可能性が高いため、connectionInfo の内容が正しくありませんでした。

dfnum

入力

INT：値

データファイルアクセス番号または-1 です。ジャーナルセグメントを含むオープン状態の Measure データファイルのジャーナルセグメントにアクセスするには、MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。データファイルアクセス番号を省略すると、-1 が指定されます。指定されたデータファイルでジャーナルセグメントを利用できず、ファイルが現在のシステムにある場合、SQL/MX または OSS サブシステムから *dfnum* が取得されます。

fname_pathid_uid

入力

INT .EXT：参照：12

SQL/MX パーティションまたは OSS ファイルのローカル内部形式の Guardian 名を含めることができる 12 ワードの配列であるか、SQL/MX パーティションまたは OSS ファイルの内部形式の *pathid* であるか、DISKFILE^ANSI^DESC および FILE^OPEN^ANSI^DESC の SQLMX^obj^desc 構造から見つけることができる ANS UID です。最後の場合、以下の *input_type* に示すように、*input_type* を MEAS_ANSUID に設定しなければなりません。

sysname

入力

INT .EXT：参照：4

ファイルがリモート場合、*filename* で指定されたファイルの Expand システム名を含むことができる 4 ワードの配列です。

crvsn

入力

INT .EXT：参照：3

指定された *pathid* または *filename* を修飾するための CRVSN 値を格納できる 3 ワードの配列です。

pathid

出力

INT .EXT：参照：12

ファイルに関連付けられている内部形式の *pathid* です。

extname

出力

STRING .EXT：参照：*

外部名が返されるバッファーで、null で終了します。

extname_max

入力

INT：値

extname バッファーが対応可能な最大長（バイト単位）です。*extname_max* は *extname* が指定されたときに必要です。

extname_len

出力

INT：参照：1

extname で返される名前の長さ（バイト単位）で、null 終止符を含みます。*extname_len* は、*extname* が指定されたときに必要です。

index

入出力

FIXED：参照：1

反復呼び出しのコンテキスト値です。MEASLISTENAME の最初の呼び出しで、index を-1f に設定する必要があります。変換が 1 つの名前に解決される場合、インデックスは-1f として返されます。

変換が複数の名前に解決される場合、インデックスには、後続の MEASLISTENAME の呼び出しに渡す必要がある内部値が含まれます。インデックスが-1f として返される場合、データはこれ以上使用できません。終了値-1f を検出する前に MEASLISTENAME 呼び出しシーケンスの繰り返しを明示的に停止するには、無効なインデックス（index := -index）を使用して最終的な呼び出しを行います。

extname_type

出力

INT：参照：1

extname で返された名前が、OSS パス名または ANSI SQL 名として扱われるかどうかを指定します。

1	OSS パス名
2	ANSI SQL 名

context_crvsn

出力

INT：参照：1

extname バッファーに返される外部名に対応する Guardian ファイル名のインスタンスの CRVSN 値を含む 3 ワードの配列です。この値は、測定期間中に同じ Guardian ファイル名の複数のインスタンスを区別するために使用されます。

extname_format

入力

INT：参照：1

extname で返される外部 ANSI SQL 名のフォーマットを指定します。

0	SQL/MX パーティション名の形式
1	ANSI SQL オブジェクト名の形式

input_type

入力

INT：参照：1

fname_pathid_uid フィールドのデータのタイプを指定します。*input_type* はデータが ANS UID の場合に必要なパラメーターですが、それ以外の場合はオプションです。

ANSI SQL 名の場合：	
0	ファイル名のデータ型
1	パス ID のデータ型
2	ANS UID のデータ型

connectionInfo

入力

INT(32) .EXT：参照：25

呼び出し側によって少なくとも 100 バイトに割り当てられます。*connectionInfo* は MEAS_SQL_MAP_INIT_の呼び出しによって入力される必要があります。*connectionInfo* は、SQL ジャーナルが使用されていない場合にのみ必要です。

MEASLISTENAME の使用上の注意

- *pathid* が *fname_pathid_uid* で渡された場合、*crvsn* も提供する必要があります。
- ANS UID が *fname_pathid_uid* で渡された場合、*input_type* パラメーターを MEAS_ANSUID になるように指定する必要があります。
- 出力として *extname* および *extname_len* を必要とする場合は、*extname_max* を指定します。
- MEASLISTENAME の呼び出し元に *extname_format* パラメーターが含まれていない場合、Measure はパラメーターにゼロを想定します。
- 返されるインデックスを別の正の値に変更しないでください。変更した結果、MEASLISTENAME の呼び出しの無限ループになる場合があります。

例

これは ANS UID を使用して MEASLISTENAME を呼び出す方法を示す pTAL 疑似コードの簡単な例です。

```
! get the ANSI SQL name for a specified UID
uid := desc.AnsiUIDs;

if ( error := measlistename( dfnum,
                             uid,
                             !sysname!,
                             !crvsn!,
                             !pathid!,
                             ansi_name,
                             ansi_name_max,
                             ansi_name_len,
                             !index!,
                             !extname_type!,
                             !context_crvsn!,
                             !extname_format!,
                             MEAS_ANSUID )) then
   return error
else
   ! use the ansi_name
```

MEASLISTEXTNAMES

Measure サブシステムが、構造化された OSS および ANSI SQL 名の情報を、指定されたサブボリュームの EXTNAMES ファイルにリストするようにします。

```
error := MEASLISTEXTNAMES ( dfnum                ! i
                           ,volume^subvol );     ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3235	err^buildingossjournal	このファイルの OSS ジャーナルセグメントはまだ作成中です。
3236	err^missingossjournal	OSS ジャーナルセグメントが必要でしたが、存在しませんでした。
3240	err^buildingsqlournal	このファイルの SQL ジャーナルセグメントはまだ作成中です。

dfnum

入力

INT：値

MEASOPEN プロシージャーによって返されるデータファイルアクセス番号です。

volume^subvol

入力

INT .EXT：参照：8

EXTNAMES ファイルのターゲットボリュームとサブボリュームです。

ローカルシステムの場合、*volume^subvol* の形式（*string*[0:15]で再定義する場合）は次のとおりです。

volume^subvol[0]	'$'
volume^subvol[1:7]	ボリューム名（スペースによるパディングを含む）
volume^subvol[8:15]	サブボリューム名（スペースによるパディングを含む）

リモートシステムでは、形式は次のとおりです。

volume^subvol[0]	'\'
volume^subvol[1]	システム番号
volume^subvol[2:7]	ボリューム名（スペースによるパディングを含む）
volume^subvol[8:15]	サブボリューム名（スペースによるパディングを含む）

MEASLISTEXTENAMES の使用上の注意

- この呼び出しを実行する前に、SQL ジャーナルセグメントを含む測定データファイルを開き、その SQL ジャーナルセグメントをアタッチするか、OSS ジャーナルセグメントを含む測定データファイルを開き、その OSS ジャーナルセグメントをアタッチするか、あるいはこれらの両方を行う必要があります。測定データファイルがアクティブな測定のものである場合や、MEASFH がまだ SQL または OSS ジャーナルセグメントを構築している場合は、エラー 3240（ERR^BUILDINGSQLJOURNAL）またはエラー 3235（ERR^BUILDINGOSSJJOURNAL）が MEASOPEN()から返されます。ジャーナルセグメントを開いて「アタッチ」するには、MEASOPEN()オプションパラメーターを使用する必要があります。詳しくは、**MEASOPEN**(503 ページ)を参照してください。
- 指定した *volume^subvol* に EXTNAMES ファイルがすでに存在している場合、新しいエントリが既存のファイルに書き込まれ、エラー 10（重複したレコード）のすべてのインスタンスが無視されます。単一の EXTNAMES ファイルには、多数のシステムからの多くの測定に関する情報を含めることができます。

MEASLISTGNAME

OSS ファイルパス名または ANSI SQL オブジェクト名を、Guardian ファイル名の同等表現に変換します。MEASLISTGNAME は、エンティティ記述子構造で MID（PATHID および CRVSN）の内容を提供します。指定された OSS ファイルパス名または ANSI SQL 名は、呼び出し時に有効でなければなりません。OSS のディレクトリエントリーまたは ANSI SQL オブジェクトのパーティションを取得するには、MEASLISTGNAME を双方向で呼び出します。

指定された OSS ファイルパス名が OSS ディレクトリを参照している場合、ファイル名には空白が（スペースが入力された状態）返されますが、PATHID および CRVSN は有効で、MID 構造での使用でディレクトリを内部形式で正しく識別します。

```
error := MEASLISTGNAME ( dfnum                    ! i
                        ,name                     ! i
                        ,name_len                 ! i
                        ,[ filename ]             ! o
                        ,[ sysname ]              ! o
```

```
,[ pathid ]                 ! o
,[ crvsn ]                  ! o
,[ extname ]                ! o
,[ extname_max ]            ! i
,[ extname_len ]            ! o
,[ index ]                  ! i,o
,[ name_type ]              ! i
,[ connectionInfo ] );      ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3233	err^invalidosspath	指定されたパス名を解決できませんでした。
3236	err^missingossjournal	OSS ジャーナルセグメントが必要でしたが、存在しませんでした。
3238	err^unknownsqlname	指定された ANSI SQL 名を変換できませんでした。
3239	err^badformatsqlname	指定された ANSI SQL 名に構文エラーが含まれていたか、完全修飾されていませんでした。
3241	err^missing^sqljournal	SQL/MX ジャーナルセグメントが必要でしたが、存在しませんでした。
3295	err^sql^api^internal	指定された ANSI SQL 名を構文解析できませんでした。
3296	err^sqlmx^map^process	*connectionInfo* が渡されなかったか、MEAS_SQL_MAP_INIT への呼び出しによって入力されなかった可能性が高いため、*connectionInfo* の内容が正しくありませんでした。

dfnum

入力

INT：値

データファイルアクセス番号または-1 です。ジャーナルセグメントを含むオープン状態の Measure データファイルのジャーナルセグメントにアクセスするには、MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。データファイルアクセス番号を省略すると、-1 が指定されます。指定されたデータファイルでジャーナルセグメントを利用できず、ファイルが現在のシステムにある場合、SQL/MX または OSS サブシステムから *dfnum* が取得されます。

name

入力

STRING .EXT：参照：*

変換される OSS ファイルパス名または ANSI SQL オブジェクト名を含むバッファーです。渡される場合、ANSI SQL オブジェクト名は完全修飾されていなければならず、ワイルドカードが含まれていてはなりません。

name_len

入力

INT：値

name で指定される OSS ファイルパス名または指定された ANSI SQL オブジェクト名の長さ（バイト単位）です。OSS パス名である場合、長さには終端 null バイトが含まれます。

filename

出力

INT .EXT：参照：12

12 ワードの配列です。*filename* は参照先のファイルのローカルフォーム Guardian 名の内部形式を返します。指定された OSS ファイルパス名が OSS ディレクトリに解決される場合、返される *filename* にはスペースが入力されます。OSS ファイルパス名がリモートファイル名に解決される場合、ローカルフォーム名が *filename* に返され、リモートシステム名が *sysname* に返されます。

sysname

出力

INT .EXT：参照：4

4 ワードの配列です。ファイルがリモートの場合、*sysname* は *filename* に含まれるファイルの Expand システム名を返します。ファイルがローカルの場合、*sysname* はスペースを返します。

pathid

出力

INT .EXT：参照：12

12 ワードの配列です。*pathid* は、指定された OSS ファイルパス名を記述する OSS 内部の *pathid* を返します。返される *pathid* 値は、エンティティ記述子テンプレートの MID 構造で使用できます。

pathid は ANSI SQL 名では使用されません。

crvsn

出力

INT .EXT：参照：3

3 ワードの配列です。*crvsn* は、返された *pathid* に関連付けられている作成バージョンのシリアル番号です。

extname

出力

STRING .EXT：参照：*

反復呼び出しの場合、特定ディレクトリの内容についての OSS ファイルパス名が返されるか、特定の ANSI SQL オブジェクトの ANSI SQL パーティション名が返されるバッファーです。

extname_max

入力

INT：値

extname バッファーが保持できる最大の長さ（バイト単位）です。*extname* が指定される場合に必要です。

extname_len

入力

INT：参照：12

反復呼び出しの場合、*extname* で返される OSS ファイルパス名または ANSI SQL パーティション名の長さ（バイト単位）です。長さには null 終止符が含まれます。

index

入出力

FIXED：参照：1

反復呼び出しのコンテキスト値です。MEASLISTGNAME の最初の呼び出しで、*index* を-1f に設定する必要があります。変換が OSS ファイルまたは ANSI SQL パーティション名に解決される場合、*index* は-1f として返されます。

変換が OSS ディレクトリまたは ANSI SQL オブジェクト名に解決される場合、*index* には、後続の MEASLISTGNAME の呼び出しに渡す必要がある内部値が含まれます。*index* が-1f として返される場合、OSS ディレクトリまたは ANSI SQL オブジェクトについてのデータはこれ以上使用できません。終了値-1f を検出する前に OSS 名の MEASLISTGNAME 呼び出しシーケンスの繰り返しを明示的に停止するには、無効なインデックス（index := -index）を使用して最終的な呼び出しを行います。

name_type

入力

INT：値

name で渡された名前が、OSS パス名または ANSI SQL 名として扱われるかどうかを指定します。省略すると、*name* は OSS のパス名として扱われます。

1	OSS パス名
2	ANSI SQL 名

connectionInfo

入力

INT(32) .EXT：参照：25

呼び出し側によって少なくとも 100 バイトに割り当てられます。*connectionInfo* は MEAS_SQL_MAP_INIT_の呼び出しによって入力される必要があります。*connectionInfo* は、SQL ジャーナルが使用されていない場合にのみ必要です。

MEASLISTGNAME の使用上の注意

- 反復呼び出しを開始した場合、*index* が-1f として返されるまで、MEASLISTGNAME を呼び出し続ける必要があります。そうしない場合、*index* を無効にして（index := -index）、最後の呼び出しで反復処理を強制終了する必要があります。反復処理が中断された場合、呼び出し元のプログラムは、OSS ファイルシステムディレクトリのオープンを保持します。このタイプの繰り返し動作は最終的に、プログラムが 1 つのプロセスを同時に開く限度に達するという結果になります。
- 返されるインデックスを別の正の値に変更しないでください。変更すると、MEASLISTGNAME の呼び出しの無限ループになる場合があります。

例

LISTGNAME が ANSI SQL 名を処理する方法を示す pTAL 疑似コードの簡単な例は次のとおりです。

反復呼び出しの例：

```
index := -1f;
do
begin
  if (error := measlistgname(dfnum,
                             ansi_name,
                             ansi_name_len,
                             guardianname,
                             sys,
                             !pathid!,
                             !crvsn!,
                             !partname!,
                             !partname_max!,
                             !partname_len!,
                             index,
                             MEAS_ANSI_SQL_NAME )) then
    ! handle error
  else if guardianname <> " " then
  begin
    ! display guardian filename
    if partname_len then
      ! display the partition name
  end
  else
    ! display the TABLE or INDEX name
end
until (index < 0f);
```

非反復呼び出しの例：

```
! get the Guardian file name and crvsn for an ANSI SQL name
! with partition
if ( error := measlistgname(-1,
                            ansi_names,
                            ansi_name_len,
                            guardianname,
                            !sysname!,
                            !pathid!,
                            crvsn,
                            !extname!,
                            !extname_max!,
                            !extname_len!,
                            !index!,
                            MEAS_ANSI_SQL_NAME,
                            connectionInfo)) then
   return error;
```

MEASLISTOSSNAMES

Measure サブシステムが、構造化された OSS ファイルパス名の変換情報をデフォルトサブボリュームの OSSNAMES ファイルにリストするようにします。

Measure G11 以降の PVU では、MEASLISTOSSNAMES は引き続き使用できますが、OSS ファイルパス名と ANSI SQL 名の両方をサポートする **MEASLISTEXTNAMES**(491 ページ)に置き換えられています。

```
error := MEASLISTOSSNAMES ( dfnum                ! i
                           ,volume^subvol );     ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3235	err^buildingossjournal	このファイルの OSS ジャーナルセグメントはまだ作成中です。
3236	err^missingossjournal	OSS ジャーナルセグメントが必要でしたが、存在しませんでした。

dfnum

入力

INT：値

MEASOPEN プロシージャーによって返されるデータファイルアクセス番号です。

volume^subvol

入力

INT .EXT：参照：8

OSSNAMES ファイルのターゲットボリュームとサブボリュームです。

```
volume^subvol[0]            = '\'
volume^subvol[1]            = system number
volume^subvol[2:7]          = volume name (padded with spaces)
volume^subvol[8:15]         = subvolume name (padded with spaces)
```

MEASLISTOSSNAMES の使用上の注意

- この呼び出しを実行する前に、OSS ジャーナルセグメントを含む測定データファイルを開き、その OSS ジャーナルセグメントをアタッチしている必要があります。測定データファイルがアクティブな測定のものである場合や、MEASFH がまだ OSS ジャーナルセグメントの構築過程にある場合は、エラー 3235（ERR^BUILDINGOSSJOURNAL）が返されます。
- 指定した *volume^subvol* フィールドに OSSNAMES ファイルがすでに存在している場合、新しい OSSNAMES エントリが既存のファイルに書き込まれ、エラー 10（重複したレコード）のすべての完了が無視されます。単一の OSSNAMES ファイルには、多数のシステムの多くの測定に関する OSSNAMES 情報を含めることができます。

MEASLISTPNAME

Guardian ファイル名または OSS pathid を、その OSS ファイルパス名の同等表現に変換します。OSS ファイルパス名は、呼び出し時に有効でなければなりません。CRVSN が指定されている場合、指定された CRVSN に一致する変換のみがパス名を返します。

Measure G11 以降の PVU では、MEASLISTPNAME は引き続き使用できますが、OSS ファイルパス名と ANSI SQL 名の両方をサポートする **MEASLISTENAME**(487 ページ)に置き換えられています。

```
error := MEASLISTPNAME ( dfnum                ! i
                        ,fnameorpathid        ! i
                        ,[ sysname ]          ! i
```

```
,[ crvsn ]              ! i
,[ pathid ]             ! o
,[ pathname ]           ! o
,[ pathname_max ]       ! i
,[ pathname_len ]       ! o
,[ index ] );           ! i,o
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3204	err^buftoosmall	返されたパス名がバッファーサイズを超過しました。
3233	err^invalidosspath	指定されたファイル名またはパス名を解決できませんでした。
3234	err^crvsnnotspecified	CRVSN が必要でしたが指定されていません。
3236	err^missingossjournal	OSS ジャーナルセグメントが必要でしたが、存在しませんでした。
3238	err^unknownsqlname	指定された ANSI SQL 名を変換できませんでした。
3239	err^badformatsqlname	指定された ANSI SQL 名に構文エラーが含まれていたか、完全修飾されていませんでした。
3295	err^sql^api^internal	指定された ANSI SQL 名を構文解析できませんでした。

dfnum

入力

INT：値

データファイルアクセス番号または-1 です。ジャーナルセグメントを含むオープン状態の Measure データファイルのジャーナルセグメントにアクセスするには、MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。データファイルアクセス番号を省略すると、-1 が指定されます。指定されたデータファイルでジャーナルセグメントを利用できないが、ファイルが現在のシステムにある場合、OSS ファイルシステムから *dfnum* が取得されます。

fnameorpathid

入力

INT .EXT：参照：12

OSS ファイルのローカルフォーム Guardian 名の内部形式または OSS pathid の内部形式を含む 12 ワードの配列です。

sysname

入力

INT .EXT：参照：4

4 ワードの配列です。ファイルがリモートの場合、配列には *filename* で指定したファイルの EXPAND システム名を含める必要があります。

crvsn

入力

INT .EXT：参照：3

指定された pathid またはファイル名を修飾するための CRVSN 値を格納できる 3 ワードの配列です。

pathid

出力

INT .EXT：参照：12

返される OSS ファイルパス名に関連付けられている内部形式の pathid です。

pathname

出力

STRING .EXT：参照：*

OSS ファイルパス名が返されるバッファーです。

pathname_max

入力

INT：値

パス名バッファーが保持できる最大の長さ（バイト単位）です。

pathname_len

出力

INT：参照：1

pathname で返される OSS ファイルパス名の長さ（バイト単位）です。

index

入出力

FIXED：参照：1

反復呼び出しのコンテキスト値です。最初の呼び出しで、*index* は-1f に初期化されます。要求された変換について他の OSS ファイルパス名が存在しない場合、*index* は-1f として返されます。それ以外の場合、*index* は反復処理のために後続の MEASLISTPNAME()呼び出しに渡すことができる内部値が含まれます。

MEASLISTPNAME の使用上の注意

- PATHID が *fnameorpathid* で渡された場合、CRVSN も指定する必要があります。
- PATHID と *pathname_len* が出力として必要な場合は、*pathname_max* を指定する必要があります。

MEASMONCONTROL

Measure サブシステムを起動または停止します。または、指定された CPU の測定制御プロセス（MEASCTL）を開始することによって Measure サブシステムに CPU を追加します。

注記:

呼び出し元プロセスは、このプロシージャーを呼び出すためのスーパーグループユーザー（255,*）プロセスアクセサー ID を持っている必要があります。

```
error := MEASMONCONTROL ( meascb            ! i,o
                        ,start              ! i
                        ,[ cpunum ] );      ! i
                        ,[ errDetail ] );   ! o
```

error

INT：値

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3201	err^missingparam	必須パラメーターが欠落していました。
3202	err^bounds	無効なパラメーターアドレスが指定されました。
3203	err^badparams	不正なパラメーター値が指定されました。
3217	err^notsuper	試行された呼び出しには、スーパーグループユーザー ID（255,n）が必要です。
3218	err^monalreadyactive	Measure サブシステムはすでに起動されているため、もう一度起動することはできません。
3219	err^nomeasmon	Measure サブシステムが起動されていないと、停止要求を送信します。

PROCESS_CREATE_からエラーが返される場合、これは *error* で返されます。*errDetail* パラメーターが指定された場合、これには PROCESS_CREATE_から返されるすべてのエラー詳細情報が含まれます。

meascb

入出力

INT：参照：$LEN(MEASCB^DEF) / 2

Measure サブシステムが後のプロシージャーコールのためにデータを格納する制御ブロックです。*meascb* を使用する最初の Measure プロシージャーを呼び出す前に、グローバルデータ領域内に制御ブロック用のスペースを割り当て、制御ブロックの各要素を-1 に初期化する必要があります。*meascb* を Measure プロシージャーに渡した後は、内容を変更しないでください。

$SYSTEM.SYS*nn*.MEASDECS ファイルには、制御ブロック記述子の構造定義（MEASCB^DEF）が含まれています。

start

入力

INT：値

true が指定された場合、MEASMONCONTROL は Measure サブシステムを開始します。false が指定された場合、MEASMONCONTROL は Measure サブシステムを停止します。Measure サブシステムを停止すると、すべての測定が停止します。

cpunum

入力

INT：値

指定された CPU の測定制御プロセス（MEASCTL）を作成します。現在アクティブなすべての測定は、新しく追加された CPU から自動的に情報の受信を開始します。このパラメーターは、*start* が true の場合にのみ適用されます。通常の動作状況では、このパラメーターは使用されません。Measure サブシステムは、再ロードされた CPU で MEASCTL を自動的に再起動します。

errDetail

出力

INT：参照：1

（H06.20/J06.09 以降の RVU）PROCESS_CREATE_エラー詳細番号を含みます（ START が true で、MEASMON を開始できず、かつ *error* に PROCESS_CREATE_のエラー番号が含まれる場合）。PROCESS_CREATE_のエラー番号とその定義の完全なリストについては、*Guardian プロシージャー コールリファレンス マニュアル*を参照してください。

MEASMONCONTROL の使用上の注意

H06.25/J06.14 以降の RVU で CID テーブル上限を 1,024,000 CID に設定するには、次のいずれかを実行します。

- DEFINE 文を実行します。

  ```
  ADD DEFINE =_SET_MAX_CIDS_TO_1024K, file a
  ```

  続いて、MEASMONCONTROL 呼び出し可能プロシージャーを呼び出すプロセスを起動して、Measure サブシステムを起動します。
- DEFINEADD を呼び出して、クラスファイル DEFINE “=_SET_MAX_CIDS_TO_1024K”を呼び出し元プロセスのコンテキストに追加し、続いて MEASMONCONTROL 呼び出し可能プロシージャーを呼び出して Measure サブシステムを起動します。

MEASMONCONTROL が DEFINE を有効にして MEASMON を開始します。

MEASMONSTATUS

現在アクティブな（または構成されている）測定の数およびその測定番号とデータファイル名を返します。オプションパラメーター *settings* はいくつかの Measure サブシステムの現在の設定を示すビットマスクを返します。

Measure G11 以降の PVU では、MEASMONSTATUS 設定パラメーターには SQL/MX ジャーナルステータスを示す新しいビット（settings.<8>）があります。

```
error := MEASMONSTATUS ( meascb              ! i,o
                        ,measurements        ! o
                        ,measnames           ! o
                        ,[ settings ] );     ! o
```

meascb

入出力

INT：参照：$LEN(MEASCB^DEF) / 2

Measure サブシステムが後のプロシージャーコールのためにデータを格納する制御ブロックです。*meascb* を使用する最初の Measure プロシージャーを呼び出す前に、グローバルデータ領域内に制御ブロック用のスペースを割り当て、制御ブロックの各要素を-1 に初期化する必要があります。*meascb* を Measure プロシージャーに渡した後は、内容を変更しないでください。

$SYSTEM.SYS*nn*.MEASDECS ファイルには、制御ブロック記述子の構造定義（MEASCB^DEF）が含まれています。

measurements

出力

INT：参照：1

現在アクティブな測定の数です。

measnames

出力

INT .EXT：参照：*

12 ワード形式の測定データファイル名の配列です。測定データファイル名の配列インデックスは、測定番号です。配列サイズは 64 ファイル（64 * 12 ワード）です。

settings

出力

INT：参照：1

定義を通じて制御される Measure サブシステムパラメーターの現在の設定を示すフラグの配列です。

settings.0:2	最大の CID 設定。 0 = 1,024,000 1 ... 7 = 現在は使用されていません。
settings.3:7	現在は使用されていません。
settings.8	SQL/MX ジャーナルセグメントのデフォルトのオン/オフ。
settings.9	OSS ジャーナルセグメントのデフォルトのオン/オフ。
settings.10	内部使用のために予約済み。
settings.11	スキュー間隔コピー時間はオン/オフです。
settings.12	CID テーブルのオン/オフをロックします。
settings.13:15	最大の CID 設定。 0 = 32,000 1 = 64,000 2 = 96,000 3 = 128,000 4 = 192,000 5 = 256,000 6 = 512,000 7 = 最大の CID 設定が settings.0:2 で検出できることが示されます

使用上の注意

プロセッサー内のカウンターレコードごとに、Measure はカウンター識別子（CID）を通じてカウンターレコードを割り当てて追跡します。CID 値は CID テーブルサイズに対応します。現在使用されている CID の数が、各プロセッサーの CPU エンティティレポートで報告されます。プロセッサーでサポートされている CID の数は制限されています。

- G05 以前の Measure PVU では、CID テーブルには、プロセッサーあたり 32,000 CID という上限があります。
- Measure D45、G05 以降の PVU では、START MEASSUBSYS 時に、32,000 CID の増分でプロセッサーあたり 64,000、96,000、または 128,000 になるように CID テーブル上限を増やすことができます。
- H シリーズ以降の RVU で、CID テーブルのデフォルトサイズは、プロセッサーあたり 64,000 CID ですが、START MEASSUBSYS 時に、32,000 CID の増分で 256,000 まで CID テーブル上限を増やすこともできます（196,000 もサポートされています）。
- Measure H04/J02 以降の PVU では、512,000 の最大値がサポートされています。
- H06.25/J06.14 以降の RVU では、1,024,000 の最大値がサポートされています。

MEASOPEN

測定データファイルへの読み取り、書き込み、または読み取りおよび書き込みアクセスを取得します。ファイルを開き、ファイルにアクセスする Measure ファイル処理（MEASFH）プロセスを作成します。*write* および *read* パラメーターを使用して、アクセスタイプを指定します。データファイルを閉じるには、MEASCLOSE プロシージャーを使用します。

リモートデータファイルにアクセスする最善の方法は、*measfh* パラメーターを使用することです。Measure ファイル処理（MEASFH）プログラムを含む（リモートシステム上の）オブジェクトファイルを指定します。Measure は次に、MEASFH プログラムを実行してデータファイルを読み取るためのプロセスをリモートシステム上で開始します。MEASOPEN にはオプションを指定するパラメーターがあります。このフィールドの 2 つのビットは、OSS ジャーナルセグメント機能のデフォルト処理をオーバーライドします。

```
error := MEASOPEN ( dfile                ! i
                  ,dfnum                 ! o
                  ,write                 ! i
                  ,read                  ! i
                  ,[ measfh ]            ! i
                  ,[ swapvol ]           ! i
                  ,[ cpunum ]            ! i
                  ,[ options ]           ! i
                  ,[ filesize ] ) ;      ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3203	err^badparams	不正なパラメーター値が指定されました。
3240	err^buildingsqljournal	このファイルの SQL ジャーナルセグメントはまだ作成中です。

dfile

入力

INT：参照：12

データファイル名を含む配列です。

dfnum

出力

INT：参照：1

データファイルアクセス番号です。後続のプロシージャーコールでこの番号を使用して、データファイルを識別します。Measure サブシステムは、Measure ファイル処理（MEASFH）プロセスオープンのファイル番号を、この一意のファイルアクセス番号として使用します。

write

入力

INT：値

アクセスタイプを指定するために *read* とともに使用されます。

書き込み	読み取り	結果
True	True/False	データファイルへの書き込みアクセスを取得し、データファイルを自動的に初期化します。初期化中に、ファイル内のすべてのデータは削除されます。ファイルコード 175 を持つ非構造化ローカルディスクファイルでなければなりません。ファイルが存在しない場合は作成されます。
False	True	データファイルへの読み取りアクセスを取得します。データファイルはローカルまたはリモートのディスクファイルでなければなりません。

read

入力

INT：値

アクセスタイプを指定するために *write* とともに使用されます。指定可能な値および結果については、*write* を参照してください。

measfh

入力

INT：参照：12

Measure ファイル処理（MEASFH）プログラムのオブジェクトファイル名を含む配列です。デフォルトでは、MEASFH オブジェクトファイルの名前は$SYSTEM.SYS*nn*.MEASFH です。

swapvol

入力

INT：参照：4

MEASFH スワップファイルのボリューム名を含む配列です。

swapvol が指定されていない場合、スワップファイルは呼び出し元のプロセスのスワップボリューム上に作成されます。

cpunum

入力

INT：値

MEASFH が作成される CPU の番号です。

options

入力

INT：値

MEASOPEN 要求を修飾するフラグの配列です。

options.0:8	現在は使用されていません。
options.9	測定データファイルにカウンターデータレコードを書き込むためのデフォルトモードを上書きします。
0	カウンターデータレコードが含められます（デフォルト）。
1	カウンターデータレコードが抑制されます。
options.10:11	現在は使用されていません。
options.12:13	SQL/MX ジャーナルセグメント処理でデフォルトモードを上書きします。
0	SQL/MX は指定されません。Measure サブシステムのデフォルト設定を使用して、ジャーナルセグメントの処理を決定します。
1	SQL/MX がオンになっており、SQL/MX ジャーナルセグメント機能が含まれます。
2	SQL/MX がオフになっており、SQL/MX ジャーナルセグメント機能が含まれません。
3	値が無効です。err^badparams が返されます。
options.14:15	OSS ジャーナルセグメント処理でデフォルトモードを上書きします。
0	OSS は指定されません。Measure サブシステムのデフォルト設定を使用して、ジャーナルセグメントの処理を決定します。
1	OSS がオンになっており、OSS ジャーナルセグメント機能が含まれます。
2	OSS がオフになっており、OSS ジャーナルセグメント機能が含まれません。
3	値が無効です。err^badparams が返されます。

filesize

入力

FIXED：値

目的のファイルサイズ（容量）です（バイト単位）。*filesize* の最小値は 133169152（127 MB）で、最大値は 1099507433472（1048572 MB）または最大ディスクサイズです（1048572 MB 未満の場合）。デフォルト値は 1073741824（1024 MB）です。-1 の値はファイルサイズが指定されていないことを示します。*filesize* が許可される最小値より小さいか、許可される最大値より大きい場合、エラー 3203（ERR^BADPARAMS）が返されます。ファイルがすでに存在する場合、*filesize* パラメーターは無視され、既存のファイルの容量を使用します。

MEASOPEN 使用上の注意

- Measure は、L シリーズ RVU の H06 以前のデータファイルバージョンの解析をサポートしていません。L シリーズの RVU での測定解析の場合、Measure は、Measure H06/J04 以降の PVU で生成された H06 データファイルバージョンと、Measure L01 以降の PVU で生成された H07 データファイルバージョンをサポートします。

  L シリーズの RVU で H06 データファイルの解析をサポートするには､H06 データファイルバージョンをサポートしている TNS/X バージョンの MEASFH（MEASFH06 と呼ばれます）を、個別の製品番号を通じて利用できます。間違ったデータファイルか MEASFH バージョンのいずれかが指定された場合、エラー 3290（ERR^VERSION^NOT^SUPPORTED）が返されます。
- 読み取りアクセス権での MEASOPEN の呼び出しは、ADD MEASUREMENT コマンドを実行した場合と同じです。MEASOPEN プロシージャーは、MEASFH プロセスを作成し、そのプロセスを開き、データファイルの名前を送信して MEASFH プロセスに追加します。MEASFH プロセスは、データファイルをスキャンし、将来のアクセス要求のために一連のインメモリテーブルインデックスを構築します（ADD MEASUREMENT 操作で行われるものと同じ）。このプロシージャーに関連付けられたディスクおよびメモリリソースコストを知っておく必要があります。
- 書き込みアクセス権での MEASOPEN の呼び出しは、データファイルの内容のパージになります。データのパージは、MEASFH プロセスによって実行されます。
- これらの LITERAL は MEASDECS および MEASCHMA で定義されます。

```
LITERAL OSSNOTSPECIFIED = 0;
LITERAL OSSFORCEDON = 1;
LITERAL OSSFORCEOFF = 2;

LITERAL  SQLNOTSPECIFIED = 0;
LITERAL  SQLFORCEDON = 1;
LITERAL  SQLFORCEDOFF = 2;
```

- 書き込みアクセス権を指定する場合、*options* パラメーターによって、OSS か SQL/MX のどちらのジャーナルセグメントが測定値で構成されているかが決まります。

  読み取りアクセス権を指定する場合、*options* パラメーターによって、Measure データファイル内の OSS か SQL/MX のどちらのジャーナルセグメントが呼び出し元のプロセスにアタッチされているか、および OSS か SQL/MX のどちらのジャーナルセグメント機能に関するエラーまたは警告メッセージが呼び出し元のプロセスに渡されるかが決まります。
- MEASOPEN の呼び出し元に *options* パラメーターが含まれていない場合、OSSFORCEDOFF または SQLFORCEDOFF が想定されます。書き込みアクセス権を指定した場合、ジャーナルセグメントの構築は行われません。読み取りアクセス権を指定した場合、データファイルから OSS または SQL/MX ジャーナルセグメントのアタッチは試みられません。*options* パラメーターのこの処理は､OSS または SQL/MX ジャーナルセグメント機能を利用できないアプリケーションによるシステムでの意図しないパフォーマンスの低下を防ぎます。
- プログラムにより測定を開始するプログラムは（たとえば、通常、システム監視、システム管理、OSS パス名変換サービスを含むプログラム）、ジャーナルセグメント機能ではなく、システムの OSS ネームサーバー機能を使用するように推奨されます。
- これらの LITERAL は MEASDECS および MEASCHMA で定義されます。

```
LITERA   SQLNOTSPECIFIED = 0
L

LITERA   SQLFORCEDON = 1
L

LITERA   SQLFORCEDOFF = 2
L
```

- Measure G11 以降の PVU では、MEASOPEN *options* パラメーターには、SQL/MX ジャーナルセグメント機能のデフォルトの処理をオーバーライドするために使用される 2 つの新しいビット

（options.<12:13>）があり、MEASOPEN は、SQL/MX ジャーナルセグメントが構築中の場合に新しいエラーコードを返します。

- Measure H02 以降の PVU では、MEASOPEN 呼び出し可能プロシージャーは、呼び出し元が、測定データファイルのサイズを選択するか、測定データファイル内のカウンターデータレコードを抑制するか、その両方を行えるようにします。

MEASREAD

測定データファイルから 1 つ以上のカウンターレコードを読み取ります。単一の呼び出しで返されるレコードの最大数は、ターゲットバッファーのサイズ *loc* によって異なります（完全なレコードのみが返されます）。データファイルは現在アクティブな測定に関連付けることができる、または非アクティブな測定からのデータを含めることができます。測定データファイルから読み取る前に、MEASOPEN プロシージャーを使用して、ファイルへの読み取りアクセス権を取得する必要があります。

```
error := MEASREAD ( dfnum                       ! i
                  ,entitydesc                   ! i
                  ,loc                          ! o
                  ,bufsize                      ! i
                  ,bytesret                     ! o
                  ,firstcall                    ! i,o
                  ,[ nomtime ]                  ! i
                  ,[ timetol ]                  ! i
                  ,[ version ]                  ! i
                  ,[ templateversion ] );       ! i
```

dfnum

入力

INT：値

データファイルアクセス番号です。MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。

entitydesc

入力

INT：参照：*

目的のカウンターレコードのエンティティタイプおよびエンティティ仕様です。MEASCONFIGURE プロシージャーの *contab* の説明にリストされている記述子の 1 つとして *entitydesc* を指定します。複数の仕様のテンプレートとして *entitydesc* を指定した場合、MEASREAD は、テンプレートに適合するすべてのカウンターレコードを返します。

（Measure H01 以降の PVU）ANSI SQL オブジェクトまたはパーティションの場合、*entitydesc* は DISCOPEN、DISKFILE、FILE、または SQLSTMT エンティティ記述子にできます。

Measure H07 以降の PVU では、MEASREAD は FABRIC エンティティ記述子を受け入れます。

loc

出力

INT .EXT：参照：*

カウンターレコードのターゲットバッファーです。カウンターレコードは、DDL レコード形式でバッファーに書き込まれます。エンティティタイプごとの DDL レコード定義については、**エンティティおよびカウンター**(167 ページ)を参照してください。

bufsize

入力

INT：値

ターゲットバッファー *loc* のサイズ（バイト単位）です。

bytesret

出力

INT：参照：1

ターゲットバッファー *loc* に返されるバイト数です。

firstcall

入出力

FIXED：参照：1

コンテキスト値です。*entitydesc* を MEASREAD に初めて渡す場合、*firstcall* を 0 に指定します。MEASREAD は、*firstcall* を変更して返します。MEASREAD がゼロ以外の値を *firstcall* に返す場合は、追加のカウンターレコードが利用できます。同じ *entitydesc* 値と、返された *firstcall* 値を使用して、MEASREAD を再度呼び出します。

nomtime

入力

FIXED：値

指定時間です。*nomtime* を省略（または-1 を指定）した場合、MEASREAD はファイルに書き込まれたすべてのカウンターレコードを返します。*nomtime* を指定して *timetol* を省略（または-1 を指定）した場合、MEASREAD は測定の開始から、*nomtime* 以降の最も早いレコードまでに書き込まれるカウンターレコードを返します。*nomtime* および *timetol* の両方を指定した場合、MEASREAD は、測定の開始から、*nomtime* に *timetol* の時間を加算した時間までに書き込まれたカウンターレコードを返します *nomtime* のデフォルトは-1 です。

timetol

入力

FIXED：値

許容値です。詳しくは、*nomtime* を参照してください。*timetol* のデフォルトは-1 です。

version

入力

INT：値

データファイルが作成されたときの Measure 製品バージョンです。値は 2 つの部分で構成されます。ビット 0:7 にはアルファベット文字（たとえば D）が含まれ、ビット 8:15 には数値（たとえば 30）が含まれています。これらを合わせて製品のバージョンを表します（たとえば D30）。

templateversion

入力

FIXED：値

（Measure G11 以降の PVU）ZMS 形式の外部レコードが必要な場合、*entitydesc* で要求されるエンティティタイプの MEASDDLS からの適切なテンプレートバージョンのリテラルです。省略されるか 0F として渡される場合、Legacy 形式のレコードが返されます。

-1F として渡された場合、H06.15/J06.04 RVU 以降では、現在のリリースの *templateversion* が使用されます。このような場合、返される外部レコードは、アプリケーションがコンパイルされたときに使用したカウンターレコード定義と一致しないことがあります。

使用上の注意

- L シリーズの RVU で作成される測定では、このプロシージャーは FABRIC 記述子をサポートします。また、このような測定では、GMS、ポート、およびファイバーの各フィールドは、ENDPOINT/SERVERNET エンティティ記述子で使用されません。
- L シリーズの RVU で、FABRIC エンティティと Legacy 形式のレコードが両方指定された場合、このプロシージャーは ERR^NO^LEGACY^STYLE を返します。
- Measure H01 以降の PVU では、MEASREAD は ANSI SQL オブジェクトまたはパーティションに、DISCOPEN、DISKFILE、FILE、および SQLSTMT エンティティ記述子を受け入れます。

MEASREAD_DIFF_

このプロシージャーは、1 つ以上のカウンターレコードのウィンドウを測定データファイルから読み取る MEASREAD の拡張バージョンです。

測定データファイルから読み取る前に、MEASOPEN プロシージャーを使用して、ファイルへの読み取りアクセス権を取得する必要があります。データファイルは現在アクティブな測定に関連付けることができる、または非アクティブな測定からのデータを含めることができます。

MEASREAD とは異なり、呼び出し元は開始時間と停止時間の両方を指定します。返されるレコードは、指定されたウィンドウ内で収集された統計情報を提供します。

指定された時間枠のすべてのレコードを取得するには、繰り返し呼び出す必要があります。特定の呼び出しでは、完全なレコードのみが返されます。特定の呼び出しで返される数は、*loc*（ターゲットバッファー）のサイズによって異なります。

Measure G09 以降の PVU では、パラメーター if^item には、長い（128 バイトの）SQL/MX 実行単位名の使用可能なフィールドテンプレート if^item^g09^def があります。新しいテンプレートは G09 より前の if^item^def テンプレートと共存するため、既存のアプリケーションと古いデータファイルとの互換性が維持されます。

```
error := MEASREAD_DIFF_ ( dfnum                    ! i
                         ,entitydesc               ! i
                         ,loc                      ! o
                         ,bufsize                  ! i
                         ,bytesret                 ! o
                         ,firstcall                ! i,o
                         ,from^time                ! i
                         ,to^time                  ! i
                         ,[ timetol ]              ! i
                         ,[ version ]              ! i
                         ,[ all^recs ]             ! i
                         ,[ zero^reports ]         ! i
                         ,[ totals ]               ! i
                         ,[ loadid ]               ! i
                         ,[ if^item ]              ! i
                         ,[ totals^num ]           ! o
                         ,[ templateversion ] );   ! i
```

dfnum

入力

INT：値

データファイルアクセス番号です。MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。

entitydesc

入力

INT .EXT：参照：*

目的のカウンターレコードのエンティティタイプおよびエンティティ仕様です。MEASCONFIGURE プロシージャーの *contab* の説明にリストされている記述子の 1 つとして *entitydesc* を指定します。複数の仕様のテンプレートとして *entitydesc* を指定した場合、MEASREAD_DIFF_は、テンプレートに適合するすべてのカウンターレコードを返します。

（Measure H01 以降の PVU）ANSI SQL オブジェクトまたはパーティションの場合、*entitydesc* は DISCOPEN、DISKFILE、FILE、または SQLSTMT エンティティ記述子にできます。SQLSTMT ANSI エンティティ記述子は、SQLSTMT カウンターレコードを取得するために ANSI SQL 名を指定する場合に使用されます。

Measure H07 以降の PVU では、MEASREAD_DIFF_は FABRIC エンティティ記述子を受け入れます。

loc

出力

INT .EXT：参照：*

カウンターレコードのターゲットバッファーです。カウンターレコードは、DDL レコード形式でバッファーに書き込まれます。エンティティタイプごとの DDL レコード定義については、**エンティティおよびカウンター**(167 ページ)を参照してください。

bufsize

入力

INT：値

ターゲットバッファー *loc* のサイズ（バイト単位）です。*bufsize* が 32,000 よりも大きい場合、エラー 3203（ERR^BADPARAMS）が返されます。

bytesret

出力

INT .EXT：参照：1

ターゲットバッファー *loc* に返されるバイト数です。

firstcall

入出力

FIXED .EXT：参照：1

コンテキスト値です。*entitydesc* を MEASREAD_DIFF_に初めて渡す場合、*firstcall* を 0 に指定します。MEASREAD_DIFF_は、*firstcall* を変更して返します。MEASREAD_DIFF_がゼロ以外の値を *firstcall* に返す場合は、追加のカウンターレコードが利用できます。同じ *entitydesc* 値と、返された *firstcall* 値を使用して、MEASREAD_DIFF_を再度呼び出します。

from^time

入力

FIXED：値

測定ウィンドウの開始時間です。*from^time* は、システムプロシージャー JULIANTIMESTAMP によって提供された 4 ワード/1 マイクロ秒形式の地方平均時に基づくユリウス日として指定します（JULIANTIMESTAMP の説明については、*Guardian プロシージャーコールリファレンスマニュアル*を参照してください）。-1 の値は測定の開始を示します。

to^time

入力

FIXED：値

測定ウィンドウの停止時間です。*to^time* は、システムプロシージャー JULIANTIMESTAMP によって提供された 4 ワード/1 マイクロ秒形式の地方平均時に基づくユリウス日として指定します（JULIANTIMESTAMP の説明については、*Guardian プロシージャーコールリファレンスマニュアル*を参照してください）。-1 の値は測定の終了を示します。

timetol

入力

FIXED：値

from^time と *to^time* で囲まれた時間枠に適用される許容値で、実質的に、*from^time* から *timetol* を引いた時間から、*to^time* に同じ量を足した時間までの時間ウィンドウになります。-1 の値を指定すると、*from^time* よりも前の最後のレコードと、*to^time* よりも後の最も早いレコードが含まれるように時間枠が広げられます。

version

入力

INT：値

データファイルが作成されたときの Measure 製品バージョンです。返される値は 2 つの部分で構成されます。ビット 0:7 にはアルファベット文字（たとえば D）が含まれ、ビット 8:15 には数値（たとえば 30）が含まれています。これらを合わせて製品のバージョンを表します（たとえば D30）。

all^recs

入力

INT：値

値 1 が渡された場合は、指定されたウィンドウ内のすべての間隔レコードが要求されます。省略するか 0 が指定された場合、概要レコードのみが返されます。

zero^reports

入力

INT：値

値 0 が渡された場合は、ゼロレコードの除外を要求します。省略するか 1 が指定された場合、ゼロレコードが返されます。

totals

入力

INT：値

0、1、または 2 の値で、それぞれ次を示します。

0	合計レコードが省略され、個々のカウンターレコードが含まれます（デフォルト）。
1	個々のレコードと合計レコードが含まれます。
2	個々のカウンターレコードが省略され、合計レコードが含まれます。

loadid

入力

INT .EXT：参照：4

返されるすべてのレコードに含まれる 8 バイトの文字列です。

if^item

入力

INT .EXT：参照：$LEN(IF^ITEM^G09^DEF) / 2

Measure G09 以降の PVU で、長い（128 バイトの）SQL/MX 実行単位名をサポートしている構造です。

```
STRUCT IF^ITEM^G09^DEF (*);
Begin
Int     Item^Name[0:15];  ! Item name in entity record
Int     Relation^Operator;! Relation Operator
                          ! MEASREAD_DIFF_の使用上の注意 (515 ページ) を参照し
てください。
                          ! Bit 0 must be 1
                          !
                          ! Bits 13:15
                          !   1 -> Equal        =
                          !   2 -> Not Equal   <>
                          !   3 -> Less Than    <
                          !   4 -> Greater Than >
Fixed Value;              ! Value used in comparison
Int     Rate;             ! Use rated or nonrated value
                          ! for condition checking:
                          !   0 -> No-Rate, 1 -> Rated
Int     SQLName[0:63];    ! for SQLSTMTs only
Int     SQLIndex;         ! SQL Index number
String  IP^addr[0:15] = SQLName;
Int     port = SQLIndex;
Int     IP^wildcard^flags = value;
End;

DEFINE  IP^addr^wildcard^flag = IP^wildcard^flags.<0>#,
        port^wildcard^flag = IP^wildcard^flags.<1>#;
```

または

if^item

入力

INT .EXT：参照：$LEN(IF^ITEM^DEF) / 2

既存のアプリケーションと G09 以前の Measure PVU での古いデータファイルとの互換性を提供する構造です。

```
STRUCT IF^ITEM^DEF (*);
Begin
Int     Item^Name[0:15];  ! Item name in entity record
Int     Relation^Operator;! Relation Operator
                          !   1 -> Equal        =
                          !   2 -> Not Equal    <>
                          !   3 -> Less Than    <
                          !   4 -> Greater Than >
Int(32) Value;            ! Value used in comparison
Int     Rate;             ! Use rated or nonrated value
                          ! for condition checking:
                          !   0 -> No-Rate, 1 -> Rated
Int     SQLName[0:15];    ! SQL run-unit in Ascii
```

```
Int     SQLIndex;          ! SQL Index number
End;
```

item^name

入力

エンティティレコードで定義されたカウンター名のいずれかです（たとえば、CPU レコードの場合は CPU-BUSY-TIME）。

relation^operator

入力

比較タイプを指定する 1 から 4 までの範囲の値を示します。

1	等しいかどうかを比較します
2	等しくないかどうかを比較します
3	「小なり」かどうかを比較します
4	「大なり」かどうかを比較します

value

入力

特定のカウンターの値との比較で使用する 32 ビットの整数です。*value* = *n*`*1000`。たとえば、17.09 は 17090 として保存されます。

rate

入力

非定格値または定格値のどちらを比較に使用するかを決定します。

- 0 は非定格値を示します（REPORT RATE OFF に相当）。
- 1 は定格値を示します（REPORT RATE ON に相当）。

sqlname

入力

SQL 実行単位を指定する 16 ワードの配列です。*sqlindex* はインデックス番号を指定します。これらの 2 つのフィールドが、SQLSTMT エンティティと組み合わせて使用されます。渡された場合、MEASREAD_DIFF_は指定された実行単位とインデックス番号を含むレコードを返します。

sqlindex

入力

SQL インデックスです。*sqlname* を参照してください。値-1 はすべての SQL インデックスを指定します。

注記:

if^item^def 構造が渡された場合、最初に初期化する必要があります。値を持たない構造を渡す場合、IF^ITEM^DEF.Item^Name を 0 に初期化する必要があります。同様に、IF^ITEM^DEF.SQLName の値が 0 の場合、SQL（*run-unit,index*）が指定されないことを意味します。

ip^addr

入力

OSS AF_INET ソケットに提供される追加の識別子です。

ip^addr は、AF_INET ソケットの FILE レコードヘッダーの一部であるアドレス/ポートの組み合わせの IP アドレス部分です。識別子は、アドレスのスペースで区切られたポートで報告されます（たとえば、"123.45.67.8 90"）。

port

入力

OSS AF_INET ソケットに提供される追加の識別子です。

port は、AF_INET ソケットの FILE レコードヘッダーの一部である IP アドレス/ポートの組み合わせのポート部分です。

ip^addr^wildcard^flags

入力

OSS AF_INET ソケットに提供される追加の識別子です。

ip^addr^wildcard^flags は *ip^addr* フィールドと *port* フィールドのワイルドカードサポートを判別します。

< 0 >	0	指定されたアドレスと一致する。
	1	任意のアドレスと一致する。
< 1 >	0	指定されたポート番号と一致する。
	1	任意のポート番号と一致する。

データレコード内の IP アドレスは、IP コンポーネントごとに 1 バイトの 16 進数で表示されます。ここでは、さまざまな *ip^addr*、*port*、および *ip^wildcard^flags* フィールド指定のいくつかの例を示します。

例 1：IP アドレス = 11.22.33.44、任意のポート番号と一致

```
if^item
 = {....
    IP^ADDR = {0xb, 0x16, 0x21, 0x2c, 0x0 <repeats 2 times>},
    PORT = 0x0,
    IP^WILDCARD^FLAGS = 0x4000
    }
```

例 2：IP アドレス = 任意の IP アドレス番号と一致、ポート 37

```
if^item
 = {....
    IP^ADDR = {0x0 <repeats 16 times>},
    PORT = 0x25,
    IP^WILDCARD^FLAGS = 0x8000
    }
```

例 3：IP アドレス = 11.22.33.44、ポート 37

```
if^item
 = {....
    IP^ADDR = {0xb, 0x16, 0x21, 0x2c, 0x0 <repeats 12 times>},
    PORT = 0x25,
    IP^WILDCARD^FLAGS = 0x0
    }
```

例 4：IPv6 アドレス = 1070::800:2006:4178、ポート 37

```
if^item
 = {....
    IP^ADDR = IP^ADDR = {0x10, 0x70, 0x0, 0x0, 0x0, 0x0, 0x0,
    0x0, 0x0, 0x0, 0x8, 0x0, 0x20, 0x6, 0x41, 0x78},
    PORT = 0x25
    IP^WILDCARD^FLAGS = 0x0
    }
```

totals^num

出力

INT(32) .EXT：参照：1

概要レコードを累積するために使用するレコードの数です。

templateversion

入力

FIXED：値

（Measure G11 以降の PVU）ZMS 形式の外部レコードが必要な場合、*entitydesc* で要求されるエンティティタイプの MEASDDLS からの適切なテンプレートバージョンのリテラルです。省略されるか 0F として渡される場合、Legacy 形式のレコードが返されます。

-1F として渡された場合、H06.15/J06.04 RVU 以降では、現在のリリースの *templateversion* が使用されます。このような場合、返される外部レコードは、アプリケーションがコンパイルされたときに使用したカウンターレコード定義と一致しないことがあります。

処理可能な形式に対応する外部レコードの取得については、**TEMPLATE-VERSION**(180 ページ)を参照してください。

MEASREAD_DIFF_の使用上の注意

- L シリーズの RVU で作成される測定では、このプロシージャーは FABRIC 記述子をサポートします。また、このような測定では、GMS、ポート、およびファイバーの各フィールドは、ENDPOINT/SERVERNET エンティティ記述子で使用されません。
- L シリーズの RVU で、FABRIC エンティティと Legacy 形式のレコードが両方指定された場合、このプロシージャーは ERR^NO^LEGACY^STYLE を返します。
- さらに長い IF^ITEM^G09^DEF テンプレートが使用された場合、define G09^format を設定する必要があります。G09^format が設定されていない場合、*sqlname* フィールドは *sqlindex* として解釈され、返されるレコードが間違っている可能性があります。

```
DEFINE G09^format         = relation^operator.0#;
DEFINE G09^rel^operator   = relation^operator.13:15#;
```

  Measure G09 データファイルにアクセスするときは、いずれかのテンプレートの形式を使用できます。また、フィールド *value* に渡された数値が以前の構造体の INT(32) *value* フィールドで渡すことのできる数を超えない場合、および *sqlname* 値が 32 文字を超えない場合、G08 以前のデータファイルにアクセスするときに G09 テンプレートを使用できます。
- Measure H01 以降の PVU では、MEASREAD_DIFF_は ANSI SQL オブジェクトまたはパーティションの DISCOPEN、DISKFILE、FILE、および SQLSTMT エンティティ記述子を受け入れます。

MEASREADACTIVE

現在アクティブなカウンターからデータを読み取ります。カウンターを読み取る前に、（一般的には **MEASMONSTATUS**(501 ページ)を使用して）有効な測定番号を入手する必要があります。

D シリーズおよび G08 以前の Measure PVU では、MEASREADACTIVE は DISCOPEN、DISKFILE、または PROCESSH カウンターを読み取ることができません。Measure G08 以降の PVU では、MEASREADACTIVE プロシージャーは、DISCOPEN または PROCESSH カウンターを読み取ることができません。これらのカウンターの値にアクセスするには、MEASREAD_DIFF_プロシージャーを使用して、現在アクティブな測定データファイルから読み取ります。測定に指定された収集間隔から、カウンター値が最近データファイルに書き込まれた頻度が分かります。

```
error := MEASREADACTIVE ( meascb                    ! i,o
                        ,measnum                    ! i
                        ,entitydesc                 ! i
                        ,loc                        ! o
                        ,bufsize                    ! i
                        ,bytesret                   ! o
                        ,[ entity^index ]           ! i,o
                        ,[ templateversion ] );     ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3204	err^buftoosmall	返されたカウンターレコードがバッファーサイズを超過しました。
3402	errm^cannotaccess	*entity^index* の値と *entity^desc* の内容が一致しません。
3404	errm^notmeasuring	*entity^index* の値と *entity^desc* の内容が一致しません。

meascb

入出力

INT：参照：$LEN(MEASCB^DEF) / 2

Measure サブシステムが後のプロシージャーコールのためにデータを格納する制御ブロックです。*meascb* を使用する最初の Measure プロシージャーを呼び出す前に、グローバルデータ領域内に制御ブロック用のスペースを割り当て、制御ブロックの各要素を-1 に初期化する必要があります。*meascb* を Measure プロシージャーに渡した後は、内容を変更しないでください。

$SYSTEM.SYS*nn*.MEASDECS ファイルには、制御ブロック記述子の構造定義（MEASCB^DEF）が含まれています。

measnum

入力

INT：値

測定番号です。MEASCONFIGURE プロシージャーによって返される *measnum* 値を使用するか、MEASMONSTATUS を使用して *measnum* 値を検出します。

entitydesc

入力

INT：参照：*

目的のアクティブなカウンターレコードのエンティティタイプおよびエンティティ仕様です。MEASCONFIGURE プロシージャーの *contab* の説明にリストされている記述子の 1 つとして *entitydesc* を指定します。MEASREADACTIVE は *entitydesc* を使用して、エンティティ制御ブロックに直接アクセスするため（エンティティ制御ブロックはその後システムデータスペースのカウンターレコードへのアクセスに使用される）、*entitydesc* は正確に 1 つのエンティティを識別する必要があります。必要に応じて、エンティティを一意に識別する名前と番号を取得するために、適切なオペレーティングシステム情報プロシージャーを使用します。

取得するデータを識別するのにフィールドのサブセットがあれば十分な場合、エンティティ記述子ですべてのフィールドを指定する必要ありません。たとえば、プロセスの CPU および PIN によってプロセスを識別する場合、プロセス名またはプログラムファイル名も指定する必要はありません。逆に、データを取得するための呼び出しで、測定の構成に使用されたフィールドよりも多くのフィールドを指定しなければならないこともあります。たとえば、測定構成の *entitydesc* でワイルドカードを指定する場合があります。測定されたオブジェクトのサブセットのみのデータを取得したい場合、MEASREADACTIVE 呼び出しの *entitydesc* では、さらに多くのフィールドが指定される可能性があります。

注記:

MEASREADACTIVE 呼び出し可能プロシージャーに渡されるエンティティ記述子では、ワイルドカードは許可されません。ワイルドカードを使用する場合は、代わりに MEAS_READACTIVE_MANY_プロシージャーを呼び出します。

（Measure H01 以降の PVU）ANSI SQL オブジェクトまたはパーティションの場合、*entitydesc* は DISKFILE、FILE、または SQLSTMT エンティティ記述子にできます。ANSI SQL 名ではなく、ファイルの Guardian 名を指定します。

Measure H07 以降の PVU では、MEASREADACTIVE は FABRIC エンティティ記述子を受け入れます。

loc

出力

INT .EXT：参照：*

カウンターレコードのターゲットバッファーです。カウンターレコードは、DDL レコード形式でバッファーに書き込まれます。エンティティタイプごとの DDL レコード定義については、**エンティティおよびカウンター**(167 ページ)を参照してください。

bufsize

入力

INT：値

ターゲットバッファー *loc* のサイズ（バイト単位）です。

bytesret

出力

INT：参照：1

ターゲットバッファー *loc* に返されるバイト数です。*bytesret* が *bufsize* より大きい場合、エラー 3204（ERR^BUFTOOSMALL）が返されます。

entity^index

入出力

FIXED：参照：1

DISKFILE レコードでの Measure G08 以降の PVU MEASREADACTIVE リクエストに適用されます。

入力では、*entity^index* は-1 か、*entity^desc* によって記述されたレコードに対する MEASREADACTIVE の以前の呼び出しによって返される値です。

出力では、*entity^index* は、Measure が MEASREADACTIVE への後続の呼び出しで *entity^desc* によって記述されるレコードを直接フェッチできるようにする内部値を含みます。-1 が指定された場合、Measure は一致するものが見つかるまでプロセッサー内のすべての DISKFILE レコードを検索します。*entity^index* が-1 でない場合、Measure はこれを、*entity^desc* によって記述された DISKFILE レコードのインデックスであるとみなします。その後、*entity^index* の値と *entity^desc* の内容が検査されます。これらが一致しない場合、エラー 3402（ERRM^CANNOTACCESS）またはエラー 3404（ERRM^NOTMEASURING）が返されます。

templateversion

入力

FIXED：値

（Measure G11 以降の PVU）ZMS 形式の外部レコードが必要な場合、*entitydesc* で要求されるエンティティタイプの MEASDDLS からの適切なテンプレートバージョンのリテラルです。省略されるか 0F として渡される場合、Legacy 形式のレコードが返されます。

-1F として渡された場合、H06.15/J06.04 RVU 以降では、現在のリリースの *templateversion* が使用されます。このような場合、返される外部レコードは、アプリケーションがコンパイルされたときに使用したカウンターレコード定義と一致しないことがあります。

MEASREADACTIVE の使用上の注意

- L シリーズの RVU では、このプロシージャーは FABRIC 記述子をサポートし、ENDPOINT/SERVERNET エンティティ記述子の GMS、ポート、およびファイバーの各フィールドをサポートしなくなりました。
- L シリーズの RVU で、FABRIC エンティティと Legacy 形式のレコードが両方指定された場合、このプロシージャーは ERR^NO^LEGACY^STYLE を返します。
- MEASREADACTIVE は、エンティティのアクティブな測定にのみ、最高 32 KB のバッファーサイズをサポートします。32 KB を超えるバッファーについては、**MEAS_READACTIVE_**(518 ページ)を参照してください。
- Measure H01 以降の PVU では、MEASREADACTIVE および MEAS_READACTIVE_は、ANSI SQL オブジェクトまたはパーティションの DISKFILE、FILE、または SQLSTMT エンティティ記述子を受け入れます。

MEAS_READACTIVE_

Measure G09 以降の PVU で現在アクティブなカウンターからデータを読み取ります。カウンターを読み取る前に、（一般的には MEASMONSTATUS プロシージャーを使用して）有効な測定番号を入手する必要があります。

MEAS_READACTIVE_プロシージャーは DISCOPEN カウンターまたは PROCESSH カウンターを読み取ることができません。これらのカウンターの値にアクセスするには、MEASREAD_DIFF_プロシージャーを使用して、現在アクティブな測定データファイルから読み取ります。測定に指定された収集間隔から、カウンター値が最近データファイルに書き込まれた頻度が分かります。

```
error := MEAS_READACTIVE_ ( meascb                    ! i,o
                          ,measnum                    ! i
                          ,entitydesc                 ! i
                          ,loc                        ! o
                          ,bufsize                    ! i
                          ,bytesret                   ! o
                          ,[ entity^index ]           ! i,o
                          ,[ templateversion ] );     ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3204	err^buftoosmall	返されたカウンターレコードがバッファーサイズを超過しました。
3402	errm^cannotaccess	*entity^index* の値と *entity^desc* の内容が一致しません。
3404	errm^notmeasuring	*entity^index* の値と *entity^desc* の内容が一致しません。

meascb

入出力

INT：参照：$LEN(MEASCB^DEF) / 2

Measure サブシステムが後のプロシージャーコールのためにデータを格納する制御ブロックです。*meascb* を使用する最初の Measure プロシージャーを呼び出す前に、グローバルデータ領域内に制御ブロック用のスペースを割り当て、制御ブロックの各要素を-1 に初期化する必要があります。*meascb* を Measure プロシージャーに渡した後は、内容を変更しないでください。

$SYSTEM.SYS*nn*.MEASDECS ファイルには、制御ブロック記述子の構造定義が含まれています（MEASCB^DEF）。

measnum

入力

INT：値

測定番号です。MEASCONFIGURE プロシージャーによって返される *measnum* 値を使用するか、MEASMONSTATUS を使用して *measnum* 値を検出します。

entitydesc

入力

INT：参照：*

目的のアクティブなカウンターレコードのエンティティタイプおよびエンティティ仕様です。MEASCONFIGURE プロシージャーの *contab* の説明にリストされている記述子の 1 つとして *entitydesc* を指定します。MEAS_READACTIVE_は *entitydesc* を使用して、エンティティ制御ブロックに直接アクセスするため（エンティティ制御ブロックはその後システムデータスペースのカウンターレコードへのアクセスに使用される）、*entitydesc* は正確に 1 つのエンティティを識別する必要があります。必要に応じて、エンティティを一意に識別する名前と番号を取得するために、適切なオペレーティングシステム情報プロシージャーを使用します。

注記:

MEAS_READACTIVE_呼び出し可能プロシージャーに渡されるエンティティ記述子では、ワイルドカードは許可されません。ワイルドカードを使用する場合は、代わりに MEAS_READACTIVE_MANY_プロシージャーを呼び出します。

（Measure H01 以降の PVU）ANSI SQL オブジェクトまたはパーティションの場合、*entitydesc* は DISKFILE、FILE、または SQLSTMT エンティティ記述子にすることができます

loc

出力

INT .EXT：参照：*

カウンターレコードのターゲットバッファーです。カウンターレコードは、DDL レコード形式でバッファーに書き込まれます。エンティティタイプごとの DDL レコード定義については、**エンティティおよびカウンター**(167 ページ)を参照してください。

bufsize

入力

INT(32)：値

ターゲットバッファー *loc* のサイズ（バイト単位）です。

bytesret

出力

INT(32)：参照：1

ターゲットバッファー *loc* に返されるバイト数です。*bytesret* が *bufsize* より大きい場合、エラー 3204（ERR ^ BUFTOOSMALL）が返されます。

entity^index

入出力

FIXED：参照：1

DISKFILE レコードのアクティブな測定要求に適用されます。

入力では、*entity^index* は-1 か、*entity^desc* によって記述されたレコードに対する MEAS_READACTIVE_の以前の呼び出しによって返される値です。

出力では、*entity^index* は、Measure が MEAS_READACTIVE_への後続の呼び出しで *entity^desc* によって記述されるレコードを直接フェッチできるようにする内部値を含みます。-1 が指定された場合、Measure は一致するものが見つかるまでプロセッサー内のすべての DISKFILE レコードを検索します。もし *entity^index* が-1 でない場合、Measure はこれを、*entity^desc* によって記述された DISKFILE レコードのインデックスであるとみなします。その後、*entity^index* の値と *entity^desc* の内容が検査されます。これらが一致しない場合、エラー 3402（ERRM^CANNOTACCESS）またはエラー 3404（ERRM^NOTMEASURING）が返されます。

templateversion

入力

FIXED：値

（Measure G11 以降の PVU）ZMS 形式の外部レコードが必要な場合、*entitydesc* で要求されるエンティティタイプについての MEASDDLS からの適切なテンプレートバージョンのリテラルです。省略されるか 0F として渡される場合、Legacy 形式のレコードが返されます。

-1F として渡された場合、H06.15/J06.04 RVU 以降では、現在のリリースの *templateversion* が使用されます。このような場合、返される外部レコードは、アプリケーションがコンパイルされたときに使用したカウンターレコード定義と一致しないことがあります。

使用上の注意

- L シリーズの RVU では、このプロシージャーは FABRIC 記述子をサポートし、ENDPOINT/ SERVERNET エンティティ記述子の GMS、ポート、およびファイバーの各フィールドをサポートしなくなりました。
- L シリーズの RVU で、FABRIC エンティティと Legacy 形式のレコードが両方指定された場合、このプロシージャーは ERR^NO^LEGACY^STYLE を返します。
- Measure H01 以降の PVU では、MEASREADACTIVE および MEAS_READACTIVE_は、ANSI SQL オブジェクトまたはパーティションの DISKFILE、FILE、または SQLSTMT エンティティ記述子を受け入れます。

MEAS_READACTIVE_MANY_

メッセージのオーバーヘッドを発生させずに、プロセッサー内で構成されている複数のアクティブカウンターレコードを取得します。このプロシージャーは、H シリーズ、J シリーズ、および L シリーズの RVU で使用できます。

MEAS_READACTIVE_MANY_プロシージャーは、PROCESSH、SQLSTMT、または USERDEF カウンターを読み取ることができません。PROCESSH カウンターの値にアクセスするには、MEASREAD_DIFF_プロシージャーを使用して、現在アクティブな測定データファイルから読み取ります。SQLSTMT または USERDEF カウンターにアクセスするには、MEAS_READACTIVE_プロシージャーを使用します。

```
error := MEAS_READACTIVE_MANY_ ( measnum                    ! i
                               ,mode                        ! i
                               ,entitydesc                  ! i
                               ,buffer                      ! o
                               ,bufsize                     ! i
                               ,recs                        ! o
                               ,context                     ! i,o
                               ,[ templateversion ] );      ! i
```

measnum

入力

INT：値

測定番号です。MEASCONFIGURE プロシージャーによって返される *measnum* 値を使用するか、MEASMONSTATUS を使用して *measnum* 値を検出します。

mode

入力

INT：値

READACTIVE_MANY_LIST (0)、READACTIVE_MANY_EXACT (1)、または READACTIVE_MANY_CHANGED (2)の値のいずれかです。これらの値のセマンティクスは、*context* パラメーターの値に依存します。

entitydesc

入力

INT .EXT：参照：*

目的のカウンターレコードのエンティティタイプおよびエンティティ仕様です。MEASCONFIGURE プロシージャーの *contab* の説明にリストされている記述子の 1 つとして *entitydesc* を指定します。複数の仕様のテンプレートとして *entitydesc* を指定した場合、MEAS_READACTIVE_MANY_は、テンプレートに適合するすべてのカウンターレコードを返します。

entitydesc にはワイルドカードを含めることができます。ワイルドカード記述子を LIST *mode* とともに使用することが、プロセッサー上で構成されているエンティティのすべてのインスタンスのレコードを取得する最適な方法です。このプロシージャーでは、記述子の照合のために、OSS パス名やプログラムファイル名などの文字列ベースのフィールドは使用しません。このような文字列が指定された場合は無視されます。これらを使用する必要がある場合は、代わりに MEASREADACTIVE プロシージャーを呼び出してください。

注記:

entitydesc には、ANSI SQL のオブジェクトまたはパーティションの場合は DISKFILE、DISCOPEN、または FILE エンティティ記述子を指定できます。ただし、現在この API では ANSI SQL 名および ANS UID のサポートがないため、これらの記述子フィールドは無視されます。

buffer

出力

INT .EXT：参照：*

カウンターレコードのターゲットバッファーです。カウンターレコードは、DDL レコード形式でバッファーに書き込まれます。エンティティタイプごとの DDL レコード定義については、**エンティティおよびカウンター**(167 ページ)を参照してください。

bufsize

入力

INT(32)：値

ターゲットバッファー *buffer* のサイズ（バイト単位）です。

recs

出力

INT .EXT：参照：1

返されるレコードの数です。

context

入出力

FIXED .EXT：参照：1

mode パラメーターの値と連動し、返されるレコードを決定します。

入力で、*context* が-1F または 0F の場合は次のようになります。

- Measure は、カウンターが *entitydesc* 内の情報と一致するまで、*entitydesc* に指定されたエンティティタイプがないかカウンタータイプを検索します。
- *mode* が READACTIVE_MANY_EXACT の場合、Measure は 1 つの項目を返します。
- *mode* が READACTIVE_MANY_LIST の場合、Measure は *bufsize* がなくなるまで、*entitydesc* が一致するすべての項目を返します。
- *mode* が READACTIVE_MANY_CHANGED の場合、Measure は *entitydesc* に一致するすべての項目を返しますが、レコードのヘッダー部分の ERROR フィールドが更新されて、レコードが最後に要求されたときから変更されたかどうかを示します。具体的には、レコードが変更されておらず、形式が ZMS の場合、*entity*.HDR.ERROR の 0 番目と 5 番目のビットは 1 に設定されます。そうでない場合、5 番目のビットは 0 です。Legacy 形式の場合、Measure はレコードが変更されていないことを示すために、*entity*.ERROR フィールドに値-4 を返します。このエラーフィールドは、返

されるカウンターレコードの一部（およびそれに固有のもの）です。これは、このプロシージャーによって返される *error* 値に関係ありません。
- 正常な値が返された場合、*context* には次に処理されるエンティティについての情報が格納されます。この情報は、その反復の以降の呼び出しのために、そのまま渡される必要があります。

出力の場合、次のようになります。

- *mode* が READACTIVE_MANY_EXACT の場合、*context* には一致したエンティティについての情報が含まれます。同じエンティティについてのデータが次回必要なときにこの値を返すと、要求のパフォーマンスが向上します。
- *mode* が READACTIVE_MANY_CHANGED または READACTIVE_MANY_LIST の場合、0F のコンテキストによって、*entitydesc* に一致するエンティティは取得できなくなっていることを示します。-1F はエラーリターンを示しています。

入力で、*context* の値が-1F または 0F 以外の場合は、次のようになります。

- *mode* が EXACT の場合、*context* によって参照されるカウンターが返されます（同じオブジェクトに適用される場合）。
- *mode* が LIST で *context* 値が同じオブジェクトに適用され、エンティティタイプが *entitydesc* と一致する場合、このカウンター以降では、*bufsize* がなくなるまで、同じタイプのカウンターレコードが返されます。LIST は、リンクされたカウンターリスト内で 1 つのパスの最後に終了します。コンテキストが同じカウンターに適用されることを確認するチェックがコンテキストで失敗した場合、常にエラーが返されます。LIST/変更されたレコードの操作は、最初から再起動する必要があります。
- *mode* が READACTIVE_MANY_CHANGED の場合、動作は LIST に類似していますが、レコードのヘッダー部分の ERROR フィールドが更新されて、レコードが最後に要求されたときから変更されたかどうかを示します。レコードが変更されておらず、形式が ZMS の場合、*entity*.HDR.ERROR の 0 番目と 5 番目のビットは 1 に設定されます。そうでない場合、5 番目のビットは 0 です。Legacy 形式の場合、レコードが変更されていないことを示すために、*entity*.ERROR フィールドに値-4 を返します。
- 正常な値が返された場合、*context* には次に処理されるエンティティについての情報が格納されます。この情報は、その反復の以降の呼び出しのために、そのまま渡される必要があります。*context* の戻り値 1F はエラーを示します。

templateversion

入力

FIXED：値

ZMS 形式の外部レコードが必要な場合、*entitydesc* で要求されるエンティティタイプについての MEASDDLS からの適切なテンプレートバージョンのリテラルです。省略されるか 0F として渡される場合、Measure は Legacy 形式のレコードを返します。

-1F として渡された場合、H06.15/J06.04 RVU 以降では、現在のリリースの *templateversion* が使用されます。このような場合、返される外部レコードは、アプリケーションがコンパイルされたときに使用したカウンターレコード定義と一致しないことがあります。

使用上の注意

- L シリーズの RVU では、このプロシージャーは FABRIC 記述子をサポートし、ENDPOINT/SERVERNET エンティティ記述子の GMS、ポート、およびファイバーの各フィールドをサポートしなくなりました。
- L シリーズの RVU で、FABRIC エンティティと Legacy 形式のレコードが両方指定された場合、このプロシージャーは ERR^NO^LEGACY^STYLE を返します。
- MEAS_READACTIVE_MANY を呼び出すアプリケーションは、取得されるレコードと同じプロセッサーで実行している必要があります。このプロシージャーは、プロセス間メッセージを交換しないので、高いパフォーマンスを備えています。複数のプロセッサーからレコードを取得するには、各プロセッサーでアプリケーションのコピーを実行します。

- MEAS_READACTIVE_MANY は、エンティティの標準的なヘッダーおよび ID 情報をすべて返すわけではありません。代わりにプロシージャーは、PIN、LDEV、OCB または ACB 番号やファイル名などの説明情報のサブセットを返します。
- ZMS 形式のレコードの場合、プロシージャーは形式とサブシステムバージョンを返します。
- ZMS 形式のレコードを要求する TEMPLATE^VERSION パラメーターを指定した場合、外部 ZMS *type*.CTRS^DEF テンプレートに応じてカウンター値が返されます。TEMPLATE^VERSION パラメーターを省略した場合、Legacy 形式の外部レコードが返されます。
- 最高のパフォーマンスを得るには、現在の RVU 用の MEASDDLS ファイルからの ZMS `entity-template-version` リテラルを使用します。以前のテンプレートバージョンを使用するか、Legacy 形式のレコードを要求すると、パフォーマンスコストが増大します。
- このインターフェイスを使用するには、アクセスするエンティティを含む測定を最初に開始する必要があります。Measure は、変更したカウンターレコードを識別するために、その測定と、対応する *measnum* を必要とします。
- MEAS_READACTIVE_MANY_は、指定されたエンティティタイプでカウンター ID（CID）のリスト全体を反復することによって、プロセッサーで構成された複数のアクティブなカウンターレコードを返します。反復中または呼び出しの間に CID が発行された場合、エンティティの次の CID へのリンクに従うことはできない可能性があるため、エラー 3402（ERRM^CANNOTACCESS）が発生します。エラー 3402 が発生した後、発行された CID に対応するエンティティの名前は利用できません。エラー 3402 に対する推奨の対応は、MEAS_READACTIVE_MANY_コンテキストをリセットし、呼び出しをもう一度やり直すことです。

MEASREADCONF

データファイルから測定の構成を返します。測定データファイルから読み取る前に、MEASOPEN プロシージャーを使用して、ファイルへの読み取りアクセス権を取得する必要があります。

Measure G09 以降の PVU では、MEASREADCONF はジャーナルセグメント機能の構成を報告する設定を取得するためパラメーターを持ちます。

```
error := MEASREADCONF ( dfnum                ! i
                      ,[ contab ]            ! o
                      ,[ bufsize ]           ! i
                      ,[ bytesret ]          ! o
                      ,[ starttime ]         ! o
                      ,[ stoptime ]          ! o
                      ,[ interval ]          ! o
                      ,[ entities ]          ! o
                      ,[ ctrspace ]          ! o
                      ,[ version ]           ! i
                      ,[ max^ent ]           ! i
                      ,[ settings ] );       ! o
```

contab, bufsize, bytesret

これらのパラメーターはグループです。つまり、他の 2 つと一緒でなければ渡すことができず、エラー 3201（ERR^MISSINGPARAM）が返されます。

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3201	err^missingparam	*dfnum* が省略されたか、*contab*、*bufsize*、または *bytesret* の少なくとも 1 つが渡されませんでした。これらのパラメーターはグループで、他の 2 つと一緒でなければ渡すことはできません。
3203	err^badparams	*bufsize* が 32,000 バイトを超えるか、*max^ent* がゼロまたは負の数値として渡されました。
3204	err^buftoosmall	返されたカウンターレコードがバッファーサイズを超過しました。

dfnum

入力

INT：値

データファイルアクセス番号です。MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。

contab

出力

INT .EXT：参照：*

測定の構成を定義する配列です。*contab* 配列の形式については、**MEASCONFIGURE**(469 ページ)を参照してください。

bufsize

入力

INT：値

ターゲットバッファー *contab* のサイズ（バイト単位）です。*bufsize* が 32,000 バイトよりも大きい場合、エラー 3203（ERR^BADPARAMS）が返されます。*contab* サイズを *bytesret* に返して取得するには、*bufsize* を少なくとも 4 バイト以上にする必要があります。

bytesret

出力

INT：参照：1

contab 配列のバイト単位のサイズ（およびターゲットバッファー *contab* に返されるバイト数）。*bytesret* が *bufsize* より大きい場合はエラー 3204（ERR^BUFTOOSMALL）が返され、*contab* 値は返されません。

starttime

出力

FIXED：参照：1

測定の開始時間です。

stoptime

出力

FIXED：参照：1

測定の停止時間です。-1 の値は停止時間が指定されていないことを示します。

interval

出力

FIXED：参照：1

測定の収集間隔です。-1 の値は収集間隔が指定されていないことを示します。

entities

出力

INT(32)：参照：MAX^T+1

同時に測定された各エンティティタイプのエンティティの最大数を含む配列です。この表に示すように、配列要素はエンティティタイプ順です。値 0 はエンティティタイプが構成に含まれていないことを示します。MEASREADCONF は、現在アクティブな測定の *entities* 配列および *ctrspace* 配列を返しませんが、MEASSTATUS は配列を返します。この配列はゼロベースで宣言する必要があります。次に例を示します。INT(32).ENTITIES[0:MAX^T];

この表は、エンティティタイプとそのリテラル値および数値識別子を示したものです。

エンティティタイプ	リテラル値	数値識別子
CPU	CPU^T	1
PROCESS	PROCESS^T	2
PROCESSH	PROCESSH^T	3
USERDEF	USERDEF^T	4
FILE	FILOP^T	5
DISCOPEN	DFILOP^T	6
DISC	DISC^T	7
DEVICE	IODEV^T	8
LINE	LINE^T	9
NETLINE	NETLINE^T	10
SYSTEM	REMSYS^T	11
CLUSTER	CLUSTER^T	12
TERMINAL	TERM^T	13
TMF	TMF^T	14
SQLPROC	SQLPROC^T	15
SQLSTMT	SQLSTMT^T	16

表は続く

エンティティタイプ	リテラル値	数値識別子
OPDISK	OPDISK^T	17
CONTROLLER	CTRL^T	18
ENDPOINT	ENDPOINT_T	18
ServerNet	SVNET^T	18
DISKFILE	DISKFILE^T	19
OSSCPU	OSSCPU^T	20
OSSNS	OSSNS^T	21
FABRIC	FABRIC_T	22
（最大値）	MAX^T	24

ctrspace

出力

INT(32)：参照：MAX^T+1

各エンティティタイプによって使用されるワード単位の最大カウンタースペースを含む配列です。*entities* について示されているように、配列要素はエンティティタイプ順です。値 0 はエンティティタイプが構成に含まれていないことを示します。MEASREADCONF は、現在アクティブな測定の *entities* 配列および *ctrspace* 配列を返しませんが、MEASSTATUS は配列を返します。この配列をゼロベースで宣言します。次に例を示します。

INT(32).CTRSPACE[0:MAX^T];

version

入力

INT：値

データファイルが作成されたときの Measure 製品バージョンです。値は 2 つの部分で構成されます。ビット 0:7 にはアルファベット文字（たとえば D）が含まれ、ビット 8:15 には数値（たとえば 30）が含まれています。これらを合わせて製品のバージョンを表します（たとえば D30）。パラメーターが渡されていないか 0 として渡された場合、バージョンはデフォルトで、現在インストールされている Measure PVU になります。

max^ent

入力

INT：値

呼び出し元が *entities* 配列および *ctrspace* 配列のために割り当てたエンティティの最大数です。この値は通常、MAX^T として指定する必要があります（MAX^T は、最大エンティティタイプコードを指定する MEASDECS ファイルのリテラル定義です）。MAX^T より小さく指定した場合、返される *entities* または *ctrspace* 配列情報の量を制限（切り捨て）します （たとえば、*max^ent* を 1 にすると戻り値は CPU 情報のみに制限されます）。*max^ent* がゼロまたは負の値として渡された場合は、エラー 3203（ERR^BADPARAMS）が返されます。*max^ent* が省略された場合、デフォルトで MAX^T になります。

settings

出力

INT：参照：1

（Measure G09 以降の PVU）データファイルによって表現される測定の構成属性を識別するフラグの配列です。

settings.0:8	現在は使用されていません。
settings.9	1 = このファイルではカウンターデータレコードは抑制されます。
settings.10	1 = SQL/MX ジャーナルセグメントは現在、このファイルの呼び出し元プロセスに接続されています。
settings.11	1 = このファイルでは、SQL/MX ジャーナルセグメントを作成中です。
settings.12	1 = SQL/MX ジャーナルセグメントはこのファイル内に存在します。
settings.13	1 = OSS ジャーナルセグメントは現在、このファイルの呼び出し元プロセスに接続されています。
settings.14	1 = このファイルでは、OSS ジャーナルセグメントを作成中です。
settings.15	1 = OSS ジャーナルセグメントはこのファイル内に存在します。

使用上の注意

- L シリーズの RVU で作成される測定では、このプロシージャーは FABRIC 記述子をサポートします。また、このような測定では、GMS、ポート、およびファイバーの各フィールドは、ENDPOINT/SERVERNET エンティティ記述子で使用されません。
- MEASREADCONF の呼び出しは、MEASCOM を使用することによる INFO *コマンドの実行と同等です。リターン情報を取得するには、MEASOPEN の呼び出しを以前に実行しており、実行中の MEASFH プロセスに対するオープンを確立している必要があります。MEASREADCONF は、MEASFH プロセスとのプロセス間通信を通じて測定の構成情報を取得します。
- MEASINFO プロシージャーは、データファイルから構成情報を取得する代替手段を提供します。
- MEASREADCONF パラメーターはオプションです。ただし、

  dfnum

  dfnum を省略した場合、エラー 3201（ERR^MISSINGPARAM）が返されます。

  contab*、*bufsize*、*bytesret

  これらのパラメーターはグループです。つまり、他の 2 つと一緒でなければ渡すことができません。そうしない場合、エラー 3201（ERR^MISSINGPARAM）が返されます。
- OSS または SQL/MX ジャーナルセグメントの構築が測定に必要な場合、ステータスは測定がアクティブである間、常に構築中と報告されます。測定を停止すると、ファイルの更新が完了するまで、設定は構築中と報告され続けます。

MEAS_RETRIEVE_TIMERCELLS_

指定されたタイマーセルの値を取得します。このプロシージャーは、Measure の外部のカウンターを維持するためにアプリケーションで細分性の高いタイマーを必要とする場合に使用します。Measure は、USERDEF エンティティに対して定義されたタイマーセルカウンターの値を要求するときに、このプロシージャーを暗黙的に呼び出します。

L シリーズの Measure ではタイマーセル機能は無効です。L シリーズで呼び出されると、この API は ERR^NOT^SUPPORTED エラーを返します。この機能は、**ビジーカウンター**(170 ページ)、**キューカウンター**(172 ページ)、および**キュービジーカウンター**(172 ページ)に置き換えられています。

タイマーセル値を取得しても値はリセットされません。

```
error := MEAS_RETRIEVE_TIMERCELLS_ ( count          ! i
                                   ,indexes        ! i
                                   ,timers );      ! i,o
```

count

入力

INT：値

取得するタイマーセルの数です。Measure は単一ランデブー中に複数のタイマーセル値を取得できるため、個々のタイマーセルを別々に要求する代わりに複数のタイマーセル値を単一の呼び出しで取得する方が合理的です。

indexes

入力

INT .EXT：参照：*count*

取得するタイマーセルのインデックスを含む配列です。

timers

入出力

FIXED .EXT：参照：*count*

タイマーセル値の配列です。入力時は配列は空です。出力では、タイマーセルのインデックスが以前のパラメーターで指定された順序と同じ順序でタイマーセルの値を含みます。タイマーセル値はマイクロ秒単位です。

MEAS_SQL_MAP_INIT_

Measure H02 以降の PVU では、MEAS_SQL_MAP_INIT_は SQL/MX マッピングセッションを開始します。これは次のいずれの API が使用される場合に行う必要があります

```
MEAS_GETDESCINFO_
MEASLISTENAME
MEASLISTGNAME
```

また、指定された Measure データファイル内に SQL ジャーナルがないため、解決要求は SQL/MX サブシステムから入力されます。

```
error := MEAS_SQL_MAP_INIT_ ( connectionInfo );     ! i,o
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3297	err^sqlmx^map^dupconnection	SQL/MX マッピングセッションは、このアプリケーションではすでに確立されています。
3298	err^sqlmx^map^init	SQL/MX マッピングセッションを開始できません。

connectionInfo

入出力

INT(32) .EXT：参照：25

呼び出し元によって少なくとも 100 バイトになるように割り当てられます。

使用上の注意

最も効率的にするには、アプリケーションプロセスにつき 1 回だけ、この API を呼び出す必要があります。

例

これは、MEAS_SQL_MAP_INIT_と MEAS_SQL_MAP_STOP_を使用する方法を示す、pTAL 疑似コードの簡単な例です。

```
int(32)  .connectionInfo[0:24];  ! at least 100 bytes
! at the start of the application
! if (error := MEAS_SQL_MAP_INIT_(connectionInfo)) then
! handle error

! do work including calls to one or more of the APIs that require the
! connectionInfo argument as input unless we are analyzing a data file
! before exiting the application

! if connectionInfo[0] <> 0d then
! MEAS_SQL_MAP_STOP_(connectionInfo);
```

MEAS_SQL_MAP_STOP_

Measure H02 以降の PVU では、MEAS_SQL_MAP_STOP_は SQL/MX マッピングセッションを停止します。これは、MEAS_SQL_MAP_INIT_を呼び出したアプリケーションプロセスが終了する前に実行する必要があります。

```
error := MEAS_SQL_MAP_STOP_ ( connectionInfo );     ! i
```

返すことができる唯一のエラーは 3202（ERR^BOUNDS）です。

connectionInfo

入力

INT(32) .EXT：参照：25

呼び出し元によって少なくとも 100 バイトになるように割り当てられます。

MEAS_SQLNAME_COMPARE_

Measure H02 以降の PVU では、MEAS_SQLNAME_COMPARE_は外部形式の 2 つの完全修飾 ANSI SQL 名を比較します。名前はワイルドカードを含むことができず、同じオブジェクトタイプにする必要があります。

```
status := MEAS_SQLNAME_COMPARE_ ( name1                ! i
                                 ,name1_length         ! i
                                 ,name2                ! i
                                 ,name2_length );      ! i
```

status

INT：参照：1

操作の結果を示すステータスコードです。

-1	*name1* < *name2* (*a.b.c* < *a.bz.c*)
0	*name1* = *name2*
1	*name1* > *name2* (*a.bz.c* > *a.b.c*)
err^badformatsqlname	1 つ以上の指定された ANSI SQL 名に構文エラーが含まれていたか、完全修飾されていませんでした。
err^sql^api^internal	1 つ以上の指定された ANSI SQL 名を構文解析できませんでした。

name1

入力

STRING .EXT：参照：*

外部形式の入力される名前 1 を含むバッファーです。名前空間キーワードが必要です。

name1_length

入力

INT：値

入力される名前 1 のサイズ（バイト単位）です。

name2

入力

STRING . EXT：参照：*

外部形式の入力される名前 2 を含むバッファーです。名前空間キーワードが必要です。

name2_length

入力

INT：値

入力される名前 2 のサイズ（バイト単位）です。

使用上の注意

- ANSI SQL オブジェクトには独立した名前空間があり、同じ名前が別の名前空間内に存在する場合、名前の各インスタンスは別のオブジェクトを示します。
- Measure で使用される名前空間は TABLE、INDEX、および MODULE です。TABLE は、CATALOG、SCHEMA、および TABLE のキーワードを使用して暗黙指定されます。INDEX は、INDEX のキーワードを使用して暗黙指定されます。MODULE は、SQLSTMT に対して暗黙指定されます（MEASCOM には MODULE キーワードがありません）。
- MEASSQLNAME_* API を呼び出すときに、入力 ANSI SQL 名の前に名前空間キーワードを付ける必要があります。

MEAS_SQLNAME_RESOLVE_

Measure H02 以降の PVU では、MEAS_SQLNAME_RESOLVE_は ANSI SQL の名前の部分を組み合わせて正規化された外部形式で完全修飾名を作成します。このルーチンはデフォルトの CATALOG と SCHEMA の設定を解決し、正規化された外部形式で ANSI SQL の完全修飾名を返します。

```
error := MEAS_SQLNAME_RESOLVE_ ( result_name                    ! o
                               ,result_name_max                 ! i
                               ,result_name_length              ! o
                               ,name                            ! i
                               ,name_length                     ! i
                               ,[ def_catalog ]                 ! i
                               ,[ def_catalog_length ]          ! i
                               ,[ def_schema ]                  ! i
                               ,[ def_schema_length ] ) ;       ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3204	err^buftoosmall	*result_name_max* は小さすぎるため、正規化された外部形式の完全修飾名を保持できません。指定された場合、必要なサイズが *result_name_length* に返されます。
3239	err^badformatsqlname	1 つ以上の指定された ANSI SQL 名に構文エラーが含まれていました。
3295	err^sql^api^internal	指定された ANSI SQL 名を構文解析できませんでした。

result_name

出力

STRING .EXT：参照：*

正規化された外部形式で返された完全修飾名を含むバッファーです。

result_name_max

入力

INT：値

result_name バッファーのサイズ（バイト単位）です。

result_name_length

出力

INT：参照：1

result_name バッファーで返される完全修飾名のサイズ（バイト単位）です。エラー 3240（ERR^BUFTOOSMALL）が返された場合、必要なサイズが表示されます。

name

入力

STRING .EXT：参照：*

完全修飾名に拡張される外部形式で入力される名前を含むバッファーです。*name* にカタログやスキーマの名前が含まれていない場合、デフォルトのカタログ（*def_catalog*）やデフォルトのスキーマ（*def_schema*）の名前が使用されます。そのとき、必要な *def_catalog* や *def_schema* の名前が欠落している場合はエラーが返されます。ワイルドカードを使用できます。名前空間キーワードが必要です。

name_length

入力

INT：値

入力される名前のサイズ（バイト単位）です。

def_catalog

入力

STRING .EXT：参照：*

入力される *name* からカタログフィールドが省略されている場合に使用されるデフォルトのカタログ名を格納するバッファーで、その場合は指定される必要があります。

def_catalog_length

入力

INT：値

def_catalog パラメーターで渡されるデフォルトカタログ名のサイズ（バイト単位）です。

def_schema

入力

STRING .EXT：参照：*

入力される *name* からスキーマフィールドが省略されている場合に使用されるデフォルトのスキーマ名を格納するバッファーで、その場合は指定される必要があります。

def_schema_length

入力

INT：値

def_schema パラメーターで渡されるデフォルトスキーマ名のサイズ（バイト単位）です。

使用上の注意

- ANSI SQL オブジェクトには独立した名前空間があり、同じ名前が別の名前空間内に存在する場合、名前の各インスタンスは別のオブジェクトを示します。
- Measure で使用される名前空間は TABLE、INDEX、および MODULE です。TABLE は、CATALOG、SCHEMA、および TABLE のキーワードを使用して暗黙指定されます。INDEX は、INDEX のキーワードを使用して暗黙指定されます。MODULE は、SQLSTMT に対して暗黙指定されます（MEASCOM には MODULE キーワードがありません）。
- MEASSQLNAME_* API を呼び出すときに、入力 ANSI SQL 名の前に名前空間キーワードを付ける必要があります。

MEAS_SQLNAME_SCAN_

Measure H02 以降の PVU では、MEAS_SQLNAME_SCAN_は、ワイルドカードで表される可能性がある完全修飾された外部形式の ANSI SQL 名を構文解析します。このルーチンは、ANSI SQL 名の構文を確認します。

```
error := MEAS_SQLNAME_SCAN_ ( name                ! i
                            ,name_length );       ! i
```

error

INT

操作の結果を示すエラーコードです。エラーコードは、次のとおりです。

エラーコード		説明
0		正常終了。
3239	err^badformatsqlname	指定された ANSI SQL 名に構文エラーが含まれていました。
3295	err^sql^api^internal	指定された ANSI SQL 名を構文解析できませんでした。

name

入力

STRING .EXT：参照：*

スキャンされる外部形式の入力される名前を含むバッファーです。名前空間キーワードが必要です。

name_length

入力

INT：値

入力される名前のサイズ（バイト単位）です。

使用上の注意

- ANSI SQL オブジェクトには独立した名前空間があり、同じ名前が別の名前空間内に存在する場合、名前の各インスタンスは別のオブジェクトを示します。
- Measure で使用される名前空間は TABLE、INDEX、および MODULE です。TABLE は、CATALOG、SCHEMA、および TABLE のキーワードを使用して暗黙指定されます。INDEX は、INDEX のキーワー

ドを使用して暗黙指定されます。MODULE は、SQLSTMT に対して暗黙指定されます（MEASCOM には MODULE キーワードがありません）。

- MEASSQLNAME_* API を呼び出すときに、入力 ANSI SQL 名の前に名前空間キーワードを付ける必要があります。

例

これらは MEAS_SQLNAME_SCAN_を呼び出す pTAL 疑似コードの簡単な例です。

```
STRING .EXT inbuf[0:1023] :=["table cat.sch.t"];

if (error := MEAS_SQLNAME_SCAN_ (inbuf,
                                 15))
  ! handle error
else
  ! ANSI SQL TABLE name is OK


STRING .EXT inbuf[0:1023] :=["module cat.sch.m"];

if (error := MEAS_SQLNAME_SCAN_ (inbuf,
                                 16))
  ! handle error
else
  ! ANSI SQL MODULE name is OK
```

MEASSTATUS

現在アクティブな測定に関する情報を返します

Measure G09 以降の PVU では、MEASSTATUS はジャーナルセグメント機能の構成を報告する設定を取得するためのパラメーターを持ちます。

```
error := MEASSTATUS ( meascb              ! i,o
                     ,measnum             ! i
                     ,cpus                ! o
                     ,starttime           ! o
                     ,stoptime            ! o
                     ,interval            ! o
                     ,entities            ! o
                     ,ctrspace            ! o
                     ,[ max^ent ]         ! i
                     ,[ settings ] );     ! o
```

meascb

入出力

INT：参照：$LEN(MEASCB^DEF) / 2

Measure サブシステムが後のプロシージャーコールのためにデータを格納する制御ブロックです。*meascb* を使用する最初の Measure プロシージャーを呼び出す前に、グローバルデータ領域内に制御ブロック用のスペースを割り当て、制御ブロックの各要素を-1 に初期化する必要があります。*meascb* を Measure プロシージャーに渡した後は、内容を変更しないでください。

$SYSTEM.SYS*nn*.MEASDECS ファイルには、制御ブロック記述子の構造定義（MEASCB^DEF）が含まれています。

measnum

入力

INT：値

測定番号です。MEASCONFIGURE プロシージャーによって返される *measnum* 値を使用します。

cpus

出力

INT：参照：1

測定されている CPU を示します。各ビットは CPU 番号を表します。ビットが 1 に設定されている場合、その CPU は測定されています。設定されていない場合、その CPU は測定されていません。

starttime

出力

FIXED：参照：1

測定の開始時間です。

stoptime

出力

FIXED：参照：1

測定の停止時間です。

interval

出力

FIXED：参照：1

測定の収集間隔です。

entities

出力

INT(32)：参照：MAX^T+1

現在測定されている各エンティティタイプのエンティティの数を含む配列です。この表に示すように、配列要素はエンティティタイプ順です。値 0 はエンティティタイプが構成に含まれていないことを示します。

エンティティタイプ	リテラル値	数値識別子
CPU	CPU^T	1
PROCESS	PROCESS^T	2
PROCESSH	PROCESSH^T	3
USERDEF	USERDEF^T	4
FILE	FILOP^T	5
DISCOPEN	DFILOP^T	6
DISC	DISC^T	7
DEVICE	IODEV^T	8

表は続く

エンティティタイプ	リテラル値	数値識別子
LINE	LINE^T	9
NETLINE	NETLINE^T	10
SYSTEM	REMSYS^T	11
CLUSTER	CLUSTER^T	12
TERMINAL	TERM^T	13
TMF	TMF^T	14
SQLPROC	SQLPROC^T	15
SQLSTMT	SQLSTMT^T	16
OPDISK	OPDISK^T	17
CONTROLLER	CTRL^T	18
ENDPOINT	ENDPOINT_T	18
ServerNet	SVNET^T	18
DISKFILE	DISKFILE^T	19
OSSCPU	OSSCPU^T	20
OSSNS	OSSNS^T	21
FABRIC	FABRIC_T	22
（最大値）	MAX^T	24

ctrspace

出力

INT(32)：参照：MAX^T+1

各エンティティタイプによって現在使用中の、ワード単位のカウンタースペースを含む配列です。*entities* について示されているように、配列要素はエンティティタイプ順です。値 0 はエンティティタイプが構成に含まれていないことを示します。

max^ent

入力

INT：値

呼び出し元が *entities* 配列および *ctrspace* 配列のために割り当てたエンティティの最大数です。

settings

出力

INT：参照：1

（Measure G09 以降の PVU）データファイルによって表現される測定の構成属性を識別するフラグの配列です。

settings.0:8	現在は使用されていません。
settings.9	1 = このファイルではカウンターデータレコードは抑制されます。
settings.10:11	現在は使用されていません。
settings.12	1 = このファイルでは、SQL/MX ジャーナルセグメントを作成中です。
settings.13	1 = SQL/MX ジャーナルセグメントはこのファイル内に存在します。
settings.14	1 = このファイルでは、OSS ジャーナルセグメントを作成中です。
settings.15	1 = OSS ジャーナルセグメントはこのファイル内に存在します。

使用上の注意

OSS または SQL/MX ジャーナルセグメントの構築が測定に必要な場合、ステータスは測定がアクティブである間、常に構築中と報告されます。測定を停止すると、ファイルの更新が完了するまで、設定は構築中と報告され続けます。

MEASWRITE_DIFF_

測定データファイルから 1 つ以上のカウンターレコードのウィンドウを読み取り、結果を構成化ファイルに配置します。

測定データファイルから読み取る前に、MEASOPEN プロシージャーを使用して、ファイルへの読み取りアクセス権を取得する必要があります。データファイルは現在アクティブな測定に関連付けることができる、または非アクティブな測定からのデータを含めることができます。

構造化ファイルの名前付けに使用される規則は MEASCOM の規則と同じです。プロシージャーでは指定されたエンティティ名を、プロシージャーコール内で指定されるボリュームおよびサブボリュームのファイル名として使用します。ファイルがすでに存在する場合は、取得した情報がファイルに追加されます。たとえば、CPU 情報を$USER.MEAS に配置することが呼び出しで要求される場合、この情報は $USER.MEAS.CPU に配置されます。

プロシージャーコールは開始時刻と終了時刻の両方を指定します。返されるレコードは、その間隔内で収集された統計情報を提供します。間隔レコード、概要レコード、またはその両方を要求することができます。

プロシージャーコールは、追加の処理を取得するためのオプション（*all^recs*、*zero^reports*、*totals*、*loadid*、および *if^item*）を指定できます。

Measure G09 以降の PVU では、パラメーター if^item には、長い（128 バイトの）SQL/MX 実行単位名の使用可能なフィールドテンプレート if^item^g09^def があります。このテンプレートは G09 以前の if^item^def テンプレートと共存するため、既存のアプリケーションと古いデータファイルとの互換性が維持されます。

```
error := MEASWRITE_DIFF_( dfnum                    ! i
                        ,entitydesc                ! i
                        ,volume^subvol             ! i
                        ,[ from^time ]             ! i
                        ,[ to^time ]               ! i
                        ,[ timetol ]               ! i
                        ,[ version ]               ! i
                        ,[ all^recs ]              ! i
```

```
,[ zero^reports ]           ! i
,[ totals ]                 ! i
,[ loadid ]                 ! i
,[ if^item ]                ! i
,[ templateversion ] );     ! i
```

dfnum

入力

INT：値

データファイルアクセス番号です。MEASOPEN プロシージャーによって返される *dfnum* 値を使用します。

entitydesc

入力

INT .EXT：参照：*

目的のカウンターレコードのエンティティタイプおよびエンティティ仕様です。MEASCONFIGURE プロシージャーの *contab* の説明にリストされている記述子の 1 つとして *entitydesc* を指定します。複数の仕様のテンプレートとして *entitydesc* を指定した場合、MEASWRITE_DIFF_は、テンプレートに適合するすべてのカウンターレコードを返します。

（Measure H01 以降の PVU）ANSI SQL オブジェクトまたはパーティションの場合、*entitydesc* は DISCOPEN、DISKFILE、FILE、または SQLSTMT エンティティ記述子にできます。SQLSTMT ANSI エンティティ記述子は、SQLSTMT カウンターレコードを取得するために ANSI SQL 名を指定する場合に使用されます。

Measure H07 以降の PVU では、MEASWRITE_DIFF_は FABRIC エンティティ記述子を受け入れます。

volume^subvol

入力

INT .EXT：参照：8

カウンターレコードのターゲットボリュームとサブボリュームです。カウンターレコードは、DDL レコード形式でファイルに書き込まれます。エンティティタイプごとの DDL レコード定義については、**エンティティおよびカウンター**(167 ページ)を参照してください。

ローカルシステムの場合、*volume^subvol* の形式（string[0:15]で再定義する場合）は次のとおりです。

```
Volume^SubVol[0]   = '$'
Volume^SubVol[1:7] = volume name (padded with spaces)
Volume^SubVol[8:15]= subvolume name (padded with spaces)
```

リモートシステムでは、形式は次のとおりです。

```
Volume^SubVol[0]   = '\'
Volume^SubVol[1]   = system number
Volume^SubVol[2:7] = volume name (padded with spaces)
Volume^SubVol[8:15]= subvolume name (padded with spaces)
```

from^time

入力

FIXED：値

測定ウィンドウの開始時間です。*from^time* は、システムプロシージャー JULIANTIMESTAMP によって提供された 4 ワード/1 マイクロ秒形式の地方平均時に基づくユリウス日として指定します。（*Guardian プロシージャーコールリファレンスマニュアル*を参照してください）。値-1 は測定の開始を示します。*from^time* を指定しない場合、デフォルトは-1 です。

to^time

入力

FIXED：値

測定ウィンドウの停止時間です。*to^time* は、システムプロシージャー JULIANTIMESTAMP によって提供された 4 ワード/1 マイクロ秒形式の地方平均時に基づくユリウス日として指定します（*Guardian プロシージャーコールリファレンスマニュアル*を参照してください）。-1 の値は測定の終了を示します。*to^time* を指定しない場合、デフォルトは-1 です。

timetol

入力

FIXED：値

from^time と *to^time* で囲まれた時間枠に適用される許容値で、実質的に、*from^time* から *timetol* を引いた時間から、*to^time* に同じ量を足した時間までの時間ウィンドウになります。-1 の値を指定すると、*from^time* よりも前の最後のレコードと、*to^time* よりも後の最も早いレコードが含まれるように時間枠が広げられます。*timetol* を指定しない場合、デフォルトは-1 です。

version

入力

INT：値

データファイルが作成されたときの Measure 製品バージョンです。値は 2 つの部分で構成されます。ビット 0:7 にはアルファベット文字（たとえば D）が含まれ、ビット 8:15 には数値（たとえば 30）が含まれています。これらを合わせて製品のバージョンを表します（たとえば D30）。

all^recs

入力

INT：値

値 1 が渡された場合は、指定されたウィンドウ内のすべての間隔レコードが要求されます。省略するか 0 が指定された場合、概要レコードのみが返されます。

zero^reports

入力

INT：値

値 0 が渡された場合は、ゼロレコードの除外を要求します。省略するか 1 が指定された場合、ゼロレコードが返されます。

totals

入力

INT：値

0、1、または 2 の値で、それぞれ次を示します。

0	（デフォルト）合計レコードが省略され、個々のカウンターレコードが含まれます。
1	個々のレコードと合計レコードが含まれます。
2	個々のカウンターレコードが省略され、合計レコードが含まれます。

loadid

入力

INT .EXT：参照：4

返されるすべてのレコードに含まれる 8 バイトの文字列です。*loadid* を指定しない場合、デフォルトはブランク（スペース）です。

if^item

入力

INT .EXT：参照：$LEN(IF^ITEM^G09^DEF) / 2

Measure G09 以降の PVU で、長い（128 バイトの）SQL/MX 実行単位名をサポートしている構造です。

```
STRUCT IF^ITEM^G09^DEF (*);
Begin
Int     Item^Name[0:15];  ! Item name in entity record
Int     Relation^Operator;! Relation Operator
                          ! MEASWRITE_DIFF_の使用上の注意 (544 ページ) を参照し
てください。
                          ! Bit 0 must be 1
                          !
                          ! Bits 13:15
                          !   1 -> Equal        =
                          !   2 -> Not Equal    <>
                          !   3 -> Less Than    <
                          !   4 -> Greater Than >
Fixed   Value;            ! Value used in comparison
Int     Rate;             ! Use rated or nonrated value
                          ! for condition checking:
                          !   0 -> No-Rate, 1 -> Rated
Int     SQLName[0:63];    ! For SQLSTMTs only
Int     SQLIndex;         ! SQL Index number
String  IP^addr[0:15] = SQLName;
Int     port = SQLIndex;
Int     IP^wildcard^flags = value;
End;

DEFINE  IP^addr^wildcard^flag = IP^wildcard^flags.<0>#,
        port^wildcard^flag = IP^wildcard^flags.<1>#;
```

または

if^item

入力

INT .EXT：参照：$LEN(IF^ITEM^DEF) / 2

既存のアプリケーションと G09 以前の Measure PVU での古いデータファイルとの互換性を提供する構造です。

```
STRUCT IF^ITEM^DEF (*);
Begin
Int     Item^Name[0:15];  ! Item name in entity record
Int     Relation^Operator;! Relation Operator
                          !   1 -> Equal        =
                          !   2 -> Not Equal    <>
                          !   3 -> Less Than    <
                          !   4 -> Greater Than >
Int(32) Value;            ! Value used in comparison
```

```
Int     Rate;              ! Use rated or nonrated value
                           ! for condition checking:
                           !   0 -> No-Rate, 1 -> Rated
Int     SQLName[0:15];     ! SQL run-unit in Ascii
Int     SQLIndex;          ! SQL Index number
End;
```

item^name

入力

エンティティレコードで定義されたカウンター名のいずれかです（たとえば、CPU レコードの場合は CPU-BUSY-TIME）。

relation^operator

入力

比較タイプを指定する 1 から 4 までの範囲の値を示します。

1	等しいかどうかを比較します。
2	等しくないかどうかを比較します。
3	「小なり」かどうかを比較します。
4	「大なり」かどうかを比較します。

value

入力

特定のカウンターの値との比較のために使用する 32 ビットの整数です。*value* = *n**1000。たとえば、17.09 は 17090 として保存されます。

rate

入力

非定格値または定格値のどちらを比較に使用するかを決定します。

- 0 は非定格値を示します（REPORT RATE OFF に類似します）。
- 1 は定格値を示します（REPORT RATE ON に類似します）。

sqlname

入力

SQL 実行単位を指定する 16 ワードの配列です。*sqlindex* はインデックス番号を指定します。これらの 2 つのフィールドが、SQLSTMT エンティティと組み合わせて使用されます。渡された場合、MEASWRITE_DIFF_は指定された実行単位とインデックス番号を含むレコードを返します。

sqlindex

入力

SQL インデックスです。値-1 はすべての SQL インデックスを指定します。

注記:

if^item^def 構造が渡された場合、最初に初期化する必要があります。値を持たない構造を渡す場合、IF^ITEM^DEF.Item^Name を 0 に初期化する必要があります。同様に、IF^ITEM^DEF.SQLName の値が 0 の場合、SQL（*run-unit,index*）が指定されないことを意味します。

ip^addr

入力

OSS AF_INET ソケットに提供される追加の識別子です。

ip^addr は、AF_INET ソケットの FILE レコードヘッダーの一部であるアドレス/ポートの組み合わせの IP アドレス部分です。識別子は、アドレスのスペースで区切られたポートで報告されます（たとえば、"123.45.67.8 90"）。

port

入力

OSS AF_INET ソケットに提供される追加の識別子です。

port は、AF_INET ソケットの FILE レコードヘッダーの一部である IP アドレス/ポートの組み合わせのポート部分です。

ip^addr^wildcard^flags

入力

OSS AF_INET ソケットに提供される追加の識別子です。

ip^addr^wildcard^flags は *ip^addr* フィールドと *port* フィールドのワイルドカードサポートを判別します。

< 0 >	0	指定されたアドレスと一致する。
	1	任意のアドレスと一致する。
< 1 >	0	指定されたポート番号と一致する。
	1	任意のポート番号と一致する。

データレコード内の IP アドレスは、IP コンポーネントごとに 1 バイトの 16 進数で表示されます。ここでは、さまざまな *ip^addr*、*port*、および *ip^wildcard^flags* フィールド指定のいくつかの例を示します。

例 1：IP アドレス = 11.22.33.44、任意のポート番号と一致

```
if^item
 = {....
    IP^ADDR = {0xb, 0x16, 0x21, 0x2c, 0x0 <repeats 2 times>},
    PORT = 0x0,
    IP^WILDCARD^FLAGS = 0x4000
    }
```

例 2：IP アドレス = 任意の IP アドレス番号と一致、ポート 37

```
if^item
 = {....
    IP^ADDR = {0x0 <repeats 16 times>},
    PORT = 0x25,
    IP^WILDCARD^FLAGS = 0x8000
    }
```

例 3：IP アドレス = 11.22.33.44、ポート 37

```
if^item
 = {....
    IP^ADDR = {0xb, 0x16, 0x21, 0x2c, 0x0 <repeats 12 times>},
```

```
    PORT = 0x25,
    IP^WILDCARD^FLAGS = 0x0
    }
```

例 4：IPv6 アドレス = 1070::800:2006:4178、ポート 37

```
if^item
 = {....
    IP^ADDR = IP^ADDR = {0x10, 0x70, 0x0, 0x0, 0x0, 0x0, 0x0,
    0x0, 0x0, 0x0, 0x8, 0x0, 0x20, 0x6, 0x41, 0x78},
    PORT = 0x25
    IP^WILDCARD^FLAGS = 0x0
    }
```

templateversion

入力

FIXED：値

（Measure G11 以降の PVU）ZMS 形式の外部レコードが必要な場合、*entitydesc* で要求されるエンティティタイプの MEASDDLS からの適切なテンプレートバージョンのリテラルです。省略されるか 0F として渡される場合、Legacy 形式のレコードが返されます。

-1F として渡された場合、H06.15/J06.04 RVU 以降では、現在のリリースの *templateversion* が使用されます。このような場合、返される外部レコードは、アプリケーションがコンパイルされたときに使用したカウンターレコード定義と一致しないことがあります。

処理可能な形式に対応する外部レコードの取得については、**TEMPLATE-VERSION**(180 ページ)を参照してください。

MEASWRITE_DIFF_の使用上の注意

- L シリーズの RVU で作成される測定では、このプロシージャーは FABRIC 記述子をサポートします。このような測定では、GMS フィールド、ポートフィールド、およびファイバーフィールドは、SERVERNET エンティティ記述子で使用されません。
- L シリーズ RVU で、FABRIC エンティティとレガシー型のレコードが両方指定された場合、このプロシージャーは ERR^NO^LEGACY^STYLE を返します。
- さらに長い IF^ITEM^G09^DEF テンプレートが使用された場合、G09^format を設定する必要があります（DEFINE の例を参照してください）。G09^format が設定されない場合、*sqlname* フィールドの一部は *sqlindex* として解釈され、返されるすべてのレコードが正しくない場合があります。

```
DEFINE G09^format       = relation^operator.0#;
DEFINE G09^rel^operator = relation^operator.13:15#;
```

- Measure G09 データファイルにアクセスするときは、いずれかのテンプレートの形式を使用できます。実際の *sqlname* 値が 32 文字を超過しない場合、Measure G08 以前のデータファイルにアクセスするときに長いテンプレートを使用することもできます。
- Measure H01 以降の PVU では、MEASWRITE_DIFF_は ANSI SQL オブジェクトまたはパーティションの DISCOPEN、DISKFILE、FILE、および SQLSTMT エンティティ記述子を受け入れます。
- Measure H04/J02 以降の PVU は、測定データファイルのサイズに応じて形式 1 または形式 2 の構造化ファイルを作成します。

エンティティ記述子

この章では、Measure のエンティティ記述子について説明します。

コマンドインターフェイスで使用されるエンティティの指定と同様に、エンティティ記述子はプログラムインターフェイスで使用されるエンティティまたはエンティティのセットを記述します。エンティティ記述子を使用して、測定するエンティティを定義したり、エンティティごとに維持するカウンターレコードを指定したりします。

エンティティ記述子は、測定を開始する前に MEASCONFIGURE プロシージャーに渡す構成テーブル（*contab*）配列の一部です。構成テーブルの本体はエンティティ記述子の配列です。エンティティ記述子は、エンティティタイプに従ってセクションに分類され、セクションは数値識別子によって順序付けられます。最初にすべての CPU 記述子（タイプ番号 1）が置かれ、次にすべての PROCESS 記述子（タイプ番号 2）、その次にすべて PROCESSH 記述子（タイプ番号 3）、以下同様となります。

次の表に、MEASCONFIGURE プロシージャーの *contab* 配列で使用できるエンティティ記述子セクションを示します。この表には、エンティティ記述子セクションごとに、MEASDECS ファイル内の関連する構造体の名前、リテラル、および数値識別子が含まれています。エンティティ記述子セクションは、この表に示されている順序で配列に指定する必要があります。

この表の後に、エンティティ記述子の詳細な定義が示されています。これらの定義はこの章ではアルファベット順に示されていますが、以下の例外があることに注意してください。

- エンティティ記述子セクションの一般的な記述子は、より具体的な記述子の前に示されています。（たとえば、DEVICE^DESC は DEVICE^CLIM^DESC の前に示されています。）
- OSS 記述子は ANSI 記述子の前に示されています。（たとえば、FILE^OPEN^OSS^DESC および FILE^OPEN^OSS^DESC^D10 は FILE^OPEN^ANSI^DESC の前に示されています。）
- CODE^SPACE^DESC、CODE^SPACE^OSS^DESC、および COUNTER^DESC は、それらが使用できる最初のエンティティ記述子の次に示されます（そのため、CODE^SPACE^DESC は PROCESSH^DESC の次に示され、CODE^SPACE^OSS^DESC は PROCESSH^OSS^DESC の次に示され、COUNTER^DESC は USERDEF^DESC の次に示されます）。

エンティティ記述子を指定する方法の概要については、**Measure プロシージャーの概要**(460 ページ)および **MEASCONFIGURE**(469 ページ)を参照してください。

表 6: エンティティ記述子とタイプ値

セクション	記述子テンプレート構造体（D シリーズの RVU を実行しているシステム）	記述子テンプレート構造体（G シリーズ以降の RVU を実行しているシステム）	タイプリテラル	数値識別子
CPU	**CPU^DESC**	**CPU^DESC**	CPU^T	1
PROCESS	**PROCESS^DESC**	**PROCESS^DESC** **PROCESS^OSS^DESC**	PROCESS^T	2
PROCESSH	**PROCESSH^DESC**	**PROCESSH^DESC** **CODE^SPACE^DESC** **PROCESSH^OSS^DESC** **CODE^SPACE^OSS^DESC**	PROCESSH^T	3

表は続く

セクション	記述子テンプレート構造体（D シリーズの RVU を実行しているシステム）	記述子テンプレート構造体（G シリーズ以降の RVU を実行しているシステム）	タイプリテラル	数値識別子
USERDEF	**USERDEF^DESC** **COUNTER^DESC**	**USERDEF^DESC** **COUNTER^DESC** **USERDEF^OSS^DESC**	USERDEF^T	4
FILE	**FILE^OPEN^DESC**	**FILE^OPEN^DESC** **FILE^OPEN^DESC^D10** **FILE^OPEN^OSS^DESC** **FILE^OPEN^OSS^DESC^D10** **FILE^OPEN^ANSI^DESC**	FILOP^T	5
DISCOPEN	**FILE^OPEN^DESC**	**FILE^OPEN^DESC** **FILE^OPEN^OSS^DESC** **FILE^OPEN^ANSI^DESC**	DFILOP^T	6
DISC	**DEVICE^DESC**	**DEVICE^CLIM^DESC** **DEVICE^SVNET^DESC** **DEVICE^SVNET^DESC^G05**	DISC^T	7
DEVICE	**DEVICE^DESC**	**DEVICE^CLIM^DESC** **DEVICE^SVNET^DESC** **DEVICE^SVNET^DESC^G05**	IODEV^T	8
LINE	**DEVICE^DESC**	**WAN^DESC**	LINE^T	9
NETLINE	**DEVICE^DESC**	**WAN^DESC**	NETLINE^T	10
SYSTEM	**SYSTEM^DESC**	**SYSTEM^DESC**	REMSYS^T	11
CLUSTER	**SYSTEM^DESC**	該当なし	CLUSTER^T	12
TERMINAL	**DEVICE^DESC**	**WAN^DESC**	TERM^T	13
TMF	**CPU^DESC**	**CPU^DESC**	TMF^T	14
SQLPROC	**SQLPROC^DESC**	**SQLPROC^DESC** **SQLPROC^OSS^DESC**	SQLPROC^T	15

表は続く

セクション	記述子テンプレート構造体（D シリーズの RVU を実行しているシステム）	記述子テンプレート構造体（G シリーズ以降の RVU を実行しているシステム）	タイプリテラル	数値識別子
SQLSTMT	**SQLSTMT^DESC**	**SQLSTMT^DESC** **SQLSTMT^OSS^DESC** **SQLSTMT^ANSI^DESC**	SQLSTMT^T	16
OPDISK	**OPDISK^DESC**	該当なし	OPDISK^T	17
CONTROLLER	**CTRL^DESC**	該当なし	CTRL^T	18
ENDPOINT	該当なし	**ENDPOINT^DESC**	ENDPOINT_T	18
SERVERNET	該当なし	**SVNET^DESC**	SVNET^T	18
DISKFILE	**DISKFILE^DESC**	**DISKFILE^DESC** **DISKFILE^OSS^DESC** **DISKFILE^ANSI^DESC**	DISKFILE^T	19
OSSCPU	該当なし	**CPU^DESC**	OSSCPU^T	20
OSSNS	該当なし	**PROCESS^DESC**	OSSNS^T	21
FABRIC	該当なし	**FABRIC_DESC**	FABRIC_T	22

注記:

記述子がすべてのエンティティを指定する場合は、D シリーズの記述子が含まれている測定アプリケーションを使用して、G シリーズのエンティティを測定できます。たとえば、アプリケーションが DEVICE^DESC 記述子を使用していて、すべての識別子フィールドに-1 またはアスタリスクが含まれている場合は、そのアプリケーションを使用して G シリーズの DEVICE エンティティを測定できます。アプリケーションを変更して DEVICE^SVNET^DESC または DEVICE^SVNET^DESC^G05 記述子を使用する必要はありません。

CPU^DESC

CPU、TMF、または OSSCPU エンティティの記述子です。

```
Struct CPU^desc (*)
  Begin
  Int    Type;          ! Entity type
  Int    Len;           ! Byte length of this record
  Int    CPU^number;
  End;
```

type

INT：値：1

次のいずれかです。

エンティティタイプ	リテラル	数値識別子
CPU	CPU^T	1
TMF	TMF^T	14
OSSCPU	OSSCPU^T	20

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。

cpu^number

INT：値：1

測定対象の CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

CTRL^DESC

D シリーズの RVU を実行しているシステム上の CONTROLLER エンティティの記述子です。

```
Struct  Ctrl^desc (*)     -- controller descriptor structure
  BEGIN
  Int    Type;            -- controller entity type
  Int    Len;             -- byte length of the descriptor
  Int    CPU^number;      -- CPU number
  Int    Channel;         -- channel number
  Int    Ctrl;            -- controller number
  Int    Ctrl^type;       -- controller type
  End;
```

type

INT：値：1

リテラル値 CTRL^T または数値識別子 18 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。

cpu^number

INT：値：1

コントローラーが構成されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

channel

INT：値：1

測定対象のコントローラーのチャネル番号です。

ctrl

INT：値：1

コントローラー番号（0～31）です。すべてのコントローラーを指定するには、リテラル-1 を使用します。

ctrl-type

INT：値：1

コントローラータイプ（Hewlett Packard Enterprise の製品番号、3129 など）です。

DEVICE^DESC

D シリーズの RVU を実行しているシステム上の DEVICE、DISC、LINE、NETLINE、または TERMINAL エンティティの記述子です。

```
Struct Device^desc (*)
  Begin
  Int    Type;       ! Entity type
  Int    Len;        ! Byte length of this record
  Int    CPU^number;
  Int    Ctl;        ! Controller number
  Int    Unit;       ! Unit number
  Int    Device^name[0:11];
  Int    Channel;    ! Channel number
  End;
```

注記:

LINE、NETLINE、および TERMINAL エンティティは、G シリーズの RVU では WAN^DESC 記述子を使用します。

type

INT：値：1

次のいずれかです。

エンティティタイプ	リテラル	数値識別子
DISC	DISC^T	7
DEVICE	IODEV^T	8
LINE	LINE^T	9
NETLINE	NETLINE^T	10
TERMINAL	TERM^T	13

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

36（この記述子のバイト単位の長さ）です。

cpu^number

INT：値：1

測定対象のデバイスが実行されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

ctl

INT：値：1

測定対象のデバイスのコントローラー番号（0～31）です。すべてのコントローラーを指定するには、リテラル-1 を使用します。

unit

INT：値：1

測定対象のデバイスのユニット番号です。すべてのユニットを指定するには、リテラル-1 を使用します。

device^name

INT：値：12

測定対象のデバイスの名前です。名前は左詰めにして空白で埋める必要があり、ドル記号（$）で始める必要があります。すべてのデバイスを指定するには、ドル記号とアスタリスク（$*）を使用します。

channel-num

INT：値：1

測定対象のデバイスのチャネル番号です。

DEVICE^CLIM^DESC

H シリーズまたは J シリーズの RVU を実行しているシステムに接続されている DEVICE または DISC エンティティの記述子です。DEVICE^CLIM^DESC には PLPT フィールドが含まれ、これは、CLIM によって接続されているデバイスを一意に識別するフィールドが含まれる構造体です。DEVICE^SVNET^DESC または DEVICE^SVNET^DESC^G05 を使用して、CLIM 以外の NonStop S シリーズのデバイスを記述できます。

```
Struct Device^CLIM^Desc (*)
  Begin
  Int    Type;                ! Entity type
  Int    Len;                 ! Byte length of this record
  Int    CPU^number;
  Int    Fabric;
  Int    ServerNET = Fabric;
  Struct GMS;
    Begin
      Int(32)  Group;         ! Group number
      Int(32)  Module;        ! Module number
      Int(32)  Slot;          ! Slot number
    End;
  Int    Device^name[0:11];
  Struct plpt (plpt^def);
  Int    Config^name[0:31];   ! OSCONFIG name for drive,adaptor,or SAC
  String Config^name^s[0:63]  = Config^name;
  End;
```

type

INT：値：1

次のいずれかです。

エンティティタイプ	リテラル	数値識別子
DISC	DISC^T	7
DEVICE	IODEV^T	8

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は、記述子に構成名が含まれていない場合は 52、記述子に構成名が含まれている場合は 116 です。

cpu^number

INT：値：1

デバイスが構成されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

fabric

INT：値：1

サポートされている物理デバイスへの特定のパスまたはパスのセットを識別するファブリックです。X ファブリックを指定するには 0、Y ファブリックを指定するには 1、または両方のファブリックを指定するにはリテラル-1 を使用します。

servernet

fabric フィールドの代替名です。

GMS

物理的位置のアドレス（グループ、モジュール、スロット）です。GMS は CLIM デバイス（DISC および DEVICE エンティティ）には使用されません。GMS 構造体は、以下のフィールドで構成されます。

group

INT(32)：値：1

測定対象のデバイスのグループ番号です。（グループは物理エンクロージャーに相当します。）すべてのグループを指定するには、リテラル-1 を使用します。

module

INT(32)：値：1

測定対象のデバイスのモジュール番号です。すべてのモジュールを指定するには、リテラル-1 を使用します。

slot

INT(32)：値：1

測定対象のデバイスのスロット番号です。すべてのスロットを指定するには、リテラル-1 を使用します。

device^name

INT：値：12

測定対象のデバイスの名前です。名前は左詰めにして空白で埋める必要があり、ドル記号（$）で始める必要があります。すべてのデバイスを指定するには、ドル記号とアスタリスク（$*）を使用します。

PLPT

（Measure H03/J01 以降の PVU）測定対象のデバイスに関する情報が含まれている構造体です。

```
Struct plpt^def (*)
  Begin
  unsigned(8) plpt^flags;      ! Flags for interpreting this struct
  unsigned(8) path;            ! Path
  Int         lun;             ! Logical unit number
  Int         partition;       ! Partition number (0, if not partitioned)
  Int         target^id;       ! 0, for CLIM; scsi-id for non-CLIM
End;
```

plpt^flags

UNSIGNED(8)：値：1

この構造体をどのように解釈すべきかを記述するビットフラグの配列です。

ビット 0 (%H80) MEAS_CLIM_REL: Measure H03 以降の PVU データファイルでは、このビットは常に 1 です。

ビット 1 (%H40) MEAS_PATH_SEL: このビットは、測定対象として複数のパスを持つ DISC を選択するときに、エンティティ記述子でのみ使用されます。1 の場合は、*GMS*、*lun*、*partition*、*target^id*、*config^name* によってではなく、*device^name* および *path* のみによってデバイス選択を指定します。

ビット 2 (%H20) MEAS_CLIM_DEVICE: この記述子が CLIM デバイス用であることを指定します。

ビット 3 (%H10) MEAS_MULTIPLE_CLIMS: （H06.23/J06.12 以降の RVU）この記述子の *config^name* フィールドに複数の CLIM 名が含まれることを指定します。DISC エンティティでのみ有効で、MEASREAD、MEASREAD_DIFF_、および MEASWRITE_DIFF_プロシージャーコールの記述子でのみ使用できます。

path

UNSIGNED(8)：値：1

測定対象のデバイスのパスです。すべてのパスを指定するには、リテラル-1 を使用します。MEAS_PATH_PRIMARY、MEAS_PATH_BACKUP、MEAS_PATH_MIRROR、および MEAS_PATH_MIRROR_BACKUP も使用できます。DISC エンティティでのみ有効です。

lun

INT：値：1

測定対象の CLIM 内のディスクに割り当てられている論理ユニット番号（lun）です。CLIM 以外のデバイスには使用されません。すべての lun を指定するには、リテラル-1 を使用します。

partition

INT：値：1

（H06.23/J06.12 以降の RVU）パーティション分割されたディスク内のパーティションに割り当てられているパーティション番号です。CLIM 以外のデバイスまたはパーティション分割されていない CLIM には使用されません。DISC エンティティでのみ有効です。

target^id

INT：値：1

測定対象の CLIM 以外のデバイスの SCSI ポート識別子です。CLIM デバイスでは使用されません。すべての SCSI ポートを指定するには、リテラル-1 を使用します。

config^name

INT：値：32

システム構成データベースで定義されている、デバイスの構成名です。

config^name は、単一の ServerNet アドレス指定可能コントローラー（SAC）または複数の ServerNet アドレス指定可能コントローラーが含まれているアダプターのいずれかの構成名にすることができます。このような場合、指定には、その SAC またはアダプターでサポートされている、要求されたタイプのすべてのデバイスが含まれます。

config^name は、null で終了し、null で埋める必要があります。最初のバイトのアスタリスクは、すべての構成名を指定します（これは `config^name^s := "*"`で設定できます）。

構成テーブルのスペースを節約するために、DEVICE^CLIM^DESC 構造体から *config^name* を省きます。Measure では記述子の長さを使用して構成テーブル内の次の記述子の先頭を確認するため、記述子の長さ（*len*）を設定して、*config^name* を使用しているかどうかを示す必要があります。*config^name* を使用しない場合は、*len* を 52 に設定し、記述子の最初の 52 バイトだけを構成テーブルに移動します。*config^name* を含めるには、*len* を 116 に設定し、記述子全体を構成テーブルに移動します。

config^name^s

デバイスの構成名の文字列表現です。

DEVICE^SVNET^DESC

G シリーズの RVU 以降を実行しているシステムに ServerNet によって接続されている DEVICE または DISC エンティティの記述子です。

DEVICE^SVNET^DESC 記述子は SCSI デバイスには使用しないでください。ServerNet/DA または FCSA によって接続されているこのような SCSI デバイスを測定するには、測定するデバイスの SCSI ポートを識別するフィールドが含まれている DEVICE^SVNET^DESC^G05 記述子を使用してください。

```
Struct Device^SvNet^desc (*)
  Begin
  Int    Type;                ! Entity type
  Int    Len;                 ! Byte length of this record
  Int    CPU^number;
  Int    ServerNET;
  Struct GMS;
    Begin
      Int(32)  Group;         ! Group number
      Int(32)  Module;        ! Module number
      Int(32)  Slot;          ! Slot number
    End;
  Int    Device^name[0:11];
  Int    Config^name[0:31];   ! OSCONFIG name for drive, adaptor, or SAC
  String Config^name^s[0:63]  = Config^name;
  End;
```

type

INT：値：1

次のいずれかです。

エンティティタイプ	リテラル	数値識別子
DISC	DISC^T	7
DEVICE	IODEV^T	8

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は、記述子に構成名が含まれていない場合は 44、記述子に構成名が含まれている場合は 108 です。

cpu^number

INT：値：1

デバイスが構成されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

servernet

INT：値：1

ServerNet ファブリックです。このフィールドは、サポートされている物理デバイスへの特定のパスまたはパスのセットを識別します。X ファブリックを指定するには 0、Y ファブリックを指定するには 1、または両方のファブリックを指定するにはリテラル-1 を使用します。

GMS

物理的位置のアドレス（グループ、モジュール、スロット）です。GMS 構造体は、以下のフィールドで構成されます。

group

INT(32)：値：1

測定対象のデバイスのグループ番号です。（グループは物理エンクロージャーに相当します。）すべてのグループを指定するには、リテラル-1 を使用します。

module

INT(32)：値：1

測定対象のデバイスのモジュール番号です。すべてのモジュールを指定するには、リテラル-1 を使用します。

slot

INT(32)：値：1

測定対象のデバイスのスロット番号です。すべてのスロットを指定するには、リテラル-1 を使用します。

device^name

INT：値：12

測定対象のデバイスの名前です。名前は左詰めにして空白で埋める必要があり、ドル記号（$）で始める必要があります。すべてのデバイスを指定するには、ドル記号とアスタリスク（$*）を使用します。

config^name

INT：値：32

システム構成データベースで定義されている、ディスクまたはデバイスの構成名です。

config^name は、単一の ServerNet アドレス指定可能コントローラー（SAC）または複数の ServerNet アドレス指定可能コントローラーが含まれているアダプターのいずれかの構成名にすることができます。このような場合、指定には、その SAC またはアダプターでサポートされている、要求されたタイプのすべてのデバイスが含まれます。

config^name は、null で終了し、null で埋める必要があります。最初のバイトのアスタリスクは、すべての構成名を指定します。

構成テーブルのスペースを節約するために、DEVICE^SVNET^DESC 構造体から *config^name* を省くことができます。Measure では記述子の長さを使用して構成テーブル内の次の記述子の先頭を確認するため、記述子の長さ（*len*）を設定して、*config^name* を使用しているかどうかを示す必要があります。*config^name* を使用しない場合は、*len* を 44 に設定し、記述子の最初の 44 バイトだけを構成テーブルに移動します。*config^name* を含めるには、*len* を 108 に設定し、記述子全体を構成テーブルに移動します。

config^name^s

デバイスの構成名の文字列表現です。

DEVICE^SVNET^DESC^G05

G シリーズ以降の RVU を実行しているシステムに Servernet/DA や FCSA などの SCSI 接続によって接続されている、DEVICE または DISC エンティティの記述子です。DEVICE^SVNET^DESC^G05 には、SCSI デバイスを一意に識別するために必要な SCSI ポート識別子のフィールドが含まれています。DEVICE^SVNET^DESC または DEVICE^SVNET^DESC^G05 を使用して、SCSI 以外の NonStop S シリーズのデバイスを記述できます。

```
Struct Device^SvNet^Desc^G05 (*)
  Begin
  Int    Type;                ! Entity type
  Int    Len;                 ! Byte length of this record
  Int    CPU^number;
  Int    Fabric
  Int    ServerNET;
  Struct GMS;
    Begin
      Int(32)  Group;         ! Group number
      Int(32)  Module;        ! Module number
      Int(32)  Slot;          ! Slot number
    End;
  Int    Device^name[0:11];
  Fixed  SCSI^Id;
  Int    Config^name[0:31];   ! OSCONFIG name for drive,adaptor,or SAC
  String Config^name^s[0:63]  = Config^name;
  End;
```

type

INT：値：1

次のいずれかです。

エンティティタイプ	リテラル	数値識別子
DISC	DISC^T	7
DEVICE	IODEV^T	8

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は、記述子に構成名が含まれていない場合は 52、記述子に構成名が含まれている場合は 116 です。

cpu^number

INT：値：1

デバイスが構成されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

servernet

INT：値：1

ServerNet ファブリックです。このフィールドは、サポートされている物理デバイスへの特定のパスまたはパスのセットを識別します。X ファブリックを指定するには 0、Y ファブリックを指定するには 1、または両方のファブリックを指定するにはリテラル-1 を使用します。

ServerNet/DA によって接続されているデバイスの場合は、リテラル-1 を使用します。

FCSA によって接続されているデバイスの場合は、0 を使用して X ファブリックを指定します。慣例では、デュアルファブリックコントローラーは X によって指定されます。

GMS

物理的位置のアドレス（グループ、モジュール、スロット）です。GMS 構造体は、以下のフィールドで構成されます。

group

INT(32)：値：1

測定対象のデバイスのグループ番号です。（グループは物理エンクロージャーに相当します。）すべてのグループを指定するには、リテラル-1 を使用します。

module

INT(32)：値：1

測定対象のデバイスのモジュール番号です。すべてのモジュールを指定するには、リテラル-1 を使用します。

slot

INT(32)：値：1

測定対象のデバイスのスロット番号です。すべてのスロットを指定するには、リテラル-1 を使用します。

device^name

INT：値：12

測定対象のデバイスの名前です。名前は左詰めにして空白で埋める必要があり、ドル記号（$）で始める必要があります。すべてのデバイスを指定するには、ドル記号とアスタリスク（$*）を使用します。

scsi^id

INT(64)：値：1

測定対象のデバイスの SCSI ポート識別子です。すべての SCSI ポートを指定するには、リテラル-1 を使用します。

config^name

INT：値：32

システム構成データベースで定義されている、デバイスの構成名です。

config^name は、単一の ServerNet アドレス指定可能コントローラー（SAC）または複数の ServerNet アドレス指定可能コントローラーが含まれているアダプターのいずれかの構成名にすることができます。このような場合、指定には、その SAC またはアダプターでサポートされている、要求されたタイプのすべてのデバイスが含まれます。

config^name は、null で終了し、null で埋める必要があります。最初のバイトのアスタリスクは、すべての構成名を指定します。

構成テーブルのスペースを節約するために、DEVICE^SVNET^DESC^G05 構造体から *config^name* を省きます。Measure では記述子の長さを使用して構成テーブル内の次の記述子の先頭を確認するため、記述子の長さ（*len*）を設定して、*config^name* を使用しているかどうかを示す必要があります。*config^name* を使用しない場合は、*len* を 52 に設定し、記述子の最初の 52 バイトだけを構成テーブルに移動します。*config^name* を含めるには、*len* を 116 に設定し、記述子全体を構成テーブルに移動します。

config^name^s

デバイスの構成名の文字列表現です。

DISKFILE^DESC

DISKFILE エンティティの記述子です。

OSS ファイルパス名の場合は、テンプレート **DISKFILE^OSS^DESC**(558 ページ)を使用してください。

```
Struct Diskfile^desc (*);      -- Diskfile descriptor declaration
  Begin
  Int    Type;                 -- Diskfile entity type
  Int    Len;                  -- byte length of the descriptor
  Int    Filecode;             -- file code
  Int    Filetype;             -- file type
  Int    File^name[0:11];      -- as it says
  End;
```

type

INT：値：1

リテラル値 DISKFILE^T または数値識別子 19 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は 32 です。

filecode

INT：値：1

3 桁のファイルコードです（たとえば、オブジェクトファイルの場合は 100、EDIT ファイルの場合は 101）。ファイルコードの完全なリストについては、*File Utility Program (FUP) Reference Manual* を参照してください。

filetype

INT：値：1

測定対象のファイルのタイプです。次の 4 つのファイルタイプがあります。

ファイルタイプ	値
非構造化	0
相対値	1
エントリーシーケンス	2
キーシーケンス	3

file^name

INT：値：12

測定対象のファイルの名前です。この名前は、左詰めにして空白で埋められ、ローカル内部形式である必要があります。*file^name* 内のボリューム、サブボリューム、およびファイル名の各フィールドには、アスタリスク（*）を使用できます。

DISKFILE^OSS^DESC

OSS ファイルパス名をサポートする DISKFILE エンティティの記述子です。

```
Struct Diskfile^OSS^desc (*)           -- Diskfile descriptor declaration
  Begin
  Int    Type;                         -- Diskfile entity type
  Int    Len;                          -- byte length of the descriptor
  Int    Filecode;                     -- file code
  Int    Filetype;                     -- file type
  Int    File^name[0:11];              -- as it says
  Struct File^name^MID( MID^def );     -- Internal identifier for OSS
pathname
  End;
```

次に、DISKFILE^DESC 記述子とは異なる DISKFILE^OSS^DESC フィールドの定義を示します。その他のすべてのフィールドの定義については、**DISKFILE^DESC**(557 ページ)を参照してください。

FILE^NAME^MID

OSS パス名の内部形式のファイル名が含まれる構造体です。

```
STRUCT MID^def (*)                    ! High level view of Measure ID
  begin
  Int    Pathid[0:11];
```

```
  Int    Crvsn[0:2];
  End;
```

pathid

INT：値：12

OSS ファイルセットに属している指定されたディスクファイルの OSS ファイルパス名の内部形式表現です。すべての PATHID を指定するには、PATHID[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

DISKFILE^ANSI^DESC

ANSI SQL オブジェクトをサポートする DISKFILE エンティティの記述子です。

```
Struct  Diskfile^ANSI^desc (*)      -- Diskfile descriptor declaration
  Begin
  Int    Type;                      -- Diskfile entity type
  Int    Len;                       -- byte length of the descriptor
  Int    Filecode;                  -- file code
  Int    Filetype;                  -- file type
  Int    File^name[0:11];           -- as it says
  Struct File^name^MID( MID^def );  -- Internal identifier
                                    -- for OSS pathname
  Struct SQLMX^Obj^Desc (ANSI^SQL^object^def);   -- ANSI SQL name
                                                 -- selection info
  End;
```

次に、DISKFILE^DESC 記述子とは異なる DISKFILE^ANSI^DESC フィールドの定義を示します。その他のすべてのフィールドの定義については、**DISKFILE^DESC**(557 ページ)を参照してください。

FILE^NAME^MID

OSS パス名の内部形式のファイル名が含まれる構造体です。

```
STRUCT MID^def (*)       ! High level view of Measure ID
  begin
  Int    Pathid[0:11];
  Int    Crvsn[0:2];
  end;
```

pathid

INT：値：12

OSS ファイルパス名の内部形式表現です。内部形式表現を使用してディスクファイルにアクセスした場合、すべての PATHID 値を指定するには、Pathid[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

SQLMX^obj^desc

INT：値：12

ANSI SQL オブジェクト定義の内部形式表現です。このフィールドは MEAS_GETDESCINFO_プロシージャーを使用して入力され、その入力は実際の ANSI SQL 名ですが、出力はこの構造体の内容です。

ENDPOINT_DESC

L シリーズ以降の RVU を実行しているシステム上の ENDPOINT エンティティの記述子です。

```
Struct  Endpoint_desc (*) FIELDALIGN (SHARED2);
  Int    Type;                                  -- Endpoint entity type
  Int    Len;                                   -- byte length of the
descriptor
  Int    CPU^number;                            -- CPU number
  Int    Ctrl^type;                             -- controller type
  String  reserved1[0:3];                       -- Reserved for future use
  Int(32) Remote^Cluster;                       -- SuperCluster number
  String  reserved2[0:3];                       -- Reserved for future use
  Int(32) Remote^CPU;                           -- IPC, RIPC destination CPU
  Int(32) Node^class;                           -- controller type
  Int     Config^name[0:31];                    -- OSCONFIG name for
adapter, SAC or CLIM
  String  Config^name^s[0:63]                   = Config^name;
  End;
```

type

INT：値：1

リテラル値 ENDPOINT_T または数値識別子 18 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は、構成名が含まれていない ENDPOINT_T 記述子の場合は 28、構成名が含まれている SVNET^DESC 記述子の場合は 92 です。

cpu^number

INT：値：1

測定対象の SAC が構成されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

ctrl^type

INT：値：1

この SAC タイプの Hewlett Packard Enterprise 製品番号です。すべての SAC タイプを指定するには、リテラル-1 を使用します。製品番号については、各コントローラーに固有のドキュメントを参照してください。

また、IPC/RIPC LINKER の場合は 0、IPC/RIPC LISTENER の場合は 1、ストレージサブシステムから記録される CLMS アクティビティを指定する場合は 2、CIP サブシステムから記録される CLMS アクティビティを指定する場合は 3 です。これらの値のリテラルは、ファイル MEASDECS および MEASCHMA にあります。

reserved1

INT：値：3

現在は使用されていません。

remote^cluster

INT(32)：値：1 モジュールを再定義します

エンティティと通信中のシステムの EndPoint ノード番号です。すべての EndPoint クラスターシステムを指定するには、リテラル-1 を使用します。

reserved2

INT：値：3

現在は使用されていません。

remote^CPU

INT(32)：値：1 サブデバイスを再定義します

エンティティと通信中のリモートノード上の CPU の番号です。すべてのリモートの CPU を指定するには、リテラル-1 を使用します。

node^class

INT(32)：値：1

測定対象の SAC の一般的なクラスです。L シリーズの RVU では、ノードクラス IPC、RIPC、CLIM、CLMI、CLMO、および CLMS のみがサポートされます。ノードクラス名は、二重引用符で囲み、空白で埋めて 4 バイトにする必要があります。

node^class^clim = "CLIM",

node^class^clim^ip = "CLMI",

node^class^clim^open = "CLMO",

node^class^clim^storage = "CLMS",

node^class^colo = "COLO",

node^class^lan = "ENET",

node^class^ipc = "IPC",

node^class^mont = "MONT",

node^class^nioc = "NIOC",

node^class^ripc = "RIPC",

node^class^scsi = "SCSI",

node^class^wan = "SWAN",

node^class^all = " ";*

config^name

INT：値：32

システム構成データベースで定義されている、測定対象のアダプター、SAC、または CLIM の構成名です。

config^name は、複数の EndPoint アドレス指定可能コントローラーが含まれているアダプターの構成名にすることもできます。このような場合、指定には、そのアダプターでサポートされている、要求されたタイプのすべてのデバイスが含まれます。

config^name は、null で終了し、null で埋める必要があります。最初のバイトのアスタリスクは、すべての構成名を指定します。

構成テーブルのスペースを節約するために、ENDPOINT^DESC 構造体から *config^name* を省くことができます。Measure では記述子の長さを使用して構成テーブル内の次の記述子の先頭を確認するため、記述子の長さ（*len*）を設定して、*config^name* を使用しているかどうかを示す必要があります。*config^name* を使用しない場合は、*len* を 28 に設定し、記述子の最初の 28 バイトだけを構成テーブルに移動します。*config^name* を含めるには、*len* を 92 に設定し、記述子全体を構成テーブルに移動します。

config^name^s

構成名の文字列表現です。

FABRIC_DESC

L シリーズの RVU を実行しているシステム上の FABRIC エンティティの記述子です。

```
Struct Fabric_desc (*) FIELDALIGN (SHARED2);        -- Fabric descriptor
structure
  Begin
  Int    Type;                                      -- Entity type
  Int    Len;                                       -- Byte length of this
record
  Int    CPU_number;
  Int    Endpoint_type;                             -- CPU, STORAGE, IP or
Open
  Int    Target_CPU;                                -- Used for CPU endpoint
  Int    Target_node;                               -- ServerNet node number
(G-, H-, or
                                                    -- J-series RVUs)
                                                    -- or Expand node number

(L-series RVUs)
  Int    Target_node_name[0:3];
  String Target_node_name_s = Target_node_name;
  String Virtual_chan_name_s[0:7];
  String Endpoint_name_s[0:7];
  End;
```

type

INT：値：1

リテラル値 FABRIC_T または数値識別子 22 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。設定例は、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。

cpu_number

INT：値：1

エンドポイントとの間のトラフィックが測定される CPU 番号です。すべての CPU を指定するには、リテラル-1 を使用します。

endpoint_type

INT：値：1

エンドポイントのタイプです。1 は CPU（ローカルまたはリモート）、2 はストレージ CLIM、3 は IP CLIM、4 は Telco CLIM です。すべてのエンドポイントタイプを指定するには、リテラル-1 を使用します。これらの値のリテラルは、T0933 ファイル、extNs.h、および Measure の MEASDECS および MEASCHMA ファイルにあります。

target_cpu

INT：値：1

エンドポイントタイプが CPU の場合は、ターゲット CPU の CPU 番号です。有効な値は 0 から 15 です。他のすべてのタイプの場合、このフィールドは使用されません。すべての CPU を指定するには、リテラル-1 を使用します。

target_node

INT：値：1

G、H、J シリーズの RVU の ServerNet ノード番号または L シリーズの RVU 上の Expand ノード番号です。また、これは ENDPOINT/SERVERNET 記述子の remote^cluster フィールドと同じです。有効な値は 0 から 254 です。これは CPU エンドポイントタイプについてのみ使用されます。すべてのリモートノードを指定するには、リテラル-1 を使用します。

target_node_name

INT：値：4

target_node_name_s

STRING：値：8

エンドポイントタイプが CPU の場合は、target_node の名前です。他のすべてのタイプの場合、このフィールドは使用されません。この長さは 8 バイトに制限されます。最初のバイトは「\」で、すべての未使用バイトは空白である必要があります。すべてのノード名を示すには、「*」を使用します。

virtual_chan_name_s

STRING：値：8

エンドポイントへの仮想チャネルの名前です。この長さは 8 バイトに制限されます。未使用のすべてのバイトを null にする必要があります。すべての仮想チャネル名を示すには、「*」を使用します。

endpoint_name_s

STRING：値：8

エンドポイントの名前です。CPU エンドポイントの場合、このフィールドは使用されません。CLIM エンドポイントでは、このフィールドは長さが 8 バイトに制限されている CLIM 名です。未使用のすべてのバイトを null にする必要があります。すべてのエンドポイント名を示すには、「*」を使用します。

FILE^OPEN^DESC

DISCOPEN エンティティまたは FILE エンティティ用の記述子で、EDIT ファイル、オブジェクトファイル、および一時ファイルを記述するためのフィールドを提供します。OSS ファイルのパス名と ANSI SQL 名を指定するために、別々の記述子が使用されます。

- DISCOPEN および FILE エンティティ用に OSS ファイルのパス名を指定するには、**FILE^OPEN^OSS^DESC**(567 ページ)テンプレートを使用します。
- DISCOPEN および FILE エンティティ用に ANSI SQL 名を指定するには、**FILE^OPEN^ANSI^DESC**(569 ページ)テンプレートを使用します。

さらに、オープンとオープナープロセス（そのプロセス名とプログラムファイル）を所有するプロセスを記述するためのフィールドを提供するために、FILE エンティティによって別の 2 つの記述子が使用されます。これらは **FILE^OPEN^DESC^D10**(566 ページ)および **FILE^OPEN^OSS^DESC^D10**(567 ページ)です。

上記の個々のエンティティ記述子形式は、その長さによっても区別されます。

エンティティ記述子	バイト数
FILE^OPEN^DESC	46
FILE^OPEN^OSS^DESC	76
FILE^OPEN^DESC^D10	80
FILE^OPEN^OSS^DESC^D10	110 または 140
FILE^OPEN^ANSI^DESC	164

```
Struct File^open^desc (*);
  Begin
  Int    Type;                     ! Entity type
  Int    Len;                      ! Byte length of this record
  Int    Opener^CPU;               ! Opener's CPU
  Int    Opener^PIN;               ! Opener's PIN
  Int    File^number;              ! File ACB number
  Int    OCB^number = File^number; ! Discopen OCB
  Int    File^name[0:11];          ! File name being opened
  Int    DP^CPU;                   ! Disc process' cpu number (primary or
backup)
  Int    File^open^type = DP^CPU;  ! Type of file open
  Int    System^number;            ! Of remote file or remote opener.
  Int    System^name[0:3]          ! Of remote file or remote opener; Set
                                   ! to all blanks for local system files
                                   ! or openers
  End;
```

type

INT：値：1

次のいずれかです。

エンティティタイプ	リテラル	数値識別子
FILE	FILEOP^T	5
DISCOPEN	DFILEOP^T	6

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。設定例は、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は 46 です。

opener^cpu

INT：値：1

オープナープロセスが実行されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

opener^pin

INT：値：1

オープナープロセスのプロセス識別番号です。すべての PIN を指定するには、リテラル-1 を使用します。

測定対象のファイルにアクセスしているプロセスを識別するには、*opener^cpu* および *opener^pin* を使用します。

file^number

INT：値：1

測定対象のファイルのファイル識別番号です。すべてのファイルを指定するには、リテラル-1 を使用します。

ocb^number

INT：値：1

これは、使用されなくなりました。

file^name

INT：値：12

測定対象のファイルの名前です。ファイル名は、左詰めにして空白で埋められ、ローカル内部形式である必要があります。*file^name* 内のボリューム、サブボリューム、およびファイル名の各フィールドには、アスタリスク（*）を使用できます。

dp^cpu | file^open^type

INT：値：1

DISCOPEN 仕様のディスクプロセス（プライマリまたはバックアップ）を含む CPU の番号です。G10 Measure PVU 以降では、*dp^cpu* は使用されなくなり、*file^open^type* に置き換えられました。

DISCOPEN エンティティの場合、*file^open^type* には CPU 番号か、すべての CPU を意味する値-1 を含める必要があります。

FILE エンティティの場合、*file^open^type* には以下のいずれかの値を指定できます。

```
ALLFILES      = %B0000000111111111,
GUARDIAN      = %B1111111111111111,
ENSCRIBE      = %B1111111011111111,
UNSTRUCT      = %B1111111011100001,
RELFILE       = %B1111111011100010,
ENTRYFILE     = %B1111111011100100,
KEYFILE       = %B1111111011101000,
SQLFILE       = %B1111111111100000,
PROCFILE      = %B1111111011110000,
OSS           = %B0000000011100000,
OSSDISK       = %B0111111011100000,
OSSFIFO       = %B1101111011100000,
OSSPIPE       = %B1011111011100000,
OSSSOCKET     = %B1110000011100000,
OSSUNIXSOCKET = %B1110011011100000,
OSSUNIXDGRAM  = %B1110111011100000,
OSSUNIXSTREAM = %B1111011011100000,
OSSINETSOCKET = %B1111100011100000,
OSSINETDGRAM  = %B1111101011100000,
```

```
OSSINETSTREAM = %B1111110011100000;
```

system^number

INT：値：1

オープナープロセスが実行されているシステムの番号です。すべてのシステムを指定するには、リテラル-1 を使用します。

system^name

INT：値：4

オープナープロセスが実行されているシステムの名前です。システム名は、左詰めにして空白で埋める必要があります。先頭にバックスラッシュ（\）を付けた後に内部形式のシステム名が続く必要があります。すべてのシステムを指定するには、バックスラッシュとアスタリスク（\ *）を使用します。

FILE^OPEN^DESC^D10

オープン（プロセス名）とオープナープロセス（プログラム名のファイル）を所有しているプロセスを記述するためのフィールドを提供する FILE エンティティの記述子です。

OSS ファイルパス名の場合は、テンプレート **FILE^OPEN^OSS^DESC^D10**(567 ページ)を使用してください。

```
Struct File^open^desc^d10 (*);
  Begin
  File^open^desc^fields;
  Int    Measflags;                 ! To be used later, initialize
                                    ! to 0 for now
  Int    Opener^pname[0:3];         ! opener's process name
  Int    Opener^program^fname[0:11]; ! opener's program file name
  End;
```

次に、FILE^OPEN^DESC 記述子とは異なる FILE^OPEN^DESC^D10 フィールドの定義を示します。その他のすべてのフィールドの定義については、**FILE^OPEN^DESC**(563 ページ)を参照してください。

File^open^desc^fields

FILE^OPEN^DESC 記述子のフィールドです。

measflags

INT：値：1

今後使用するためのフィールドです。0 に初期化します。

opener^pname

INT：値：4

オープナープロセスの名前です。プロセス名は、左詰めにして空白で埋める必要があります。ドル記号（$）から始め、内部形式でプロセス名を続ける必要があります。すべてのプロセスを指定するには、ドル記号にアスタリスクを続けて（$*）使用します。

opener^program^fname

INT：値：12

オープナープロセスが実行しているオブジェクトファイルです。プロセス名は、左詰めにして空白で埋められ、ローカルの内部ファイル名形式である必要があります。ボリューム、サブボリューム、およびファイル名の各フィールドには、アスタリスク（*）を使用できます。

FILE^OPEN^OSS^DESC

OSS ファイルパス名をサポートする、G06.12 以降の RVU を実行中のシステム上の DISCOPEN エンティティまたは FILE エンティティ用の記述子です。

```
Struct File^open^oss^desc (*);
  Begin
  File^open^desc^fields;
  Struct File^name^MID;      ! Internal identifier for OSS pathname
    Begin
    Int    Pathid[0:11];     ! Internal format OSS file pathname
    Int    Crvsn[0:2];       ! Creation Volume Sequence Number
    End;
  End;
```

次に、FILE^OPEN^DESC 記述子とは異なる FILE^OPEN^OSS^DESC フィールドの定義を示します。その他のすべてのフィールドの定義については、**FILE^OPEN^DESC**(563 ページ)を参照してください。

File^open^desc^fields

FILE^OPEN^DESC 記述子のフィールドです。

FILE^NAME^MID

OSS パス名の内部形式のファイル名が含まれる構造体です。

pathid

INT：値：12

OSS ファイルパス名の内部形式表現です。内部形式表現を使用してディスクファイルにアクセスした場合、すべての PATHID 値を指定するには、Pathid[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

FILE^OPEN^OSS^DESC^D10

オープン（プロセス名）とオープナープロセス（プログラム名のファイル）を所有するプロセスを記述するためのフィールドを提供し、OSS ファイルパス名をサポートする、FILE エンティティの記述子です。

OPENER^PROGRAM^FNAME^MID 記述子を含めることはオプションです。

OPENER^PROGRAM^FNAME^MID を含めずに FILE^NAME^MID を指定することはできますが、OPENER^PROGRAM^FNAME^MID を単独で使用することはできません。

```
Struct File^open^oss^desc^d10 (*)
  Begin
  File^open^desc^fields;
  Struct File^name^MID( MID^def );           ! Internal id for OSS pathname
  Int    Measflags;                          ! To be used later,
                                             ! initialize to 0 for now
  Int    Opener^pname[0:3];                  ! opener's process name
  Int    Opener^program^fname[0:11];         ! opener's program file name
  Struct Opener^program^fname^MID( MID^def ); ! Internal identifier
                                             ! for OSS pathname
  End;
```

次に、FILE^OPEN^DESC 記述子とは異なる FILE^OPEN^OSS^DESC^D10 フィールドの定義を示します。その他のすべてのフィールドの定義については、**FILE^OPEN^DESC**(563 ページ)を参照してください。

File^open^desc^fields

FILE^OPEN^DESC 記述子のフィールドです。

FILE^NAME^MID

OSS パス名の内部形式のファイル名が含まれる構造体です。

```
STRUCT MID^def (*)       ! High level view of Measure ID
  begin
  Int    Pathid[0:11];
  Int    Crvsn[0:2];
  end;
```

pathid

INT：値：12

OSS ファイルパス名の内部形式表現です。内部形式表現を使用してディスクファイルにアクセスした場合、すべての PATHID 値を指定するには、Pathid[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

measflags

INT：値：1

今後使用するためのフィールドで、0 に初期化されます。

opener^pname

INT：値：4

オープナープロセスの名前です。プロセス名は、左詰めにして空白で埋める必要があります。ドル記号（$）から始め、内部形式でプロセス名を続ける必要があります。すべてのプロセスを指定するには、ドル記号にアスタリスクを続けて（$*）使用します。

opener^program^fname

INT：値：12

オープナープロセスが実行しているオブジェクトファイルです。プロセス名は、左詰めにして空白で埋められ、ローカルの内部ファイル名形式である必要があります。ボリューム、サブボリューム、およびファイル名の各フィールドには、アスタリスク（*）を使用できます。

OPENER^PROGRAM^FNAME^MID

ファイルが OSS ファイルセットに属している場合は、OPENER^PROGRAM^FILENAME の OSS パス名の内部形式のファイル名を含む構造体です。

pathid

INT：値：12

ファイルが OSS ファイルセットに属している場合は、OPENER^PROGRAM^FILENAME の OSS ファイルパス名の内部形式表現です。すべての PATHID を指定するには、Pathid[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

FILE^OPEN^ANSI^DESC

ANSI SQL オブジェクトをサポートする FILE または DISCOPEN エンティティの記述子です。FILE^OPEN^ANSI^DESC 記述子フィールドは MEAS_GETDESCINFO_プロシージャーを使用して入力され、その入力は実際の ANSI SQL 名ですが、出力はこの構造体の内容です。

```
Struct File^open^ANSI^desc (*)
  Begin
  File^open^desc^fields;
  Struct File^name^MID( MID^def );             ! Internal id for OSS pathname
  Int    Measflags;                            ! To be used later,
                                               ! initialize to 0 for now
  Int    Opener^pname[0:3];                    ! opener's process name
  Int    Opener^program^fname[0:11];           ! opener's program file name
  Struct Opener^program^fname^MID( MID^def );  ! Internal id for OSS pathname
  Struct SQLMX^Obj^Desc (ANSI^SQL^object^def); ! ANSI SQL Name selection info
  End;
```

次に、FILE^OPEN^DESC 記述子とは異なる FILE^OPEN^ANSI^DESC フィールドの定義を示します。その他のすべてのフィールドの定義については、**FILE^OPEN^DESC**(563 ページ)を参照してください。

File^open^desc^fields

FILE^OPEN^DESC 記述子のフィールドです。

FILE^NAME^MID

OSS パス名の内部形式のファイル名が含まれる構造体です。

```
STRUCT MID^def (*)       ! High level view of Measure ID
  begin
  Int    Pathid[0:11];
  Int    Crvsn[0:2];
  end;
```

pathid

INT：値：12

OSS ファイルパス名の内部形式表現です。内部形式表現を使用してディスクファイルにアクセスした場合、すべての PATHID 値を指定するには、Pathid[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

measflags

INT：値：1

今後使用するためのフィールドで、0 に初期化されます。

opener^pname

INT：値：4

オープナープロセスの名前です。プロセス名は、左詰めにして空白で埋める必要があります。ドル記号（$）から始め、内部形式でプロセス名を続ける必要があります。すべてのプロセスを指定するには、ドル記号にアスタリスクを続けて（$*）使用します。

opener^program^fname

INT：値：12

オープナープロセスが実行しているオブジェクトファイルです。プロセス名は、左詰めにして空白で埋められ、ローカルの内部ファイル名形式である必要があります。ボリューム、サブボリューム、およびファイル名の各フィールドには、アスタリスク（*）を使用できます。

OPENER^PROGRAM^FNAME^MID

ファイルが OSS ファイルセットに属している場合は、OPENER^PROGRAM^FILENAME の OSS パス名の内部形式のファイル名を含む構造体です。

pathid

INT：値：12

ファイルが OSS ファイルセットに属している場合は、OPENER^PROGRAM^FILENAME の OSS ファイルパス名の内部形式表現です。すべての PATHID を指定するには、Pathid[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

SQLMX^obj^desc

INT：値：12

ANSI SQL オブジェクト定義の内部形式表現です。このフィールドは MEAS_GETDESCINFO_プロシージャーを使用して入力され、その入力は実際の ANSI SQL 名ですが、出力はこの構造体の内容です。

OPDISK^DESC

D シリーズの RVU を実行しているシステム上の OPDISK エンティティの記述子です。

```
Struct Opdisk^desc (*)
  Begin
  Int    Type;               ! Entity type
  Int    Len;                ! Byte length of this record
  Int    CPU^number;
  Int    Ctl;                ! Controller number
  Int    Unit;               ! Unit number
  Int    Device^name[0:11];
  Int    Volume^name[0:11];
  Int    Channel;            ! Channel number
  End;
```

type

INT：値：1

リテラル値 OPDISK^T または数値識別子 17 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。

cpu^number

INT：値：1

光ディスクが構成されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

ctl

INT：値：1

測定対象の光ディスクのコントローラー番号（0～31）です。すべてのコントローラーを指定するには、リテラル-1 を使用します。

unit

INT：値：1

測定対象の光ディスクのユニット番号です。すべてのユニットを指定するには、リテラル-1 を使用します。

device^name

INT：値：12

測定対象の光ディスクのデバイス名です。この名前は、左詰めにして空白で埋める必要があります。ドル記号（$）から始め、内部形式でデバイス名を続ける必要があります。すべてのデバイスを指定するには、ドル記号にアスタリスクを続けて（$*）使用します。

volume^name

INT：値：12

測定対象の光ディスクボリュームの名前です。

channel-num

INT：値：1

測定対象の光ディスクのチャネル番号です。

PROCESS^DESC

PROCESS エンティティの記述子です。

OSS ファイルのパス名には、テンプレート **PROCESS^OSS^DESC**(573 ページ)を使用してください。

PROCESS^DESC 記述子は、OSSNS エンティティにも使用されます。

```
Struct Process^desc (*);
  Begin
  Int    Type;       ! Entity type
  Int    Len;        ! Byte length of record. In case of
                     ! process it includes
                     ! length of all the code-space
                     ! descriptors and for userdef
                     ! it includes length of all the counter
                     !    descriptors.
```

```
Int     CPU^number;
Int     PIN;
Int     Process^name[0:3];
Int     Program^file^name[0:11];
End;
```

type

INT：値：1

次のいずれかです。

エンティティタイプ	リテラル	数値識別子
PROCESS	PROCESS^T	2
OSSNS	OSSNS^T	21

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は 40 です。

cpu^number

INT：値：1

プロセスが実行されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

pin

INT：値：1

プロセスのプロセス識別番号です。以下のリテラル値を使用できます。

- -1（すべての PIN を指定する場合）
- -2（すべてのシステムプロセスを指定する場合）
- SYSPROCS（SYSGEN でインストールされたすべてのプロセスを指定する場合）

process^name

INT：値：4

プロセス名です。この名前は、左詰めにして空白で埋める必要があります。ドル記号（$）から始め、内部形式でプロセス名を続ける必要があります。*process^name* はシステムプロセス名にすることができます。すべてのプロセスを指定するには、ドル記号にアスタリスクを続けて（$*）使用します。

program^file^name

INT：値：12

プロセスが実行しているオブジェクトファイルです。この名前は、左詰めにして空白で埋められ、ローカルの内部ファイル名形式である必要があります。*file^name* 内のボリューム、サブボリューム、およびファイル名の各フィールドで、アスタリスク（*）を使用できます。ただし、ボリューム名は、ドル記号（$）で始める必要があります。たとえば、すべてのプログラムファイル名を指定するには、「$* * *」の値を使用します。

PROCESS^OSS^DESC

OSS ファイルパス名をサポートする PROCESS エンティティの記述子です。

```
Struct Process^OSS^desc (*);
  Begin
  Int    Type;                   ! Entity type
  Int    Len;                    ! Byte length of record. In case of
                                 ! process it includes
                                 ! length of all the code-space descriptors
                                 ! and for userdef it includes length of
                                 ! all the counter descriptors.
  Int    CPU^number;
  Int    PIN;
  Int    Process^name[0:3];
  Int    Program^file^name[0:11];
  Struct Program^file^name^MID;  ! Internal identifier for OSS pathname
    Begin
    Int    Pathid[0:11];         ! Internal format OSS file pathname
    Int    Crvsn[0:2];           ! Creation Volume Sequence Number
    End;
  End;
```

type

INT：値：1

リテラル値 PROCESS^T または数値識別子 2 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は 70 です。

cpu^number

INT：値：1

プロセスが実行されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

pin

INT：値：1

プロセスのプロセス識別番号です。以下のリテラル値を使用できます。

- -1（すべての PIN を指定する場合）
- -2（すべてのシステムプロセスを指定する場合）
- SYSPROCS（SYSGEN でインストールされたすべてのプロセスを指定する場合）

process^name

INT：値：4

プロセス名です。この名前は、左詰めにして空白で埋める必要があります。ドル記号（$）から始め、内部形式でプロセス名を続ける必要があります。*process^name* はシステムプロセス名にすることができます。すべてのプロセスを指定するには、ドル記号にアスタリスクを続けて（$*）使用します。

program^file^name

INT：値：12

プロセスが実行しているオブジェクトファイルです。この名前は、左詰めにして空白で埋められ、ローカルの内部ファイル名形式である必要があります。*file^name* 内のボリューム、サブボリューム、およびファイル名の各フィールドで、アスタリスク（*）を使用できます。ただし、ボリューム名は、ドル記号（$）で始める必要があります。たとえば、すべてのプログラムファイル名を指定するには、「$* * *」の値を使用します。

PROGRAM^FILE^NAME^MID

プログラムファイルの OSS ファイルパス名の内部形式のファイル名が含まれる構造体です。

pathid

INT：値：12

プログラムファイルの OSS ファイルパス名の内部形式表現です。ファイルが OSS ファイルシステムに所属している場合は、すべての PATHID 値を指定するには、Pathid [0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

PROCESSH^DESC

PROCESSH エンティティの記述子です。PROCESSH エンティティの記述子は長さが可変で、初期 PROCESSH^DESC 構造体に続けて **CODE^SPACE^DESC**(576 ページ)テンプレートを含めることができます。

OSS ファイルパス名の場合は、**PROCESS^OSS^DESC**(573 ページ)および **CODE^SPACE^OSS^DESC**(578 ページ)のテンプレートを使用してください。

```
Struct Processh^desc (*);
  Begin
  Int    Type;         ! Entity type
  Int    Len;          ! Byte length of record. In case of processh it
includes
                       ! length of all the code-space descriptors and for
userdef
                       ! it includes length of all the counter descriptors.
  Int    CPU^number;
  Int    PIN;
  Int    Process^name[0:3];
  Int    Program^file^name[0:11];
  Fixed  Conftime;     ! Set to 0F, for use by MEASFH only
  Int    Code^spaces; ! Number of code spaces to be measured
  End;                ! A subsequent code^space^desc is not needed for an
Entitydesc of
                       ! a previously installed entity.  But for Contab use,
as many
                       ! code^space^desc as desired must imediately follow
this struct.
```

type

INT：値：1

リテラル値 PROCESSH^T または数値識別子 3 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子とそれに関連するすべての CODE^SPACE^DESC 記述子の長さ（バイト単位）です。CODE^SPACE^DESC 拡張が記述子に含まれない場合は *len* は 50 です。

cpu^number

INT：値：1

測定対象のプロセスが実行されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

pin

INT：値：1

測定対象のプロセスのプロセス識別番号です。以下のリテラル値を使用できます。

- -1（すべてのプロセスを指定する場合）
- SYSPROCS（SYSGEN でインストールされたすべてのプロセスを指定する場合）
- ALLTIMES（すべてのプロセスの実行時間を測定する場合）
- ALLINTR（すべてのプロセスの割り込み時間のみを測定する場合）

pin を ALLTIMES または ALLINTR のリテラル値として指定した場合、SC.0、SL.*n*、SCR、または SLR のコード領域を指定するには、*code-space* を使用する必要があります。

process^name

INT：値：4

測定対象のプロセスの名前です。この名前は、左詰めにして空白で埋める必要があります。ドル記号（$）を含め、内部形式でプロセス名を続ける必要があります。*process^name* はシステムプロセス名にすることができます。すべてのプロセス名を指定するには、リテラル-1 を使用します。

program^file^name

INT：値：12

測定対象のプロセスが実行しているオブジェクトファイルの名前です。この名前は、左詰めにして空白で埋められ、ローカルの内部ファイル名形式にする必要があります。*file^name* 内のボリューム、サブボリューム、およびファイル名の各フィールドで、アスタリスク（*）を使用できます。ただし、ボリューム名は、ドル記号（$）で始める必要があります。たとえば、すべてのプログラムファイル名を指定するには、「$* * *」の値を使用します。

conftime

FIXED：値：1

0F として *conftime* を指定します。Measure サブシステムのみで使用します。

code^spaces

INT：値：1

測定対象のコード領域について記述した、下記の CODE^SPACE^DESC 記述子の数です。

注記:

PROCESSH^DESC 記述子を MEASREAD プロシージャーに渡すときに、CODE^SPACE^DESC 記述子を含めないでください。

CODE^SPACE^DESC

コード領域指定の記述子です。

```
Struct Code^space^desc (*);
  Begin
  Int      Code^space;   ! <0>=0 user,       =1 system
                         ! <1>=0 non lib,    =1 lib
                         ! <2>=0 non native, =1 native
                         ! <7>=0 no OSS,     =1 OSS MID included.
                         ! <8:15> space number
  Int      Code^ranges^file^name[0:11];
  String   Code^ranges^file^name^s = Code^ranges^file^name;
  End;
```

code^space

INT：値：1

次のような形式のコード領域番号です。

.0:2 は領域を識別します。H シリーズ以降の RVU では、この値は非ネイティブコードに対してのみ意味を持ちます。

0 = ユーザーコード — TNS または高速化（UC）
1 = ユーザーコード — TNS/R ネイティブ（UCR）
2 = ユーザーライブラリ — TNS または高速化（UL）
3 = ユーザーライブラリ — TNS/R ネイティブ（ULR）
4 = システムコード — TNS または高速化（SC）
5 = システムコード — TNS/R ネイティブ（SCR）
6 = システムライブラリ — TNS または高速化（SL）
7 = システムライブラリ — TNS/R ネイティブ（SLR）

.3:7 は現在は使用されていません。

.7 は、コード領域テンプレートが CODE^SPACE^OSS^DESC 形式であることを示します。

.8:15 は、TNS および高速化されたコード領域のみに関連し、領域番号を示します。

UC	0~15
UL	0~15
SC	0
SL	0～31

code^ranges^file^name

INT：値：12

測定対象のコードが含まれているファイルの名前です。*code^ranges^file^name* は、TNS オブジェクトファイル、EDIT ファイル、または Executable and Linking Format（ELF）ファイルである必要があります。この名前は、左詰めにして空白で埋められ、ローカル内部形式である必要があります。

code^ranges^file^name がオブジェクトファイルの場合、MEASFH はプロシージャー名とアドレスを調べて、各プロシージャーを構成に追加します。*code^ranges^file^name* が EDIT ファイルの場合は、コード範囲をアドレスの昇順で指定する必要があります。

code-range-name code-range-address

指定する値は次のとおりです。

code-range-name

測定対象のコード範囲のプロシージャー名です。TNS/R および TNS/E コード範囲にはドル記号（$）を使用できます。

code-range-address

測定対象のコード範囲のアドレスです。

TNS の EDIT ファイルでは、*code-range-address* は、8 進数のアドレス、または前の 8 進数のアドレスからのオフセットです。

TNS/R、TNS/E、および TNS/X の EDIT ファイルでは、*code-range-address* は、16 進数の仮想アドレス、または前の 16 進数のアドレスからのオフセットです。

前の *code-range-address* からのオフセットを示すには、8 進数または 16 進数のアドレスの前にプラス記号（+）を付けます。プラス記号を付けずに指定された最後の *code-range-address* がオフセットに加算されて、有効なアドレスが求められます。

code^ranges^file^name^s

OSS ファイルパス名の内部形式のファイル名の文字列表現です。

PROCESSH^OSS^DESC

OSS ファイルパス名をサポートする G06.12 以降の RVU を実行中のシステム上の PROCESSH エンティティ用の記述子です。PROCESSH エンティティ記述子は長さが可変で、いくつかの CODE^SPACE^DESC または CODE^SPACE^OSS^DESC テンプレートを、初期 PROCESSH^OSS^DESC 記述子に続けて含めることができます。これらのテンプレートは、任意の順序で使用できます。CODE^SPACE フィールド（ビット 7）のビットは、記述子の形式と次の記述子へのオフセットを指定します。

注記:

入力時に PROCESSH^DESC.CONFTIME を 0F に初期化することにより、PROCESSH^DESC が使用されるか、PROCESSH^OSS^DESC が使用されるかが決まります。

```
Struct Processh^OSS^desc (*)
  Begin
  process^oss^desc^fields;
  Fixed  Conftime;           ! Set to 0F, for use by MEASFH only
  Int    Code^spaces;        ! Number of code spaces to be measured
  End; ! A subsequent code^space^desc is not needed for an Entitydesc of
       ! a previously installed entity.  But for Contab use, as many
       ! code^space^desc as desired must imediately follow this struct.
```

次に、PROCESS^OSS^DESC 記述子とは異なる PROCESSH^OSS^DESC フィールドの定義を示します。その他のすべてのフィールドの定義については、**PROCESS^OSS^DESC**(573 ページ)を参照してください。

process^oss^desc^fields

PROCESS^OSS^DESC 記述子内のフィールドです。

len

INT：値：1

この記述子とそれに関連する CODE^SPACE^OSS^DESC 記述子の長さ（バイト単位）です。PROGRAM^FILE^NAME^MID 構造体は含まれるが、CODE^SPACE^DESC や CODE^SPACE^OSS^DESC 拡張子が含まれない場合、*len* は 80 です。

conftime

FIXED：値：1

0F として *conftime* を指定します。Measure サブシステムのみで使用します。

code^spaces

INT：値：1

下記の CODE^SPACE^OSS^DESC または CODE^SPACE^DESC 記述子の数です。これらの記述子は、測定対象のコード領域を記述します。

注記:

PROCESSH^DESC 記述子を MEASREAD プロシージャーに渡すときに、CODE^SPACE^DESC 記述子も CODE^SPACE^OSS^DESC 記述子も含めないでください。

CODE^SPACE^OSS^DESC

OSS ファイルパス名をサポートするコード領域指定の記述子です。

```
Struct Code^space^oss^desc (*);
                                    ! To be used in place of
code^space^desc
                                    ! structure when OSS file pathname has
                                    ! been used in the code^range^file^name
                                    ! specification.
  Begin
  Int      Code^space;              ! <0>=0 user,       =1 system
                                    ! <1>=0 non lib,    =1 lib
                                    ! <2>=0 non native, =1 native
                                    ! <7>=0 no OSS,     =1 OSS MID
included.
                                    ! <8:15> space number
  Int      Code^ranges^file^name[0:11];
  String   Code^ranges^file^name^s = Code^ranges^file^name;
  Struct   Code^ranges^file^name^MID; ! Internal identifier for OSS pathname
    Begin
    Int    Pathid[0:11];            ! Internal format OSS file pathname
    Int    Crvsn[0:2];              ! Creation Volume Sequence Number
    End;
  End;
```

次に、CODE^SPACE^DESC 記述子とは異なる CODE^SPACE^OSS^DESC フィールドの定義を示します。その他のすべてのフィールドの定義については、**CODE^SPACE^DESC**(576 ページ)を参照してください。

CODE^RANGES^FILE^NAME

OSS ファイルパス名の内部形式のファイル名が含まれる構造体です。

pathid

INT：値：12

コードファイルの OSS ファイルパス名の内部表現です。すべての PATHID を指定するには、PATHID[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

SQLPROC^DESC

SQLPROC エンティティの記述子です。

OSS ファイルパス名については、**SQLPROC^OSS^DESC**(580 ページ)を使用してください。

```
Struct Sqlproc^desc (*);
  Begin
  Int    Type;                    ! Entity type
  Int    Len;                     ! Byte length of record.
  Int    CPU^number;
  Int    PIN;
  Int    Process^name[0:3];
  Int    Program^file^name[0:11]
  End;
```

type

INT：値：1

リテラル値 SQLPROC^T または数値識別子 15 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は 40 です。

cpu^number

INT：値：1

測定対象のプロセスが実行されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

pin

INT：値：1

測定対象のプロセスのプロセス識別番号です。すべての PIN を指定するには、リテラル-1 を使用します。

process^name

INT：値：4

測定対象のプロセスの名前です。この名前は、左詰めにして空白で埋める必要があります。ドル記号（$）を含め、内部形式でプロセス名を続ける必要があります。すべてのプロセスを指定するには、ドル記号にアスタリスクを続けて（$*）使用します。

program^file^name

INT：値：12

測定対象のプロセスが実行しているオブジェクトファイルの名前です。この名前は、左詰めにして空白で埋められ、ローカル内部形式である必要があります。*program^file^name* 内のボリューム、サブボリューム、およびファイル名の各フィールドの指定にアスタリスク（*）を使用できます。

SQLPROC^OSS^DESC

OSS ファイルパス名をサポートする SQLPROC エンティティの記述子です。

```
Struct Sqlproc^OSS^desc (*);
  Begin
  Int    Type;                  ! Entity type
  Int    Len;                   ! Byte length of record.
  Int    CPU^number;
  Int    PIN;
  Int    Process^name[0:3];
  Int    Program^file^name[0:11];
  Struct Program^file^name^MID; ! Internal identifier for OSS pathname
    Begin
    Int    Pathid[0:11];        ! Internal format OSS file pathname
    Int    Crvsn[0:2];          ! Creation Volume Sequence Number
    End;
  End;
```

次に、SQLPROC^DESC 記述子とは異なる SQLPROC^OSS^DESC フィールドの定義を示します。その他のすべてのフィールドの定義については、**SQLPROC^DESC**(579 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。記述子に PROGRAM^FILENAME^MID 構造体が含まれていない場合は、*len* は 40 です。記述子に PROGRAM^FILENAME^MID 構造体が含まれている場合、*len* は 70 です。

PROGRAM^FILE^NAME^MID

OSS パス名の内部形式のファイル名が含まれる構造体です。

pathid

INT：値：12

指定したプログラムファイルが OSS ファイルセットの一部である場合、そのファイル名の OSS ファイルパス名の内部形式表現です。すべての PATHID を指定するには、PATHID[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

SQLSTMT^DESC

SQLSTMT エンティティの記述子です。

OSS ファイルパス名については、**SQLSTMT^OSS^DESC**(582 ページ)を使用してください。

ANSI SQL ファイル名については、**SQLSTMT^ANSI^DESC**(583 ページ)を使用してください。

注記:

測定値を構成しているときに、記述子のプロセス部分だけが使用されます。リスト表示を実行するときは、すべてのフィールドを使用できます。

```
Struct Sqlstmt^desc (*);
  Begin
```

```
Int    Type;                  ! Entity type
Int    Len;                   ! Byte length of record.
Int    CPU^number;
Int    PIN;
Int    Process^name[0:3];
Int    Program^file^name[0:11]
Int    Run^unit[0:15];        ! procedure or run-unit name
Int    Stmt^index;            ! statement index number within the run-unit
String Run^Unit^s = Run^unit;
End;
```

type

INT：値：1

リテラル値 SQLSTMT^T または数値識別子 16 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は 74 です。

cpu^number

INT：値：1

測定対象のプロセスが実行されている CPU の番号です。

pin

INT：値：1

測定対象のプロセスのプロセス識別番号です。以下のリテラル値を使用できます。

- -1（すべてのプロセスを指定する場合）
- SYSPROCS（SYSGEN でインストールされたすべてのプロセスを指定する場合）

process^name

INT：値：4

測定対象のプロセスの名前です。この名前は、左詰めにして空白で埋める必要があります。ドル記号（$）を含め、内部形式でプロセス名を続ける必要があります。すべてのプロセスを指定するには、ドル記号とアスタリスク（$*）を使用します。

program^file^name

INT：値：12

測定対象のプロセスが実行しているオブジェクトファイルの名前です。この名前は、左詰めにして空白で埋められ、ローカル内部形式である必要があります。*file^name* 内のボリューム、サブボリューム、およびファイル名の各フィールドで、アスタリスク（*）を使用できます。ただし、ボリューム名は、ドル記号（$）で始める必要があります。たとえば、すべてのプログラムファイル名を指定するには、「$* * *」の値を使用します。

run^unit

INT：値：32

測定対象の SQL 文が含まれるプロシージャーの名前です。すべての実行単位を含めるには、プロシージャーの名前を空白で埋める必要があります。アスタリスク（*）のワイルドカード文字は使用しないでください。

ANSI SQL には使用されません。

stmt^index

INT：値：1

指定された実行単位（プロシージャー）内の文のインデックス番号です。

ANSI SQL には使用されません。

run^unit^s

実行単位名の文字列表現です。

SQLSTMT^OSS^DESC

OSS ファイルパス名、SQL/MP プロシージャー、または SQL/MX モジュールをサポートする SQLSTMT エンティティの記述子です。

注記:

測定値を構成しているときに、記述子のプロセス部分だけが使用されます。リスト表示を実行するときは、すべてのフィールドを使用できます。

```
Struct Sqlstmt^OSS^desc (*);
  Begin
  Int    Type;                      ! Entity type
  Int    Len;                       ! Byte length of record.
  Int    CPU^number;
  Int    PIN;
  Int    Process^name[0:3];
  Int    Program^file^name[0:11];
  Struct Program^file^name^MID;     ! Internal identifier for OSS pathname
    Begin
    Int    Pathid[0:11];            ! Internal format OSS file pathname
    Int    Crvsn[0:2];              ! Creation Volume Sequence Number
    End;
  Int    Run^unit[0:63];            ! procedure or run-unit name
  Int    Stmt^index;                ! statement index number within the run-
unit
  String Run^Unit^s = Run^unit;
  End;
```

次に､SQLSTMT^DESC 記述子とは異なる SQLSTMT^OSS^DESC フィールドの定義を示します。その他のすべてのフィールドの定義については、**SQLSTMT^DESC**(580 ページ)を参照してください。

PROGRAM^FILE^NAME^MID

OSS パス名の内部形式のファイル名が含まれる構造体です。

pathid

INT：値：12

指定したプログラムファイル名の OSS ファイルパス名の内部形式表現です。すべての PATHID を指定するには、PATHID[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

SQLSTMT^ANSI^DESC

ANSI SQL ファイル名をサポートする SQLSTMT エンティティの記述子です。

注記:

ANSI SQL 名を記述子の *run^unit* フィールドにコピーするときには、一重引用符で囲まないでください。

注記:

測定値を構成しているときに、記述子のプロセス部分だけが使用されます。リスト表示を実行するときは、すべてのフィールドを使用できます。

```
Struct Sqlstmt^ANSI^desc (*)
  Begin                                    ! Used for SQL/MX ANSI SQL name
run-unit
                                           ! comparisons only.
  Int    Type;                             ! Entity type
  Int    Len;                              ! Byte length of record.
  Int    CPU^number;
  Int    PIN;
  Int    Process^name[0:3];
  Int    Program^file^name[0:11];
  Struct Program^file^name^MID;            ! Internal identifier for OSS
pathname
    Begin
    Int    Pathid[0:11];                   ! Internal format OSS file
pathname
    Int    Crvsn[0:2];                     ! Creation Volume Sequence Number
    End;
  Int    Version;                          ! Version of Sqlstmt desc; added
in G11
                                           ! Use the literal
sqlstmt^ANSI^desc^version
  Int    Stmt^index;                       ! statement index number within
the run-unit
  Int    Run^unit^name^len;
  Int    Minimum^ANSI^Run^unit[0:63];      ! desc is at least this long...
  Int    Run^unit = Minimum^ANSI^Run^unit; ! up to 384 words
  String Run^Unit^s = Run^unit;
  End;
```

次に、SQLSTMT^DESC および SQLSTMT^OSS^DESC 記述子とは異なる SQLSTMT^ANSI^DESC フィールドの定義を示します。その他のすべての SQLSTMT^ANSI^DESC フィールドの定義については、**SQLSTMT^DESC**(580 ページ)と **SQLSTMT^OSS^DESC**(582 ページ)を参照してください。

version

INT：値：1

このタイプの記述子を指定する sqlstmt^ANSI^desc^version です。

stmt^index

INT：値：1

ANSI SQL モジュール内での文のインデックス番号です。

run^unit^name^len

INT：値：1

run^unit フィールドの ANSI SQL モジュール名の実際の長さです。

注記:

ANSI SQL モジュール名は、128 バイトより短くすることも長くすることもできます。128 バイトを超える場合、「追加」領域に割り当てられるメモリがあることを確認する必要があります。

***minimum^ANSI^run^unit*[0:(*min^wlen^ANSI^SQLNAME* - 1)];**

INT：値：64

ここで *min^wlen^ANSI^SQLNAME* は、このフィールドの最小の長さである 64 語です。

run^unit = minimum^ANSI^run^unit

実行単位名の単語表現です。

run^unit^s = run^unit

実行単位名の文字列表現です。

注記:

実行単位名は、有効な任意の ANSI SQL モジュール名にすることができ、一重引用符で囲みません。

SVNET^DESC

G シリーズ以降の RVU を実行しているシステム上の SERVERNET エンティティの記述子です。

```
Struct  SvNet^desc (*)                              -- ServerNET descriptor
structure
  BEGIN
  Int    Type;                                      -- controller entity type
  Int    Len;                                       -- byte length of the
descriptor
  Int    CPU^number;                                -- CPU number
  Int    Ctrl^type;                                 -- controller type
  Struct GMS;
    Begin
    Int(32)  Group;                                 -- Group number
    Int(32)  Module;                                -- Module number
    Int(32)  Remote^Cluster = Module;               -- SuperCluster number
    Int(32)  Slot;                                  -- Slot number
    End;
  Int(32) Subdevice;                                -- controller type
  Int(32) Remote^CPU = Subdevice;                   -- IPC, RIPC destination CPU
  Struct PF = Subdevice;
    Begin
    Int      Port;                                  -- port number (for CLIM)
    Int      Fiber;                                 -- fiber number (for CLIM)
    End;
  Int(32) Node^class;                               -- controller type
  Int     Config^name[0:31];                        -- OSCONFIG name for
adapter, SAC, or CLIM
  String  Config^name^s[0:63] = Config^name;
  End;
```

type

INT：値：1

リテラル値 SVNET^T または数値識別子 18 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。*len* は、構成名が含まれていない SVNET^DESC 記述子の場合は 28、構成名が含まれている SVNET^DESC 記述子の場合は 92 です。

cpu^number

INT：値：1

測定対象の SAC が構成されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

ctrl^type

INT：値：1

この SAC タイプの Hewlett Packard Enterprise 製品番号です。すべての SAC タイプを指定するには、リテラル-1 を使用します。製品番号については、各コントローラーに固有のドキュメントを参照してください。

また、IPC/RIPC LINKER の場合は 0、IPC/RIPC LISTENER の場合は 1、ストレージサブシステムから記録される CLMS アクティビティを指定する場合は 2、CIP サブシステムから記録される CLMS アクティビティを指定する場合は 3 です。これらの値のリテラルは、ファイル MEASDECS および MEASCHMA にあります。

GMS

測定対象の SAC の物理的位置のアドレス（グループ、モジュール、スロット）です。これらは L シリーズの RVU で行われた測定には使用されません。GMS 構造体は、以下のフィールドで構成されます。

group

INT(32)：値：1

グループの数です （グループは物理エンクロージャーに相当します）。すべてのグループを指定するには、リテラル-1 を使用します。

module

INT(32)：値：1

モジュール番号です。すべてのモジュールを指定するには、リテラル-1 を使用します。

remote^cluster

INT(32)：値：1 モジュールを再定義します

エンティティと通信中のシステムの ServerNet ノード番号です。すべての ServerNet クラスターシステムを指定するには、リテラル-1 を使用します。

slot

INT(32)：値：1

スロット番号です。すべてのスロットを指定するには、リテラル-1 を使用します。

subdevice

INT(32)：値：1

測定対象の SAC のサブデバイス番号です。すべての SAC を指定するには、リテラル-1 を使用します。サブデバイス番号について詳しくは、各 SAC に固有のドキュメントを参照してください。

remote^CPU

INT(32)：値：1 サブデバイスを再定義します

エンティティと通信中のリモート ServerNet クラスター上の CPU の番号です。すべてのリモートの CPU を指定するには、リテラル-1 を使用します。

PF

CLIM のポートファイバー構造体です。GMS フィールドと同様、これは L シリーズの RVU での測定には使用されません。PF 構造体は、以下のフィールドで構成されます。

port

INT：値：1

測定対象の CLIM のポート番号です。すべての CLIM を指定するには、リテラル-1 を使用します。

fiber

INT：値：1

測定対象の CLIM のファイバー番号です。すべての CLIM を指定するには、リテラル-1 を使用します。

node^class

INT(32)：値：1

測定対象の SAC の一般的なクラスです。L シリーズの RVU では、ノードクラス IPC、RIPC、CLIM、CLMI、CLMO、および CLMS のみがサポートされます。ノードクラス名は、二重引用符で囲み、空白で埋めて 4 バイトにする必要があります。

node^class^clim = "CLIM",

node^class^clim^ip = "CLMI",

node^class^clim^open = "CLMO",

node^class^clim^storage = "CLMS",

node^class^colo = "COLO",

node^class^lan = "ENET",

node^class^ipc = "IPC",

node^class^mont = "MONT",

node^class^nioc = "NIOC",

node^class^ripc = "RIPC",

node^class^scsi = "SCSI",

node^class^wan = "SWAN",

node^class^all = " ";*

config^name

INT：値：32

システム構成データベースで定義されている、測定対象のアダプター、SAC、または CLIM の構成名です。

config^name は、複数の ServerNet アドレス指定可能コントローラーが含まれているアダプターの構成名にすることもできます。このような場合、指定には、そのアダプターでサポートされている、要求されたタイプのすべてのデバイスが含まれます。

config^name は、null で終了し、null で埋める必要があります。最初のバイトのアスタリスクは、すべての構成名を指定します。

構成テーブルのスペースを節約するために、SVNET^DESC 構造体から *config^name* を省くことができます。Measure では記述子の長さを使用して構成テーブル内の次の記述子の先頭を確認するため、記述子の長さ（*len*）を設定することで、*config^name* を使用しているかどうか反映す必要があります。*config^name* を使用しない場合は、*len* を 28 に設定し、記述子の最初の 28 バイトだけを構成テーブルに移動します。*config^name* を含めるには、*len* を 92 に設定し、記述子全体を構成テーブルに移動します。

config^name^s

構成名の文字列表現です。

SYSTEM^DESC

CLUSTER エンティティ（D シリーズの RVU のみ）または SYSTEM エンティティの記述子です。

```
Struct System^desc (*)
  Begin
  Int    Type;                ! Entity type
  Int    Len;                 ! Byte length of this record
  Int    LH^CPU;              ! Line handlers cpu number (primary or backup)
  Int    System^number;
  Int    System^name[0:3];
  End;
```

type

INT：値：1

次のいずれかです。

エンティティタイプ	リテラル	数値識別子
SYSTEM	REMSYS^T	11
CLUSTER	CLUSTER^T	12

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。

lh^cpu

INT：値：1

回線ハンドラー（プライマリまたはバックアップ）を含む CPU の数です。すべての CPU を指定するには、リテラル-1 を使用します。

system^number

INT：値：1

測定対象のネットワークトラフィックの送信元であるリモートシステムのシステム番号です。すべてのシステムを指定するには、リテラル-1 を使用します。

system^name

INT：値：4

測定対象のネットワークトラフィックの送信元であるリモートシステムのシステム名です。この名前は、左詰めにして空白で埋める必要があります。バックスラッシュ（\）を含め、内部形式のシステム名を続ける必要があります。すべてのシステムを指定するには、バックスラッシュにアスタリスクを続けて使用します（\ *）。

USERDEF^DESC

USERDEF エンティティの記述子です。USERDEF^DESC は COUNTER^DESC とともに使用されます。

OSS ファイルのパス名については、**USERDEF^OSS^DESC**(590 ページ)を使用してください。

```
Struct Userdef^desc (*);
  Begin
  Int    Type;                        ! Entity type
  Int    Len;                         ! Byte length of record.
  Int    CPU^number;
  Int    PIN;
  Int    Process^name[0:3];
  Int    Program^file^name[0:11];
  Int    Numcounters;                 ! Number of counters to be installed
  End;  ! A subsequent counter^desc is not needed for an Entitydesc of
        ! a previously installed entity.  But for Contab use, as many
        ! Counter^desc as desired must imediately follow this struct.
```

type

INT：値：1

リテラル値 USERDEF^T または数値識別子 4 です。

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子と、それに関連付けられている COUNTER^DESC 記述子の長さ（バイト単位）です。**COUNTER^DESC**(589 ページ)を参照してください。

cpu^number

INT：値：1

測定対象のプロセスが実行されている CPU の番号です。

pin

INT：値：1

測定対象のプロセスのプロセス識別番号です。すべての PIN を指定するには、リテラル-1 を使用します。

process^name

INT：値：4

測定対象のプロセスの名前です。この名前は、左詰めにして空白で埋める必要があります。ドル記号（$）で開始し、内部形式である必要があります。すべてのプロセスを指定するには、ドル記号にアスタリスクを続けて使用します（$*）。

program^file^name

INT：値：12

測定対象のプロセスが実行しているオブジェクトファイルの名前です。この名前は、左詰めにして空白で埋められ、ローカル内部形式である必要があります。*program^file^name* 内のボリューム、サブボリューム、およびファイル名の各フィールドを指定するには、アスタリスク（*）を使用します。

numcounters

INT：値：1

下記の COUNTER^DESC 記述子の数です。各 COUNTER^DESC 記述子は、測定対象のユーザー定義のカウンターについて記述します。

注記:

MEASREAD または MEASREADACTIVE プロシージャーに USERDEF^DESC 記述子を渡すときに、COUNTER^DESC 記述子を含めないでください。

COUNTER^DESC

ユーザー定義カウンター指定を記述します。OSS ファイルパス名も COUNTER^DESC を使用します。

```
Struct Counter^desc (*);
                        ! Numcounters of these structs go here
  Begin                 ! when used as a Contab descriptor
  Int    Name[0:7];     ! User assigned counter name
  Int    Type;          ! 1=accum, 2=busy, 3=queue
  Int    Index;         ! 0:index counters will be allocated
  End;
```

name

INT：値：8

左詰めの、空白で埋められたカウンター名です。

type

INT：値：1

これらのリテラル値のいずれかです。

ACCUM	累積カウンター、32 ビット
FACCUM	累積カウンター、64 ビット
BUSY	ビジーカウンター
QUEUE	キューカウンター
QBUSY	キュービジーカウンター
TCELLBUSY	ビジーカウンター、タイマーセルで維持

表は続く

TCELLQUE　キューカウンター、タイマーセルで維持
UE

TCELLQBU　キュービジーカウンター、タイマーセルで維持
SY

index

INT：値：1

このカウンター名に関連付けられたカウンターの配列に対処するために必要な最上位のインデックス値です。カウンターが単一の値の場合は、*index* を 0 に設定します。

USERDEF^OSS^DESC

G06.12 以降の RVU を実行しているシステム上の USERDEF エンティティの記述子です。USERDEF^OSS^DESC には、OSS パス名のサポートが含まれます。USERDEF^OSS^DESC も COUNTER^DESC とともに使用されます。

```
Struct Userdef^OSS^desc (*);
  Begin
  Int    Type;                          ! Entity type
  Int    Len;                           ! Byte length of record.
  Int    CPU^number;
  Int    PIN;
  Int    Process^name[0:3];
  Int    Program^file^name[0:11];
  Struct Program^file^name^MID;         ! Internal identifier for OSS pathname
    Begin
    Int    Pathid[0:11];                ! Internal format OSS file pathname
    Int    Crvsn[0:2];                  ! Creation Volume Sequence Number
    End;
  Int    Numcounters;                   ! Number of counters to be installed
  End;  ! A subsequent counter^desc is not needed for an Entitydesc of
        ! a previously installed entity. But for Contab use, as many
        ! Counter^desc as desired must immediately follow this struct.
```

次に、USERDEF^DESC 記述子とは異なる USERDEF^OSS^DESC フィールドの定義を示します。その他のすべての USERDEF^OSS^DESC フィールドの定義については、**USERDEF^DESC**(588 ページ)を参照してください。

PROGRAM^FILE^NAME^MID

OSS パス名の内部形式のファイル名が含まれる構造体です。

pathid

INT：値：12

指定したプログラムファイル名の OSS ファイルパス名の内部形式表現です。すべての PATHID を指定するには、PATHID[0]を-1 に初期化します。

crvsn

INT：値：3

OSS ディスクファイルの一意のインスタンスを識別する作成バージョンのシリアル番号です。すべての CRVSN 値を指定するには、CRVSN[0]を-1 に初期化します。

WAN^DESC

G シリーズ以降の RVU を実行しているシステムでの LINE、NETLINE、または TERMINAL エンティティの記述子です。

```
Struct  WAN^desc (*)           -- Line, Netline, and Terminal Descriptors
  Begin
  Int    Type;                 -- Entity type
  Int    Len;                  -- byte length of this structure
  Int    CPU^number;
  Int    Trackid[0:2];         -- Burned for SEEPROM
  String Trackid^s = trackid;
  Int    Clip;                 -- 1 to 3 for SWAN, 1 to 6 for SWAN II
  Int    Line;                 -- 0 or 1
  Int    Device^name[0:11];
  String Device^name^s = device^name;
  Int    Device^type;
  Int    Device^subtype;
  End;
```

type

INT：値：1

次のいずれかです。

エンティティタイプ	リテラル	数値識別子
LINE	LINE^T	9
NETLINE	NETLINE^T	10
TERMINAL	TERM^T	13

正の *type* 値には、測定の構成に指定されたエンティティが含まれます。負の *type* 値は、指定されたエンティティを測定から除外します。例については、**エンティティ記述子の指定**(461 ページ)を参照してください。

len

INT：値：1

この記述子の長さ（バイト単位）です。

cpu^number

INT：値：1

測定する回線が構成されている CPU の番号です。すべての CPU を指定するには、リテラル-1 を使用します。

trackid

INT：値：3

SCF サブシステムにより 3880 コントローラーについて報告された（および通常は外部ラベルにも印刷される）6 バイトの ASCII 識別子です。すべての TRACKID を指定するには、最初のバイトでアスタリスク（*）を使用し、残りのフィールドを空白で埋めます。

clip

INT：値：1

SWAN コントローラー内で測定されている通信回線インターフェイスプロセッサー（CLIP）の数です（SWAN I では 1 から 3、SWAN II では 1 から 6）。すべての CLIP を指定するには、リテラル-1 を使用します。

line

INT：値：1

SWAN コントローラー内で CLIP により制御される特定の回線です（0 または 1）。両方の回線を表示するには、リテラル-1 を使用します。

device^name

INT：値：12

測定されている回線のデバイス名です。この名前は、左詰めにして空白で埋める必要があります。ドル記号（$）で開始し、内部形式である必要があります。すべてのデバイスを指定するには、ドル記号にアスタリスクを続けて使用します（$*）。

device^name^s

デバイス名の文字列表現です。

device^type

INT：値：1

SCF LISTDEV リストで報告されているように、61 (X.25)などの特定のサブシステムのデバイスタイプです。すべてのデバイスを指定するには、リテラル-1 を使用します。

device^subtype

INT：値：1

SCF LISTDEV リストに報告されているように、62 (X25AM)などの特定のサブシステムのデバイスサブタイプです。すべてのデバイスを指定するには、リテラル-1 を使用します。

Web サイト

全般的な Web サイト

Hewlett Packard Enterprise Information Library

http://www.hpe.com/info/EIL

Single Point of Connectivity Knowledge（SPOCK）Storage 互換性マトリックス

http://www.hpe.com/storage/spock

ストレージのホワイトペーパーおよび分析レポート

http://www.hpe.com/storage/whitepapers

その他の Web サイトについては、**サポートと他のリソース**を参照してください。

サポートと他のリソース

Hewlett Packard Enterprise サポートへのアクセス

- ライブアシスタンスについては、Contact Hewlett Packard Enterprise Worldwide の Web サイトにアクセスします。

  http://www.hpe.com/assistance
- ドキュメントとサポートサービスにアクセスするには、Hewlett Packard Enterprise サポートセンターの Web サイトにアクセスします。

  http://www.hpe.com/support/hpesc

ご用意いただく情報

- テクニカルサポート登録番号（該当する場合）
- 製品名、モデルまたはバージョン、シリアル番号
- オペレーティングシステム名およびバージョン
- ファームウェアバージョン
- エラーメッセージ
- 製品固有のレポートおよびログ
- アドオン製品またはコンポーネント
- 他社製品またはコンポーネント

アップデートへのアクセス

- 一部のソフトウェア製品では、その製品のインターフェイスを介してソフトウェアアップデートにアクセスするためのメカニズムが提供されます。製品のドキュメントを確認し、推奨されるソフトウェアアップデートの方法を特定します。
- 製品のアップデートをダウンロードするには、以下のいずれかに移動します。

  Hewlett Packard Enterprise サポートセンター

  http://www.hpe.com/support/hpesc

  Hewlett Packard Enterprise サポートセンター：ソフトウェアのダウンロード

  http://www.hpe.com/support/downloads

  Software Depot

  http://www.hpe.com/support/softwaredepot
- eNewsletters およびアラートにサブスクライブするには、以下の Web サイトにアクセスします。

  http://www.hpe.com/support/e-updates-ja
- お客様の権利の表示や更新を行ったり、契約と保証をプロフィールとリンクさせたりするには、Hewlett Packard Enterprise サポートセンターの **More Information on Access to Support Materials** ページをご覧ください。

  http://www.hpe.com/support/AccessToSupportMaterials

重要:

一部のアップデートにアクセスするには、Hewlett Packard Enterprise サポートセンターからアクセスするときに製品の権利付与情報が必要になる場合があります。関連する権利付与情報を使って HP パスポートをセットアップしておく必要があります。

カスタマーセルフリペア（CSR）

Hewlett Packard Enterprise カスタマーセルフリペア（CSR）プログラムでは、ご使用の製品をお客様ご自身で修理することができます。CSR 部品を交換する必要がある場合、お客様のご都合のよいときに交換できるよう直接配送されます。一部の部品は CSR の対象になりません。Hewlett Packard Enterprise もしくはその正規保守代理店が、CSR によって修理可能かどうかを判断します。

リモートサポート（HPE 通報サービス）

リモートサポートは、保証またはサポート契約の一部としてサポートデバイスでご利用いただけます。リモートサポートは、インテリジェントなイベント診断を提供し、ハードウェアイベントを Hewlett Packard Enterprise に安全な方法で自動通知します。これにより、ご使用の製品のサービスレベルに基づいて、迅速かつ正確な解決が行われます。ご使用のデバイスをリモートサポートに登録することを強くおすすめします。

ご使用の製品にリモートサポートの追加詳細情報が含まれる場合は、検索を使用してその情報を見つけてください。

リモートサポートおよびプロアクティブケア情報

HPE 通報サービス

http://www.hpe.com/jp/hpalert

HPE プロアクティブ ケアサービス

http://www.hpe.com/services/proactivecare-ja

HPE プロアクティブケアサービス：サポートされている製品のリスト

http://www.hpe.com/services/proactivecaresupportedproducts（英語）

HPE プロアクティブケアアドバンストサービス：サポートされている製品のリスト

http://www.hpe.com/services/proactivecareadvancedsupportedproducts

保証情報

ご使用の製品の保証を確認するには、Hewlett Packard Enterprise サポートセンターで入手できる *サーバー、ストレージ、電源、ネットワーク、およびラック製品の安全と準拠に関する情報*を参照します。

http://www.hpe.com/support/Safety-Compliance-EnterpriseProducts

追加保証情報

HPE ProLiant と x86 サーバーおよびオプション

http://www.hpe.com/support/ProLiantServers-Warranties

HPE エンタープライズサーバー

http://www.hpe.com/support/EnterpriseServers-Warranties

HPE ストレージ製品

http://www.hpe.com/support/Storage-Warranties

HPE ネットワーク製品

http://www.hpe.com/support/Networking-Warranties

規定に関する情報

安全、環境、および規定に関する情報については、Hewlett Packard Enterprise サポートセンターから *サーバー、ストレージ、電源、ネットワーク、およびラック製品の安全と準拠に関する情報*を参照してください。

http://www.hpe.com/support/Safety-Compliance-EnterpriseProducts

規定に関する追加情報

Hewlett Packard Enterprise は、REACH（欧州議会と欧州理事会の規則 EC No 1907/2006）のような法的な要求事項に準拠する必要に応じて、弊社製品の含有化学物質に関する情報をお客様に提供することに全力で取り組んでいます。この製品の含有化学物質情報レポートは、次を参照してください。

http://www.hpe.com/info/reach

RoHS、REACH を含む Hewlett Packard Enterprise 製品の環境と安全に関する情報と準拠のデータについては、次を参照してください。

http://www.hpe.com/info/ecodata

社内プログラム、製品のリサイクル、エネルギー効率などの Hewlett Packard Enterprise の環境に関する情報については、次を参照してください。

http://www.hpe.com/info/environment

ドキュメントに関するご意見、ご指摘

Hewlett Packard Enterprise では、お客様により良いドキュメントを提供するように努めています。ドキュメントを改善するために役立てさせていただきますので、何らかの誤り、提案、コメントなどがございましたら、ドキュメントフィードバック担当（**docsfeedback@hpe.com**）へお寄せください。この電子メールには、ドキュメントのタイトル、部品番号、版数、およびドキュメントの表紙に記載されている刊行日をご記載ください。オンラインヘルプの内容に関するフィードバックの場合は、製品名、製品のバージョン、ヘルプの版数、およびご利用規約ページに記載されている刊行日もお知らせください。

エラーメッセージ

Measure サブシステムは、エラーメッセージとエラーコードを生成します。この付録では、MEASCOM（Measure のコマンド言語インターフェイス）によって生成されるエラーメッセージについて説明します。Measure の呼び出し可能プロシージャーによって生成されるエラーコードの説明については、**エラーコード**(627 ページ)を参照してください。

この付録では、MEAS とメッセージ番号に続いて各メッセージを示しています。MEASCOM の使用中にエラーが発生した場合は、次のように入力してください。

```
HELP error-number
```

MEAS 2000

```
Comment. For more information type HELP number.
```

メッセージの説明

原因

MEASCOM では、HELP *number* と入力することで特定の MEASURE メッセージについての詳細情報を取得できます。

影響

このメッセージは、現在の処理に影響はありません。

リカバリ

何も必要ありません。このコメントが表示されないようにするには、COMMENTS SUPPRESS 2000 と入力してください。

MEAS 3000

```
WARNING. System system-name does not exist; check system name.
```

メッセージの説明

原因

指定されたシステムがネットワークに認識されませんでした。

影響

コマンドは実行されますが、該当するシステムの番号または名前を記述子に追加できませんでした。

リカバリ

目的のシステムを正しく指定したか確認してください。

MEAS 3001

```
WARNING. Device dev-name does not exist; check spelling.
```

メッセージの説明

原因

指定されたデバイスが見つからなかったか、コマンドに対して有効なデバイスタイプではありません。

影響

コマンドは実行されますが、システムの現在の状態ではデバイスは測定されません。

リカバリ

目的のデバイスが正しく指定されていること、またはデバイスが正しいタイプであることを確認してください。

MEAS 3002

```
WARNING. Measurement was not started due to reason listed above.
```

メッセージの説明

原因

START MEASUREMENT コマンドがその処理中のある時点で失敗しました。

影響

失敗した理由は、呼び出し可能プロシージャーから返されたエラーとして一覧表示されます。

リカバリ

HELP *error code number* と入力して適切な修正方法を確認し、操作を再度実行してください。（ALLOCATESEGMENT について詳しくは、*Guardian プロシージャーコール・リファレンス マニュアル*を参照してください。）

MEAS 3003

```
WARNING. Measurement was not stopped due to reason listed above.
```

メッセージの説明

原因

STOP MEASUREMENT コマンドがその処理中のある時点で失敗しました。

影響

失敗した理由は、呼び出し可能プロシージャーから返されたエラーとして一覧表示されます。

リカバリ

HELP *error code number* と入力して適切な修正方法を確認し、操作を再度実行してください。（ALLOCATESEGMENT について詳しくは、*Guardian プロシージャーコール・リファレンス マニュアル*を参照してください。）

MEAS 3004

```
WARNING. Logging was not on so command had no effect.
```

メッセージの説明

原因

LOG STOP が発行されましたが、ログが有効ではありませんでした。

影響

要求は無視されます。

リカバリ

通知メッセージです。修正処置は必要ありません。

MEAS 3005

```
WARNING. Changing current plot file to current data file.
```

メッセージの説明

原因

ADD PLOT が発行されましたが、プロットに関連付けられている古いデータファイルが、現在のデータファイルから一覧表示されたものと同じではありませんでした。

影響

前回のプロットの内容が失われます。

リカバリ

通知メッセージです。修正処置は必要ありません。

MEAS 3006

```
WARNING. Volume not network-accessible with current system default.
```

メッセージの説明

原因

リモートシステムのデフォルトとして、デフォルトの 7 文字のボリュームが有効になっていました。

影響

現在のデフォルトによって拡張されるすべてのファイル名が無効になります。

リカバリ

デフォルトのシステムまたはボリュームを変更してください。

MEAS 3007

```
WARNING. No entities present to ADD; LIST records before trying ADD.
```

メッセージの説明

原因

追加できるレコードがないのに、ADD PLOT が実行されました。

影響

要求は無視されます。

リカバリ

プロットへの追加を試みる前に、LIST *entity* コマンドを実行して目的のレコードのリストを取得してください。

MEAS 3008

```
WARNING. Plot became full during ADD; there can be a maximum of 26 entities;
some entities were not ADDed.
```

メッセージの説明

原因

26 個を超える項目をプロットに追加しようとしました。

影響

要求した項目の一部は追加されませんでした。

リカバリ

目的のカウンターを追加する前に、不要な項目を削除して 26 個以下にしてください。

MEAS 3009

WARNING. Item not found -- *item-name*.

メッセージの説明

原因

指定されたプロット項目が見つからなかったか、プロット文字をコマンドでも使用した（たとえば、DELETE PLOT CPU-BUSY-TIME (A)）場合は、カウンター A が適切な項目名ではありませんでした。

影響

要求した操作は無視されます。

リカバリ

項目のスペルが正しいことと、正しいプロット文字を入力したことを確認してください。

MEAS 3010

WARNING. Measurement not found -- *filename*.

メッセージの説明

原因

指定されたデータファイルが見つかりませんでした。

影響

要求した操作は無視されます。

リカバリ

ファイル名のスペルが正しいことと、指定されたファイルが追加されたことを確認してください。

MEAS 3011

WARNING. Counter not found -- *counter-name*.

メッセージの説明

原因

指定されたカウンターが見つかりませんでした。

影響

要求した操作は無視されます。

リカバリ

カウンター名のスペルが正しいことと、指定されたカウンターが追加されたことを確認してください。

MEAS 3012

WARNING. Measfh option ignored; DELETE then ADD to change Measfh.

メッセージの説明

原因

以前に追加されたデータファイルに対して ADD MEASUREMENT コマンドが発行されましたが、MEASFH プロセスがまだアクティブです。

影響

最新の ADD MEASUREMENT コマンドが現在のデータファイルを変更しましたが、MEASFH プログラムファイル名が未変更のままです。

リカバリ

セッション中に同じデータファイルで別の MEASFH プロセスを使用するには、DELETE MEASUREMENT コマンドを使用して現在の MEASFH プロセスを削除してから、ADD MEASUREMENT コマンドを再発行して新しい MEASFH プロセスを指定してください。

MEAS 3013

```
WARNING. Measurement in the current data file is from current-datafile-
version, which is a later release than this version (current-meascom-version)
of MEASCOM.
```

メッセージの説明

原因

現在のデータファイルでの測定が、実行中の MEASCOM のバージョンよりも後の RVU のものです。これ以降の RVU は外部レコード形式が変更されている可能性があります。

影響

データファイルが追加されます。

リカバリ

2 つの RVU 間で外部レコード形式に違いがないことが確実な場合、この警告メッセージは無視できます。そうでない場合は、データファイル内のものと同じかそれ以降の RVU の MEASCOM を使用してください。

MEAS 3014

```
WARNING. The specified Loadid has more than 5 characters, sequencing will be
suppressed.
```

メッセージの説明

原因

LISTALL コマンドを使用したときに、各測定間隔を指定するために 3 桁のシーケンス ID が LOADID に追加されました。3 文字は使用できなかったために、これは抑止されました。

影響

シーケンスは抑止されます。

リカバリ

LOADID には 5 文字以下の文字を使用してください。

MEAS 3015

```
WARNING. MEASCOM is attempting to allocate or resize its extended segment to
be number bytes.
```

メッセージの説明

原因

MEASCOM が 16 MB より大きい拡張セグメントを要求しています。

影響

システムエラーが発生しない限り（発生した場合はエラーが表示される）、コマンドは続行されます。

リカバリ

セグメントが大きすぎると考えられる場合は、MEASCOM を停止し、MEASCOM に返されるレコードの数を減らすように LIST コマンドを指定してください。（たとえば、IF 句を使用してください。）

MEAS 3016

WARNING. The item *item-name* is obsolete for the current datafile version; results of the operation may be unexpected.

メッセージの説明

原因

入力した項目名が処理中のデータファイルで保守されなくなっています。

影響

コマンドは続行されますが、この操作は予期しない結果を招く場合があります。

リカバリ

処理中のデータファイルの PVU に対して有効な項目名を使用してください。

MEAS 3017

WARNING. The specified file name was resolved using the current default system (*current-system-name*), which is different from the system where the data file was collected (*system-name*). Therefore, this command may not LIST any records.

メッセージの説明

原因

LIST FILE コマンドで解決されたファイル名で現在のデフォルトのシステムが使用されましたが、データファイルが収集されたシステムとは異なっています。

影響

コマンドは続行されますが、おそらくデータは返されません。

リカバリ

LIST コマンドにシステム名とファイル名を入力してください。たとえば、LIST FILE \A.$B.C.D と指定するか、SYSTEM \A と指定してから LIST コマンドを繰り返してください。

MEAS 3018

WARNING. TO and FOR are both set -- FOR will be ignored; if FOR is desired then RESET TO.

メッセージの説明

原因

SHOW または SET コマンドの実行中に表示される場合は、デフォルトの属性に TO と FOR の両方のオプションの値が含まれています。LIST コマンドの実行中に表示される場合は、デフォルトの属性とコマンドライン属性によって、TO と FOR のオプションが両方とも設定されました。

影響

コマンドは続行されますが、FOR オプションは無視されます。

リカバリ

TO オプションを使用するには、リカバリは不要です。それ以外の場合は、RESET コマンドを使用して TO の値をクリアして、FOR の値が使用されるようにしてください。

MEAS 3019

WARNING. MEASCTL's current SWAP volume is *volume*; do NOT bring *volume* down while MEASURE is still running.

メッセージの説明

原因

MEASCTL が$SYSTEM 以外の SWAP ボリュームで起動されました。MEASCTL の SWAP ボリュームが使用不能になった場合（たとえば、PUP DOWN 操作が実行された場合）、システム全体が停止します。

影響

コマンドは実行されます。

リカバリ

リカバリは必須ではありません。ディスク領域によって可能な場合は、MEASURE サブシステムを停止し、SWAP ボリュームとして$SYSTEM（またはミラーボリューム）を使用して再起動してください。次のコマンドを使用してください。

```
MEASCOM /SWAP $SYSTEM/ START MEASSUBSYS
```

MEAS 3020

```
WARNING. No current data file; select a current data file using ADD.
```

メッセージの説明

原因

実行されたコマンドが、その機能を実行するために現在の測定データファイルを必要としていました。

影響

要求した操作は無視されます。

リカバリ

必要なデータファイルで ADD MEASUREMENT を実行してください。

MEAS 3021

```
WARNING. Measurement not active; unable to perform request.
```

メッセージの説明

原因

このコマンドは、測定が現在アクティブな場合にのみ有効です。

影響

要求した操作は無視されます。

リカバリ

通知メッセージです。修正処置は必要ありません。

MEAS 3022

```
WARNING. No more configuration data is available.
```

メッセージの説明

原因

要求されたタイプで構成されているエンティティがこれ以上ありません。

影響

要求した操作は無視されます。

リカバリ

検索を再開するか、別のエンティティタイプに変更してください。

MEAS 3023

```
WARNING. CRVSN was not specified, translation may not be correct.
```

メッセージの説明

原因

LISTPNAME コマンドで CRVSN 修飾子を指定せずに Guardian ファイル名が指定されました。

影響

OSS ファイルパス名への Guardian ファイル名の変換は試みられますが、指定されたファイル名が一時的なエンティティの場合は、報告される名前が正しくない可能性があります。

リカバリ

Measure エンティティレポートで報告される CRVSN 値を含めてください。

MEAS 3024

```
WARNING. OSS journal segment is under construction, not attached.
```

メッセージの説明

原因

測定のシャットダウン時にまだ OSS ジャーナルセグメントを構築している最中だったデータファイルに対して、ADD MEASUREMENT コマンドが入力されました。

影響

ファイルは正常に開きますが、OSS ジャーナルセグメントは接続されません。

リカバリ

通常は、処置は不要です。変換は、現在のシステム上の OSS ファイルシステムの名前空間から処理できます。後で OSS ジャーナルセグメントにアクセスする必要がある場合は、OSS ジャーナルセグメントの構築が完了した後、DELETE MEASUREMENT および ADD MEASUREMENT コマンドを使用して、Measure データファイルを削除してから、もう一度追加してください。

MEAS 3025

```
WARNING. Default Catalog not set.
```

メッセージの説明

原因

デフォルトの SQL カタログを指定せずに SQLSCHEMA コマンドが発行されていました。デフォルトの SQL カタログがまだ設定されていませんでした。

影響

デフォルトの SQL スキーマは設定されていますが、SQLCATALOG コマンドが発行されていない場合、後続のコマンドで使用する ANSI 名がエラーで失敗する可能性があります。

リカバリ

SQLCATALOG コマンドを発行して、デフォルトの SQL カタログを設定してください。

MEAS 3026

```
WARNING. Counter data records have been suppressed; active counter data
records are not available for PROCESSH.
```

メッセージの説明

原因

PROCESSH エンティティが含まれている測定に対して START MEASUREMENT コマンドが入力されて、カウンターデータレコードの抑制が選択されました。

影響

コマンドは続行されますが、PROCESSH レコードは測定に使用できません。

リカバリ

PROCESSH レコードが必要な場合は、測定を停止し、カウンターデータレコードを抑制せずに再開してください。PROCESSH レコードが必要ない場合、修正処置は必要ありません。

MEAS 3027

```
WARNING. Counter data records have been suppressed; OSS and/or SQL journal
segments will not be produced.
```

メッセージの説明

原因

OSS または SQL（あるいはその両方）のジャーナルセグメントが選択されている測定に対して START MEASUREMENT コマンドが入力されて、カウンターデータレコードの抑制が選択されました。

影響

コマンドは続行されますが、測定用の OSS または SQL（あるいはその両方）のジャーナルセグメントは生成されません。

リカバリ

OSS または SQL（あるいはその両方）のジャーナルセグメントが必要な場合は、測定を停止し、カウンターデータレコードを抑制せずに再開してください。OSS または SQL（あるいはその両方）のジャーナルセグメントが必要でない場合、修正処置は必要ありません。

MEAS 3050

```
Exceeded depth of 4 OBEY files; rework OBEY nesting.
```

メッセージの説明

原因

5 つ以上の OBEY ファイルを含めようとしました。

影響

コマンドは中断され、すべての OBEY ファイルが閉じられます。

リカバリ

OBEY ファイルのネストを修正して、5 つ以上のファイルが使用されないようにしてください。

MEAS 3051

```
Expecting a name of nn characters or less; shorten name.
```

メッセージの説明

原因

入力する文字列に含める文字数は、多くても、エラーメッセージで示される文字数である必要があります。

影響

要求した操作は無視されます。

リカバリ

文字列を短くして適切な長さにし、コマンドを再試行してください。

MEAS 3052

File is not network-accessible; shorten volume name.

メッセージの説明

原因

リモートアクセスのために、8 文字以上の文字が含まれるボリューム名でファイル名が指定されました。

影響

要求した操作は無視されます。

リカバリ

リモートアクセスの場合、ボリューム名に 8 文字以上を含めることはできません。ボリューム名を短くして適切な長さにしてください。

MEAS 3053

Value out of range; value must be 3000:3049, 3116, 3233, 3276, 3277 or 3299.

メッセージの説明

原因

入力した数が現在の項目の許容範囲外でした。

影響

要求した操作は無視されます。

リカバリ

エラーメッセージで示された範囲内の値を入力してください。

MEAS 3054

System does not exist; check system name.

メッセージの説明

原因

要求で指定されたシステムが存在しません。

影響

ファイル名は拡張できないため、要求した操作は無視されます。

リカバリ

システム名のスペルが正しいことを確認してください。

MEAS 3055

Illegal syntax; see HELP or the Measure Reference Manual.

メッセージの説明

原因

パーサーで構文エラーが発生しました。

影響

要求した操作は無視されます。

リカバリ

HELP またはこのマニュアルの構文に従って、構文を修正してください。

コマンド構文の説明については、**MEASCOM コマンド**(36 ページ)を参照してください。測定対象のエンティティを識別するための構文については、**エンティティおよびカウンター**(167 ページ)参照してください。

MEAS 3056

```
Subsystem error; please report to Tandem Computers Incorporated. Trap %n, CS
= %n, P = %nnnnn, E = %nnnnn, L = %nnnnn, S = %nnnnn.
```

メッセージの説明

原因

MEASCOM によってトラップが発生しました。

影響

要求した操作は無視されます。

リカバリ

エラーメッセージに示された情報のコピーをとって、Hewlett Packard Enterprise の担当者にお問い合わせください。

MEAS 3057

```
Ambiguous DELETE command; specify which occurrence to delete.
```

メッセージの説明

原因

DELETE PLOT が要求されましたが、入力したカウンター名がプロット定義に複数回出現しています。

影響

要求した操作は無視されます。

リカバリ

コマンドを繰り返して、プロット文字を付加して特定の出現箇所を指定するか、アスタリスク（*）を付加してすべての出現箇所を指定してください。

MEAS 3058

```
Expecting a valid process or file name.
```

メッセージの説明

原因

MEASCOM が有効なプロセス名（たとえば、$XMM）または有効なファイル名（たとえば、$*disk.subvol.file*、または$*disk.subvol.**）を期待していました。

影響

要求した操作は無視されます。

リカバリ

有効なプロセスまたはファイル名を使用して要求を再発行してください。

MEAS 3059

Only an individual file can be specified; replace *'s.

メッセージの説明

原因

コマンドに対して単一のファイル名が入力されませんでした。

影響

要求した操作は無視されます。

リカバリ

ワイルドカード文字（*）を特定のファイル名と置き換えてください。

MEAS 3060

Expecting a disk file name; ensure *filename* is a valid disk file name.

メッセージの説明

原因

コマンドに恒久的なディスクファイルが指定されませんでした。

影響

要求した操作は無視されます。

リカバリ

入力した名前が$*a.b.c* の形式になっていることを確認してください。

MEAS 3061

The USERDEF for this counter was not found; ADD the USERDEF.

メッセージの説明

原因

現在の ADD COUNTER コマンドに入力されたプロセス指定が、ADD USERDEF コマンドによって以前に追加されていませんでした。

影響

要求した操作は無視されます。

リカバリ

プロセス指定が正しいことを確認するか、USERDEF を追加して、操作を再度実行してください。

MEAS 3062

Counter already exists; DELETE and then ADD to change counter.

メッセージの説明

原因

すでに存在しているカウンターを追加しようとしました。

影響

要求した操作は無視されます。

リカバリ

カウンタータイプまたは配列インデックスを変更するには、カウンターを削除してから追加してください。

MEAS 3063

No more data files can be added; DELETE unnecessary files.

メッセージの説明

原因

65 個以上のデータファイルを追加しようとしました。

影響

要求した操作は無視されます。

リカバリ

不要なデータファイルを削除してから、必要なものを追加してください。

MEAS 3064

Expecting a disk file name; ensure *file* is a valid disk filename.

メッセージの説明

原因

コマンドに有効または一意のディスクファイルが指定されませんでした。

影響

要求した操作は無視されます。

リカバリ

入力した名前が$*a.b.c* または$*a.#x* の形式になっていることを確認してください。

MEAS 3065

File must be local; see Measure Reference Manual.

メッセージの説明

原因

指定または拡張されたファイル名はリモートで、コマンドにはローカルファイル名が必要でした。

影響

要求した操作は無視されます。

リカバリ

このコマンドで使用されるファイル名の説明については、**MEASCOM コマンド**(36 ページ)の特定のコマンドの説明を参照してください。その後、ローカルファイルを指定して、操作を再度実行してください。

MEAS 3066

The name specified in the IF condition (*item-name*) was not found; check the spelling of the name.

メッセージの説明

原因

IF 条件で使用された項目名が、リストされているエンティティタイプに関連付けられた項目名ではありませんでした。

影響

要求した操作は無視されます。

リカバリ

項目のスペルを確認し、項目名が存在していることを確認してから、操作を再度実行してください。

MEAS 3067

The name specified in the BY clause (*item-name*) was not found; check the spelling of the name.

メッセージの説明

原因

BY 条件で使用された項目名が、リストされているエンティティタイプに関連付けられた項目名ではありませんでした。

影響

要求した操作は無視されます。

リカバリ

項目のスペルを確認し、項目名が存在していることを確認してから、操作を再度実行してください。

MEAS 3068

No current data file; select a current data file using ADD.

メッセージの説明

原因

実行されたコマンドが、その機能を実行するために現在の測定データファイルを必要としていました。

影響

要求した操作は無視されます。

リカバリ

必要なデータファイルで ADD MEASUREMENT を実行してください。

MEAS 3069

Command option has been repeated; delete one and reissue command.

メッセージの説明

原因

オプションが同じコマンドで複数回指定されました。

影響

要求した操作は無視されます。

リカバリ

このオプションの 1 つの指定を除いてすべて削除してから、コマンドを繰り返してください。

MEAS 3070

Both FOR and TO have been specified; delete one and reissue command.

メッセージの説明

原因

同じコマンドに FOR と TO の両方のオプションが指定されました。一度に指定できるオプションは 1 つだけです。

影響

要求した操作は無視されます。

リカバリ

オプションのいずれかを削除して、コマンドを繰り返してください。

MEAS 3071

item-name not found; check the spelling and/or the appropriate entity report.

メッセージの説明

原因

入力されたプロットカウンター名が見つかりませんでした。

影響

要求した操作は無視されます。

リカバリ

スペルを確認し、適切なレポートを使用して、入力した名前が有効な項目名であることを確認してください。Legacy 形式のレポート項目は同等の ZMS 形式のレポートには表示されなくなっていて、Legacy 形式のインターフェイスでしかプロットできません。また、ZMS 形式のレポート項目は同等の Legacy 形式のレポートには表示されなくなっていて、ZMS 形式のインターフェイスでしかプロットできません。

MEAS 3072

Measurement not active; unable to perform request.

メッセージの説明

原因

その時点で測定がアクティブな場合にのみ有効なコマンドを発行しようとしました。

影響

要求した操作は無視されます。

リカバリ

通知メッセージです。修正処置は必要ありません。

MEAS 3073

A remote filename is not allowed when dealing with DISCOPENs or DISKFILEs.

メッセージの説明

原因

DISCOPEN または DISKFILE コマンドに対して（現在の SYSTEM を使用して）入力または拡張されたファイル名が、リモートファイルの名前でした。

影響

要求した操作は無視されます。

リカバリ

DISCOPEN と DISKFILE は常にローカルです。DISCOPEN の場合、ファイル名の後にリモートオープナーを指定できます（たとえば、$a.b.c (\sys)）。DISKFILE の場合、オープナー情報はエンティティと関連付けられていないため、リモートオープナーは許可されません。

MEAS 3074

```
Plot list is full; must DELETE an item before ADD.
```

メッセージの説明

原因

すでにいっぱいになっているプロットに項目を追加しようとしました。

影響

要求した操作は無視されます。

リカバリ

いくつかのカウンターを削除してから、ADD を再度実行してください。

MEAS 3075

```
Expecting a single alpha character; see HELP.
```

メッセージの説明

原因

プロット文字が 1 つのアルファベット文字ではありませんでした。

影響

要求した操作は無視されます。

リカバリ

1 つのアルファベット文字だけを入力してください。詳細を表示するには、HELP と入力してください。

MEAS 3079

```
Measurement space overflow; DELETE unneeded descriptors.
```

メッセージの説明

原因

ADD の実行中に測定領域がいっぱいになりました。

影響

要求した操作は無視されます。

リカバリ

測定領域から不要な記述子を削除して、操作を再度実行してください。

MEAS 3080

Allocate segment error *number* on work segment.

メッセージの説明

原因

MEASCOM プライマリ作業セグメント上の割り当てセグメントから、エラーが発生して返されました。

影響

要求した操作は無視されます。

リカバリ

MEASCOM はセグメントなしでも実行できますが、実行できるのは制限された容量でのみです。可能な場合は割り当てセグメントで発生した問題を修正して、MEASCOM を再度実行してください。

MEAS 3081

`The IF clause on an identifier field (i.e. name) is not supported for the current datafile version.`

メッセージの説明

原因

非カウンター項目に対する IF 句の使用は、D10 より前のデータファイルではサポートされていません。

影響

要求した操作は中断されます。

リカバリ

カウンター項目に対応する項目名を使用するか、IF 句を削除してください。

MEAS 3083

`Internal buffer full; try making command less complex.`

メッセージの説明

原因

Measure がコマンドを展開したときに、バッファーがいっぱいになりました。

影響

要求したコマンドは無視されます。

リカバリ

コマンドをいくつかのコマンドに分割して、コマンドを別々に実行するか、MEASCOM が最小限の拡張を実行するように、現在のファイル名を修飾してください。

MEAS 3084

`The DISCOPEN and DISKFILE entities require the filename to be either a permanent or temporary disk filename (e.g. $a.b.* or $system.#*).`

メッセージの説明

原因

現在のファイル名が有効なディスクファイル名ではありません。

影響

要求したコマンドは実行されません。

リカバリ

入力されたファイル名を、有効な恒久的または一時的なディスクファイル名に変更してください。

MEAS 3085

`"char" not allowed in current string; delete symbol and retry.`

メッセージの説明

原因

エラーメッセージで表示される記号が、入力した文字列の有効な単語区切り記号ではありません。

影響

要求した操作は無視されます。

リカバリ

この文字を削除するか、有効な単語区切り文字に置き換えてから、コマンドを再度実行してください。

MEAS 3086

```
START attempted on an ADDed datafile; DELETE and then START.
```

メッセージの説明

原因

一覧表示のために追加されたデータファイルに対して START を試みました。START コマンドによってデータがパージされるため、コマンドは中止されます。MEASCOM 環境では、コマンド START、STOP、または ADD によってデータファイルが追加されます。このメッセージは、データが誤ってパージされるのを防ぐための試みです。

影響

要求した操作は中断されます。

リカバリ

START 操作がまだ必要な場合は、データファイルを DELETE して、START を再度実行してください。

MEAS 3087

```
Expecting Help key length of 30 characters or less; shorten string.
```

メッセージの説明

原因

HELP コマンドで入力したキー文字列が 30 文字を超えています。

影響

要求した操作は実行されません。

リカバリ

文字列を短くしてください。

MEAS 3089

```
Expecting a number with 3 or less decimal digits.
```

メッセージの説明

原因

3 桁を超える数字を入力しました。

影響

要求した操作は実行されません。

リカバリ

桁数を減らしてから、コマンドを再入力してください。

MEAS 3090

Incompatible PROCESSH records in the *filename* data file. Please use a proper version of MEASCOM.

メッセージの説明

原因

データファイルの PROCESSH レコード形式が、現在実行中の MEASCOM と互換性がありません。

影響

要求した操作は実行されません。

リカバリ

適切なバージョンの MEASCOM を取得してください。

MEAS 3091

The existing DDL format structured file for the entity specified in the LIST command has a different record size.

メッセージの説明

原因

レコードサイズがデータファイル内のサイズと異なっています。

影響

要求した操作は実行されません。

リカバリ

既存の構造化ファイルを削除して MEASCOM に作成させるか、別のサブボリュームを使用してください。対応する DDL ディクショナリを必ず再生成してください。

MEAS 3092

SYSTEM-PROCESSES is not allowed in this command.

メッセージの説明

原因

SQLPROC および SQLSTMT コマンドでは、SYSTEM-PROCESSES は許可されません。

影響

要求した操作は実行されません。

リカバリ

コマンドを適切な形式に変更して、再度実行してください。

MEAS 3094

Incompatible datafile version. The LISTALL command is not supported for this version of the datafile.

メッセージの説明

原因

C10 以前のデータファイルに対して LISTALL コマンドが発行されました。

影響

コマンドは中止されます。指定された MEASFH は要求を満たすことはできません。

リカバリ

別のコマンドを試してください。レコードを取得するには、LIST コマンドを使用してください。

MEAS 3095

Command option "*list-option*" not allowed with LISTALL command when STRUCTURED format is specified.

メッセージの説明

原因

LISTALL コマンドで不正な *list-option* が指定されました。

影響

コマンドは中止されます。指定したオプションは満たされません。

リカバリ

コマンドを再度実行し、エラーの原因となった *list-option* は含めないでください。

MEAS 3096

Command option "TOLERANCE OFF" not allowed with LIST command.

メッセージの説明

原因

C10 以前のデータファイルに対して TOLERANCE OFF オプションが指定されました。

影響

コマンドは中止されます。MEASFH は要求を満たすことはできません。

リカバリ

コマンドを再度実行し、TOLERANCE オプションは含めないでください。

MEAS 3097

Error *number* was reported earlier on this data file. Operation cannot be carried out.

メッセージの説明

原因

データファイルが追加されたときに、MEASFH がエラーまたは警告を報告しました。データファイルに問題がある可能性があります。

影響

コマンドは中止されます。コマンドを正しく実行できませんでした。

リカバリ

このデータファイルで別のコマンドを試してください。

MEAS 3098

FC and ! are not allowed on input longer than 132 characters; reenter the desired command in full.

メッセージの説明

原因

有効なレコードには最大 2100 文字を含めることができますが、FC コマンドを使用した変更または!コマンドを使用した再実行の候補となるのは、132 文字以下の有効なレコードのみです。

影響

コマンドは実行されません。

リカバリ

目的のコマンドは省略せずに入力する必要があります（長いコマンドを実行するには、エディターと OBEY コマンドを使用してください）。

MEAS 3099

`Line specified is not in the HISTORY buffer; check buffer and retry.`

メッセージの説明

原因

Measure が再実行しようとしている行が HISTORY バッファーにありません。

影響

コマンドは実行されません。

リカバリ

HISTORY コマンドを実行して、コマンドがまだ HISTORY バッファー内にあることを確認してください。そうでない場合は、コマンドを省略せずに再入力する必要があります。

MEAS 3100

`The command is too long, the maximum number of characters an effective record can contain is 2100 (`*`nnnn`* `were entered); try to enter multiple commands or remove embedded spaces as possible to shorten the command(s) entered.`

メッセージの説明

原因

アンパサンド（&）を使用した場合でも、MEASCOM がレコードをスキャンしてコマンドを実行する前に 1 つのレコードに入力できる文字数には制限があります。現在の入力ストリームは 2100 文字の制限を超えました。

影響

コマンドは実行されません。

リカバリ

複数のコマンドを入力した場合は、各コマンドを別々に入力してみてください。1 つのコマンドだけを入力した場合は、コマンドをいくつかのコマンドに分割して別々に実行してみてください。コマンドを分解できない場合は、埋め込みのスペースを削除してコマンドを短くしてみてください。

MEAS 3101

`filename` `was not started due to Process_Create_ error` *`number1`* `(`*`number2`*`); correct error and retry command.`

メッセージの説明

原因

示された Process_Create_エラーのため、指定されたプログラムファイルが起動されませんでした。

影響

コマンドは実行されません。

リカバリ

Process_Create_エラーの原因を修正して、コマンドを再度実行してください。

MEAS 3102

filename could not be opened due to File System error *number*.

メッセージの説明

原因

示されたファイルシステムエラーのため、指定されたプログラムファイルを開けませんでした。

影響

コマンドは実行されません。

リカバリ

ファイルシステムエラーの原因を修正して、コマンドを再度実行してください。

MEAS 3103

filename did not read the startup message due to File System error *number*.

メッセージの説明

原因

示されたファイルシステムエラーのため、指定されたプログラムファイルが送信された起動メッセージを読み取りませんでした。

影響

コマンドは実行されません。

リカバリ

ファイルシステムエラーの原因を修正して、コマンドを再度実行してください。

MEAS 3104

SCALE-FROM must be less than SCALE-TO; make sure the values are correct before retrying the LIST PLOT command.

メッセージの説明

原因

SCALE-FROM 属性の有効な値が、PLOT オブジェクトの SCALE-TO 属性以上の値でした。

影響

コマンドは実行されません。

リカバリ

SHOW PLOT を実行して、各属性のデフォルト値を確認してください。LIST PLOT コマンドラインで入力した属性は、デフォルト値をオーバーライドします。コマンドを再試行するときは、有効な値に整合性があることを確認してください。

MEAS 3105

Expecting one of the following keywords *keyword-list*.

メッセージの説明

原因

検出された単語が有効なキーワードではありません。

影響

コマンドは実行されません。

リカバリ

有効なキーワードを入力してください（いくつかはエラーテキストに表示されますが、リストが長すぎる場合はすべてが表示されるわけではありません）。

MEAS 3106

item-name is a valid item name, but it can only be used in the BY clause; use a different item name, or a counter name that appears within the entity report.

メッセージの説明

原因

BY 句でしか使用できない項目名が、IF 句で使用されました。

影響

コマンドは実行されません。

リカバリ

別の項目名を使用するか、エンティティレポートに示されるカウンター名を使用してください。

MEAS 3107

item-name is a valid item name, but it can only be used in the By and IF clauses; use a counter name that appears within the entity report.

メッセージの説明

原因

BY 句と IF 句でしか使用できない項目名が、PLOT コマンドで使用されました。

影響

コマンドは実行されません。

リカバリ

エンティティレポートに示されるカウンター名を使用してください。

MEAS 3108

item-name is not a unique item-name; do a HELP *entity-type* COUNTERS (e.g. HELP CPU COUNTERS) to determine the number of characters required to make the abbreviation unique.

メッセージの説明

原因

入力された項目名が複数の項目名と一致しました。

影響

コマンドは実行されません。

リカバリ

入力した名前に文字を追加してください。

MEAS 3109

item-name is a new item name that is not valid for the current datafile version; use a different item name.

メッセージの説明

原因

処理中の（ダウンリビジョンの）データファイルに含まれていない項目名が入力されました。

影響

コマンドは実行されません。

リカバリ

有効な項目名を使用してください。

MEAS 3110

The current string of digits contains a digit which is greater than the current base of *number*; delete the digit and retry the command.

メッセージの説明

原因

入力された数字が現在の基準には大きすぎます（たとえば、8 進数は 7 以下でなければならないため、ADD CPU %8 は誤りです）。

影響

コマンドは実行されません。

リカバリ

入力した数字を、使用している基準と一致するように変更してください。

MEAS 3111

Native Mode code maps do not have space numbers; if you wish to sample native mode maps specify either UCr, ULr, SCr, or SLr.

メッセージの説明

原因

ネイティブモードのコードマップに対して指定されたコードマップが正しくありません。

影響

コマンドは実行されません。

リカバリ

ネイティブモードのコードを測定するには、UCr、ULr、SCr、または SLr を指定してください。

MEAS 3112

The datafile version is *version* and cannot be analyzed with F40 Meascom.

メッセージの説明

原因

G シリーズ RVU を使用していますが、データファイルのバージョンが F40.00 の前のバージョンです。

影響

コマンドは実行されません。

リカバリ

F40.00 より前のデータファイルを分析するには、データファイルのバージョンと一致する F40.00 より前のバージョンの MEASCOM と、適切な MEASFH を使用してください。

MEAS 3113

```
Unable to open data file.
```

メッセージの説明

原因

システムが扱うデータが多すぎたため、MEASFH がデータファイルを開くことができませんでした（ファイルシステムエラー 21）。

影響

コマンドは実行されません。

リカバリ

コマンドを再試行してください。

MEAS 3114

```
A journal segment is required but not available for request.
```

メッセージの説明

原因

現在のノードで収集されなかったデータファイルに対してコマンドが入力されたため、それに対して現在使用可能なジャーナルセグメントがありません。

影響

コマンドは実行されません。

リカバリ

必要な変換が追加したデータファイルからの変換ではない場合は、測定を DELETE して、変換が現在のシステムから実行されるようにコマンドを再発行してください。別のノードで収集されたデータの変換が必要な場合は、測定を開始するときに OSS または SQL オプション、あるいはその両方を指定してください。

MEAS 3115

```
PAGESIZE must be specified in the range of 6 and 128 inclusive.
```

メッセージの説明

原因

無効な PAGESIZE 設定が指定されました。

影響

要求した操作は中断されます。

リカバリ

6～128 までの値を指定してください。

MEAS 3116

```
Unable to translate OSS file pathname.
```

メッセージの説明

原因

要求された変換が、OSS 名前空間または現在の OSS ジャーナルセグメントで見つかりませんでした。

影響

コマンドは実行されません。

リカバリ

要求が OSS ジャーナルセグメントを含む ADD データファイルの一部である場合は、現在のシステムが ADD データファイルのシステムと同じであれば、データファイルを DELETE して、システムから変換を試みることができます。

MEAS 3117

```
Unable to translate an ANSI SQL name.
```

メッセージの説明

原因

要求された変換が、ANSI SQL 名前空間または現在の ANSI SQL ジャーナルセグメントで見つかりませんでした。

影響

コマンドは実行されません。

リカバリ

要求が ANSI SQL ジャーナルセグメントを含む ADD データファイルの一部である場合は、現在のシステムがデータファイルのシステムと同じであれば、データファイルを削除して、システムから変換を試みることができます。LISTACTIVE DISKFILE コマンドの場合は、パーティションが指定されていることを確認してください。それ以外の場合、リカバリはできません。

MEAS 3118

```
Syntax errors in specified ANSI SQL name.
```

メッセージの説明

原因

ANSI SQL 名に構文エラーが含まれていました。

影響

要求した操作は中断されます。

リカバリ

正しい形式の ANSI SQL 名を使用してコマンドを再度入力してください。

MEAS 3121

```
Default Catalog or Schema missing.
```

メッセージの説明

原因

完全修飾 SQL ANSI 名を作成するために使用できる十分な情報がありません。デフォルトの SQL カタログ、デフォルトの SQL スキーマ、またはその両方が欠落しています。

影響

コマンドは実行されません。

リカバリ

ENV コマンドを発行して、デフォルトの SQL カタログと SQL スキーマの設定を確認してください。欠落しているデフォルト設定を追加するには、適切な SQLCATALOG または SQLSCHEMA コマンド、あるいはその両方を発行してください。

MEAS 3122

`The data file format does not support CLIM devices, or path selection.`

メッセージの説明

原因

H03 より前のデータファイルで CLIM デバイスまたはパス選択を指定した DISC または DEVICE エンティティに対して、LIST コマンドが発行されました。

影響

コマンドは実行されません。

リカバリ

CLIM デバイス、パス選択、lun、またはパーティション番号を指定しないよう、LIST コマンドを変更してください。

MEAS 3123

`Allocate segment error` *`nnnn`* `on work segment.`

メッセージの説明

原因

ALLOCATESEGMENT から、エラーが発生して返されました。MEASCOM プライマリ作業セグメントが最大容量に達して、拡張することができません。

影響

要求した操作は無視されます。

リカバリ

より小さいレコードのサブセットを取得するよう、LIST コマンドを変更してください。

MEAS 3124

`Partition number is not allowed for this command and entity.`

メッセージの説明

原因

ADD DEVICE、LIST DEVICE、LISTALL DEVICE、および LISTACTIVE DEVICE コマンドではパーティション番号を指定できません。

影響

コマンドは実行されません。

リカバリ

パーティション番号を指定しないようにコマンドを変更してください。

MEAS 3125

A *sac-list* specification is not allowed for this command and entity.

メッセージの説明

原因

ADD DEVICE、ADD DISC、LIST DEVICE、LISTALL DEVICE、LISTACTIVE DEVICE、および LISTACTIVE DISC コマンドでは、複数の CLIM 名は指定できません。

影響

コマンドは実行されません。

リカバリ

複数の CLIM 名を指定しないようにコマンドを変更してください。

MEAS 3126

Path is not allowed for this command and entity.

メッセージの説明

原因

ADD DEVICE、LIST DEVICE、LISTALL DEVICE、および LISTACTIVE DEVICE コマンドではパスのサフィックスを指定できません。

影響

コマンドは実行されません。

リカバリ

パスのサフィックスを指定しないようにコマンドを変更してください。

MEAS 3127

(ERR^LEGACY^NOT^SUPPORTED)

メッセージの説明

原因

LIST コマンド STYLE LEGACY は、このエンティティではサポートされていません。

影響

コマンドは実行されません。

リカバリ

STYLE LEGACY を指定しないようにコマンドを変更してください。

MEAS 3128

STYLE ZMS is required for the current report format

メッセージの説明

原因

STYLE LEGACY は現在のレポート形式では許可されません。

影響

要求した操作は中止されます。

リカバリ

STYLE ZMS を指定するか、レポート形式を変更してください。

MEAS 3129

File System error *nnnn* occurred while opening or accessing the data export file.

メッセージの説明

原因

データエクスポートファイルを開いたりアクセスしたりしているときに、ファイルシステムエラーが発生しました。

影響

データエクスポートファイルは、作成されないか、または不完全です。

リカバリ

出力ファイルがすでに存在する場合は、それを上書きできることを確認してください。存在しない場合は、ディスクボリューム上に十分な空き領域があることと、それに対する書き込みが許可されているを確認してください。

MEAS 3130

An existing file that MEASCOM tries to open for data export has incorrect type or file code.

メッセージの説明

原因

データエクスポートファイルがすでに存在していて、不正なタイプまたはファイルコードが含まれています。

影響

要求した操作は中止されます。

リカバリ

ファイルを削除、移動、または名前を変更してください。

MEAS 3175

System is not TANDEM NonStop; see your system manager.

メッセージの説明

原因

NonStop サーバー以外のシステムで MEASCOM を実行しようとしました。

影響

MEASCOM が突然停止します。

リカバリ

システムマネージャーに確認してください。

MEAS 3176

```
System option number is not T2035; see your system manager.
```

メッセージの説明

原因

T2035 以外のオプション番号で MEASCOM を実行しようとしました。

影響

MEASCOM が突然停止します。

リカバリ

システムマネージャーに確認してください。

MEAS 3177

```
Guardian version is not B30 or greater; see your system manager.
```

メッセージの説明

原因

NonStop オペレーティングシステムの PVU は B30 以降である必要があります。

影響

MEASCOM が突然停止します。

リカバリ

システムマネージャーに確認してください。

MEAS 3178

```
Internal consistency check nnn failed; report error.
```

メッセージの説明

原因

内部チェック *nnn* が失敗しました。

影響

MEASCOM が突然停止します。

リカバリ

コマンドがかなりの量のファイル名拡張を伴う場合は、MEASCOM を再度実行して、同じコマンドを複数の手順で実行してみてください。この試みが失敗したり当初の問題の解決にならなかったりする場合は、Hewlett Packard Enterprise の担当者にお問い合わせください。

エラーコード

Measure サブシステムは、エラーメッセージとエラーコードを生成します。この付録では、Measure の呼び出し可能プロシージャーによって生成されるエラーコードについて説明します。MEASCOM（Measure のコマンドインターフェイス）によって生成されるエラーメッセージの説明については、**エラーメッセージ**(597 ページ)を参照してください。

3200

(ERR^BADMEASCB)

メッセージの説明

原因

最初のプロシージャーコールで MEASCB が-1s に設定されなかったか、Measure によって設定された後にその内容が変更されました。

影響

要求した操作は中止されます。

リカバリ

最初の Measure の呼び出しの前に MEASCB をすべて-1s に初期化し、それ以降は書き込みを行わないでください。

3201

(ERR^MISSINGPARAM)

メッセージの説明

原因

必須パラメーターが欠落していました。

影響

要求した操作は中止されます。

リカバリ

欠落しているパラメーターをプロシージャーコールの文に追加してください。

3202

(ERR^BOUNDS)

メッセージの説明

原因

無効なパラメーターアドレスが指定されました。

影響

要求した操作は中止されます。

リカバリ

パラメーターのアドレス初期化を確認して、修正してください。

3203

(ERR^BADPARAMS)

メッセージの説明

原因

呼び出し可能プロシージャーに不正なパラメーター値が指定されました。

影響

要求した操作は中止されます。

リカバリ

パラメーターの値を確認して、修正してください。

3204

(ERR^BUFTOOSMALL)

メッセージの説明

原因

バッファーサイズが小さすぎました。

影響

要求した操作は中止されます。

リカバリ

バッファーサイズを修正してください。必要なバッファーサイズは *bytesret*（返されたレコードのバイトサイズ）パラメーターに含まれています。MEASREADACTIVE プロシージャーでこのエラーが発生して、32 KB を超えるバッファーが必要な場合は、MEAS_READACTIVE_プロシージャーを使用してください。

3205

(ERR^NOTENOUGHSTACK)

メッセージの説明

原因

バッファーを送信または受信するのに十分なデータ/スタック領域がありませんでした。

影響

要求した操作は中止されます。

リカバリ

スタックデータの一部を拡張セグメントに移動して、使用可能な作業用スタックのサイズを増やしてください。

3206

(ERR^BADREPLY)

メッセージの説明

原因

Measure のプロセスから予期しない応答を受信しました。

影響

要求した操作は中止されます。

リカバリ

このエラーを再現するために必要なファイルおよびコマンドのコピーをとって、Hewlett Packard Enterprise の担当者にお問い合わせください。

3207

(ERR^NEWDESCFIELD)

メッセージの説明

原因

使用されているダウンリビジョンの MEASFH が、記述子で使用される新しいフィールドをサポートできません。

影響

要求した操作は中止されます。

リカバリ

新しいフィールドを使用せずに、要求を再発行してください。MEASCOM では、新しいフィールドを回避してください。たとえば、LIST FILE * ($PNAME)は LIST FILE * (1,52)に変更してください（1,52 が $PNAME と仮定しています）。プログラムインターフェイスを直接使用している場合は、古いバージョンの記述子を使用するか、新しいフィールドに適切なデフォルト値（たとえば、$*）のみを配置してください。

3208

(ERR^WRONGSYSTEM)

メッセージの説明

原因

構造化レコードにインデックスを付けてデータファイルから構築する MEASFH プロセスが、データファイルと別のシステム上にあります。

影響

要求した操作は中止されます。

リカバリ

MEASFH プログラムファイルまたはデータファイルのいずれかを移動して、両方とも同じノードに存在するようにしてください。リモートデータファイルに対しては読み取りアクセスのみ許可されます。

3209

(ERR^REMOTEDFILE)

メッセージの説明

原因

データファイルが呼び出し元のプロセスと同じシステム上にありませんでした。

影響

要求した操作は中止されます。

リカバリ

リモートデータファイルに対して書き込みアクセスは許可されていないため、呼び出し元のプロセスと同じシステム上のデータファイルを使用してください。

3210

(ERR^BADDFNUM)

メッセージの説明

原因

データファイルアクセス番号（DFNUM）パラメーター値が、正常な MEASOPEN 呼び出しで返された有効な DFNUM の値と一致しませんでした。

影響

要求した操作は中止されます。

リカバリ

使用された値が MEASOPEN プロシージャーによって返された値と同じであることを確認してください。

3211

(ERR^BADMEASNUM)

メッセージの説明

原因

測定番号（MEASNUM）パラメーター値が、成功した MEASCONFIGURE 呼び出しで返された有効な MEASNUM と一致しませんでした。

影響

要求した操作は中止されます。

リカバリ

使用された値が MEASCONFIGURE プロシージャーによって返された値と同じであることを確認してください。

3212

(ERR^MEASNOTACTIVE)

メッセージの説明

原因

測定が構成されず、アクティブではありませんでした。

影響

要求した操作は中止されます。

リカバリ

測定が構成されて開始されていることと、操作中に停止しなかったことを確認してください。

3213

(ERR^NOMEASCTL)

メッセージの説明

原因

CPU は動作していましたが、CPU にカウンターを割り当てて維持する MEASCTL がありませんでした。

影響

要求した操作は中止されます。

リカバリ

MEASMONCONTROL プロシージャーを使用して、欠落している MEASCTL の CPU 番号を指定してください。

3214

(ERR^MEMLOCKFAILURE)

メッセージの説明

原因

使用できる物理メモリが不十分です。

影響

要求した操作は中止されます。

リカバリ

プロセスを他のプロセッサーに再分配するか、メモリを増設してください。

3215

(ERR^LINK)

メッセージの説明

原因

LINK への呼び出しが失敗しました。

影響

要求した操作は中止されます。

リカバリ

操作を再試行してください。

3216

(ERR^NOTAVAIL)

メッセージの説明

原因

MEASREADACTIVE プロシージャーから PROCESSH、DISCOPEN、および DISKFILE レコードを使用できません。

影響

要求した操作は中止されます。

リカバリ

MEASREAD を呼び出して、これらのレコードを取得してください。

3217

(ERR^NOTSUPER)

メッセージの説明

原因

試みられた呼び出しには、スーパーグループユーザー ID（255,*n*）が必要です。

影響

要求した操作は中止されます。

リカバリ

スーパーグループユーザーに、このプロシージャーを呼び出すプロセスを実行してもらってください。

3218

(ERR^MONALREADYALIVE)

メッセージの説明

原因

Measure サブシステムはすでに起動されているため、もう一度起動することはできません。

影響

要求した操作は中止されます。

リカバリ

サブシステム起動要求で MEASMONCONTROL を呼び出す前に、Measure サブシステムが実行されていないことを確認してください。

3219

(ERR^NOMEASMON)

メッセージの説明

原因

Measure サブシステムが停止されています。

影響

要求した操作は中止されます。

リカバリ

このプロシージャーを呼び出す前に、Measure サブシステムが実行されていることを確認してください。

3220

(ERR^UDCNOTPRESENT)

メッセージの説明

原因

ユーザー定義カウンターがインストールされていないため、変更できません。

影響

要求した操作は中止されます。

リカバリ

この呼び出しの前に、変更するユーザー定義カウンターが含まれる測定を構成してください。

3221

(ERR^UDCBADBUMP)

メッセージの説明

原因

ユーザー定義カウンターに無効な BUMPTYPE が使用されました。

影響

要求した操作は中止されます。

リカバリ

BUMPTYPE パラメーターの値をチェックして、修正してください。

3222

(ERR^UDCBADINDEX)

メッセージの説明

原因

ユーザー定義カウンターに対して無効な INDEX が使用されました。

影響

要求した操作は中止されます。

リカバリ

INDEX パラメーターの値をチェックして、修正してください。

3223

(ERR^INVALIDENTITY)

メッセージの説明

原因

新規または無効なエンティティタイプが指定されました。

影響

要求した操作は中止されます。

リカバリ

エンティティタイプが有効であることと、処理中のデータファイルの RVU でサポートされていることを確認してください。

3224

(ERR^INCONSISTENCY)

メッセージの説明

原因

MEASCTL 内部整合性チェックが失敗しました。

影響

SQL アクティビティは測定されません。

リカバリ

Hewlett Packard Enterprise の担当者にお問い合わせください。

3225

(ERR^INCOMPLETEMSG)

メッセージの説明

原因

呼び出し可能プロシージャーが MEASCTL プロセスへのメッセージを完了できませんでした。

影響

要求した操作は無視されました。

リカバリ

もう一度操作を試みます。操作が引き続き失敗する場合は、Hewlett Packard Enterprise の担当者にお問い合わせください。

3226

```
(ERR^LOCALSYSNAME)
```

メッセージの説明

原因

呼び出し可能プロシージャーが Nodenumber_to_Nodename_から呼び出し元のローカルシステム名を取得できません。

影響

要求した操作は無視されました。

リカバリ

もう一度操作を試みます。操作が引き続き失敗する場合は、Hewlett Packard Enterprise の担当者にお問い合わせください。

3227

```
(ERR^MEASFH^INVALID^NAME)
```

メッセージの説明

原因

MEASFH ファイル名の処理中に Measure で内部エラーが発生しました。

影響

要求した操作は中断されました。

リカバリ

MEASFH オブジェクトファイルが存在していて、適切に保護されていることを確認してください。

3228

```
(ERR^MEASFH^CREATION)
```

メッセージの説明

原因

Process_Create_からエラーが返されました。

影響

要求した操作は中断されました。

リカバリ

MEASFH オブジェクトファイルが存在していて、適切に保護されていることを確認してください。

3229

```
(ERR^MEASFH^OPEN)
```

メッセージの説明

原因

File_Open_からエラーが返されました。

影響

MEASFH プロセスを開くことができません。要求した操作は中断されました。

リカバリ

MEASFH オブジェクトファイルが存在していて、適切に保護されていることを確認してください。

3230

```
(ERR^MEASFH^SWAPFILE)
```

メッセージの説明

原因

指定された SWAPVOL 値がエラーになりました。

影響

MEASFH プロセスまたはスワップファイルは作成されません。要求した操作は中断されます。

リカバリ

別の SWAPVOL 値を使用するか、SWAPVOL をクリアしてデフォルト値を使用してください。

3231

```
(ERR^CONFIG^NOT^AVAIL)
```

メッセージの説明

原因

MEASLISTCONFIG プロシージャーにサポートされていないエンティティタイプを指定しました。このプロシージャーは、DEVICE、DISC、FABRIC、LINE、NETLINE、OPDISK、および ENDPOINT（SERVERNET）エンティティのみをサポートしています。

影響

要求した操作は中止されます。

リカバリ

サポートされているエンティティタイプを指定して、要求を再度発行してください。

3232

```
(ERR^NOTENOUGH^PFSBUFFER) (ERR^NOTENOUGH^FPMEMORY)
```

メッセージの説明

原因

API がプロセスの柔軟なメモリプールから十分なグローバルバッファー領域を割り当てることができませんでした。

影響

操作は実行されません。

リカバリ

アプリケーションが柔軟なプールのメモリを消費しないことを確認してください。

3233

(ERR^INVALIDOSSPATH)

メッセージの説明

原因

指定された Guardian ファイル名または OSS ファイルパス名が、OSS ファイルシステムのネームサーバーに認識されなくなったか、OSS ジャーナルセグメント内のデータと一致しません。

影響

コマンドは実行されません。

リカバリ

エンティティ指定で指定する OSS ファイルおよびディレクトリ名が、測定の時点で変換可能であることを確認してください。

3234

(ERR^CRVSNNOTSPECIFIED)

メッセージの説明

原因

CRVSN 修飾子を指定せずに MEASLISTPNAME を呼び出しました。

影響

OSS ファイルパス名への Guardian ファイル名の変換は試みられますが、指定されたファイル名が一時的なエンティティの場合は、報告される名前が正しくない可能性があります。

リカバリ

MEASLISTPNAME 呼び出しに CRVSN 値を含めてください。

3235

(ERR^BUILDINGOSSJOURNAL)

メッセージの説明

原因

測定のシャットダウン時にまだ OSS ジャーナルセグメントを構築している最中だったデータファイルに対して、MEASOPEN が呼び出されました。

影響

ファイルは正常に開きますが、OSS ジャーナルセグメントは接続されません。

リカバリ

通常は、処置は不要です。変換は、現在のシステム上の OSS ファイルシステムの名前空間から処理できます。後で OSS ジャーナルセグメントにアクセスする必要がある場合は、OSS ジャーナルセグメントの構築が完了した後で、プログラムを閉じてから Measure データファイルを再度開く必要があります。ジャーナルセグメント構築手順が完了した後でも、ジャーナルセグメントの構築中に開いたファイルではデータへのアクセスが遅くなることに気付くことがあります。ファイルを閉じてから再度開くと、データへのアクセス速度が改善されます。

3236

(ERR^MISSINGOSSJOURNAL)

メッセージの説明

原因

変換に OSS ジャーナルセグメントを必要とする要求が行われましたが、Measure データファイルの現在のコンテキストには OSS ジャーナルセグメントが接続されていません。

影響

要求した操作は中断されます。

リカバリ

OSS ネームサーバーからの変換が必要な場合は、*dfnum* パラメーターを省略するか、値として-1 を渡してください。ジャーナルセグメントからの変換が必要な場合は、データファイルで収集されたものであることと、MEASOPEN 要求に対してエラー 3235（ERR^BUILDINGOSSJOURNAL）が返されなかったことを確認してください。

3237

```
(ERR^OSSDIRNOTALLOWED)
```

メッセージの説明

原因

DISKFILE コマンドに OSS ディレクトリ名が含まれていました。

影響

要求した操作は中断されます。

リカバリ

DISKFILE エンティティ指定では、完全修飾 OSS ファイルパス名のみが許可されます。一時的なファイルアクティビティまたは特定のファイルセットが必要な場合は、Guardian ファイルの命名規則を使用してください。次に例を示します。$DATA.*.*、または$*.ZYQ00000.*。

3238

```
(ERR^UNKNOWNSQLNAME)
```

メッセージの説明

原因

指定された ANSI SQL 名を変換できなかったか、指定された Guardian 名を ANSI SQL 名に変換できませんでした。

影響

要求した操作は中断されます。

リカバリ

この名前は、ANSI SQL 名前空間で最新ではなくなったか、SQL ジャーナル内に存在していません。

3239

```
(ERR^BADFORMATSQLNAME)
```

メッセージの説明

原因

指定された ANSI SQL 名前が、構文エラーのために解析できなかったか、完全修飾されていませんでした。

影響

要求した操作は中止されます。

リカバリ

エンティティ指定構文で、ANSI SQL 名が構文的に正しく、必要な場合は完全修飾されていることを確認してください。

3240

```
(ERR^BUILDINGSQLJOURNAL)
```

メッセージの説明

原因

測定のシャットダウン時にまだ SQL ジャーナルセグメントを構築している最中だったデータファイルに対して、MEASOPEN が呼び出されました。

影響

データファイルは正常に開きますが、SQL ジャーナルセグメントは接続されません。

リカバリ

通常は、処置は不要です。変換は SQL/MX サービスを通じて処理されます。後で SQL ジャーナルセグメントにアクセスする必要がある場合は、SQL ジャーナルセグメントの構成が完了した後、データファイルを閉じてから再度開いてください。

3241

```
(ERR^MISSING^SQLJOURNAL)
```

メッセージの説明

原因

変換に SQL ジャーナルセグメントを必要とする要求が行われましたが、Measure データファイルの現在のコンテキストには SQL ジャーナルセグメントが接続されていません。

影響

要求した操作は中止されます。

リカバリ

ANSI SQL ネームサーバーからの変換が必要な場合は、*dfnum* パラメーターを省略するか、値として-1 を渡してください。ジャーナルセグメントからの変換が必要な場合は、そのようなセグメントを要求したことと、MEASOPEN 要求がエラー 3240（ERR^BUILDINGSQLJOURNAL）を返さなかったことを確認してください。

3242

```
(ERR^NOT^IMPLEMENTED)
```

メッセージの説明

原因

このバージョンの Measure に存在しない Measure 機能に対して要求が行われました。

影響

要求した Measure 機能は実行されません。

リカバリ

Measure のバージョンを、指定した機能に必要なレベルにアップグレードしてください。

3243

ERR^NOSPACEOSSJOURNAL

メッセージの説明

原因

ディスク容量が十分ではなかったか、OSS または ANSI SQL ジャーナルセグメントを構築するためのセグメント割り当てが失敗しました。

影響

ファイルは正常に開きますが、ジャーナルセグメントは接続されません。

リカバリ

OSS 名の変換要求は、現在のシステム上の OSS ネームサーバーから処理されます。ANSI SQL 名の変換は、現在のシステム上の SQL/MX 経由で処理されます。OSS または SQL ジャーナルセグメントへのアクセスが必要な場合は、十分な容量のあるディスクで測定を開始してください。

3244

(ERR^NO^TIMERCELL^CONTEXT)

メッセージの説明

原因

アプリケーションが timercell 関数を呼び出しましたが、現時点で timercell コンテキストが割り当てられていません。たとえば、アプリケーションが、最初に MEAS_ALLOCATE_TIMERCELLS_を呼び出さずに、別の timercell 関数を呼び出しました。または、すべての timercell の割り当てが解除されているか、コンテキストが破損して、エラーコード 3245（ERR^BAD^TIMERCELL^CONTEXT）も返されました。

影響

要求した操作は実行されません。

リカバリ

他の timercell 関数を呼び出す前に、MEAS_ALLOCATE_TIMERCELLS_を呼び出してください。

3245

(ERR^BAD^TIMERCELL^CONTEXT)

メッセージの説明

原因

アプリケーションが timercell 関数を呼び出しましたが、timercell コンテキストが壊れています。

影響

コンテキストはクリアされ、このプロセスに対して割り当てられたすべてのタイマーセルは失われます。

リカバリ

他の timercell 関数を呼び出す前に、MEAS_ALLOCATE_TIMERCELLS_を呼び出して新しい timercell コンテキストを作成してください。

3246

(ERR^NOT^ALLOCATED)

メッセージの説明

原因

呼び出し可能な関数のターゲットとして指定されたリソースが割り当てられていません。

影響

操作は実行されません。

リカバリ

リソースを必要とするプロシージャーを呼び出す前に、リソースを割り当てるプロシージャーを呼び出してください。

3247

(WARN^TIMERCELL^ACCURACY)

メッセージの説明

原因

アプリケーションが 128 個を超えるタイマーセルを取得しようとし、十分なシステムリソースがないために、Measure が一部のカウンターの報告された値を、繰り返されるランデブー操作の時間の進行を反映するように調整できませんでした。

影響

すべてのタイマーに対するデータが返されますが、最後から 2 番目のグループ（剰余 128）のタイマーがその true 値よりわずかに小さい値を返しました。

リカバリ

一度に取得するタイマーセルは 128 未満にしてください。

3248

(ERR^NO^MORE^TIMERCELLS)

メッセージの説明

原因

アプリケーションがタイマーセルを割り当てようとしましたが、この要求を満たすにはプロセッサーの timercell プールが不十分でした。

影響

操作は発生しません。

リカバリ

すでに割り当てられているタイマーセルを使用してください。または、いくつかの既存のタイマーセルの割り当てを解除して、新しいものを割り当ててください。

3249

(ERR^NOT^SUPPORTED)

メッセージの説明

原因

サポートされなくなったリソースまたは機能に対する要求が行われました。

影響

要求した操作はエラーになります。

リカバリ

このエラーからの唯一のリカバリは、その機能を使用しないようにアプリケーションを変更することです。一般に、今後はもうその機能を使用する理由がなく、他のより良い選択肢がある場合、機能に対するサポートは除外されます。詳細については、このマニュアルを参照してください。

3250

(ERR^NO^LEGACY^STYLE)

メッセージの説明

原因

ZMS 形式のレコードのみをサポートしているエンティティに対して、Legacy 形式のレコードに対する要求が行われました。

影響

要求した操作はエラーになります。

リカバリ

Legacy 形式のレコードを指定しないように、API 呼び出しを更新してください。テンプレートバージョンのパラメーターは 0 以外の値で指定する必要があります。

3251

(ERR^BADDESC)

メッセージの説明

原因

測定対象のエンティティ記述子が含まれている CONTAB 形式が正しくありませんでした。

影響

要求した操作は中止されます。

リカバリ

不正または欠落している CONTAB のヘッダー、エンティティ記述子、またはトレーラーレコードを修正してください。呼び出しに対する CONTAB の内容を調べてください。

3252

(ERR^FILECODE)

メッセージの説明

原因

データファイルにファイルコード 175 がありませんでした。

影響

要求した操作は中止されます。

リカバリ

ファイルをパージして、ファイルコード 175 を含む新しいファイルを作成してください。

3253

(ERR^NOTDISCFILE)

メッセージの説明

原因

データファイルがディスクファイルではありませんでした。

影響

要求した操作は中止されます。

リカバリ

読み取りアクセスの前に、測定データファイルをテープからディスクにコピーしてください。

3254

(ERR^WRONGVERSION)

メッセージの説明

原因

MEASFH のバージョンがデータファイルのバージョンと一致しませんでした。

影響

要求した操作は中止されます。

リカバリ

データファイルを生成するために使用した RVU からの MEASFH プログラムファイル名を指定してください。

3255

(ERR^FILEINUSE)

メッセージの説明

原因

すでに開いている測定データファイルで測定を開始しようとしました。

影響

要求した操作は中止されます。

リカバリ

ファイルを閉じて、操作を再度実行してください。

3256

(ERR^ISEGOVERFLOW)

メッセージの説明

原因

データファイルのインデックスセグメントをソートしようとしているときに、MEASFH プロセスがエラーを受信しました。

影響

要求した操作は中止されます。

リカバリ

ファイルを閉じて、操作を再度実行してください。

3257

(ERR^RSEGOVERFLOW)

メッセージの説明

原因

データファイルのレコードアドレス配列セグメントがオーバーフローしました。

影響

要求した操作は中止されます。

リカバリ

アスタリスク（*）を使用してすべての項目をリストするのではなく、特定の項目を要求することによって、LIST 対象の項目の数を減らしてみてください。PROCESSH エンティティの場合は、LIST コマンドに `if code-range > 0` を指定してみてください。

3258

`(ERR^CSEGOVERFLOW)`

メッセージの説明

原因

MEASFH の外部カウンターレコードセグメントがオーバーフローしました。

影響

要求した操作は中止されます。

リカバリ

アスタリスク（*）を使用してすべての項目をリストするのではなく、特定の項目を要求することによって、LIST 対象の項目の数を減らしてみてください。PROCESSH エンティティの場合は、LIST コマンドに `if code-range > 0` を指定してみてください。

3259

`(ERR^MEASNOTDONE)`

メッセージの説明

原因

測定が完了しませんでした。

影響

要求した操作は中止されます。

リカバリ

測定を停止して、操作を再度実行してください。

3260

`(ERR^NOOPEN)`

メッセージの説明

原因

データファイルが開かれていませんでした。

影響

要求した操作は中止されます。

リカバリ

データファイルに対して MEASOPEN を呼び出して、失敗した操作を再度実行してください。

3261

(ERR^INTTOOSMALL)

メッセージの説明

原因

時間間隔が小さすぎました。

影響

要求した操作は中止されます。

リカバリ

1 秒以上の時間間隔で、操作を再度実行してください。

3262

(ERR^NOCTRREC)

メッセージの説明

原因

要求されたエンティティのカウンターレコードが見つかりませんでした。

影響

要求した操作は中止されます。

リカバリ

測定対象のエンティティ記述子が含まれている測定の CONTAB パラメーターを調べてください。現在の記述子が含まれているエンティティ記述子をチェックし、測定中にこのエンティティが動作していたことを確認してください。

3263

(ERR^CRFILETYPE)

メッセージの説明

原因

PROCESSH の code-range ファイルがディスクファイルではありませんでした。

影響

要求した操作は中止されます。

リカバリ

code-range ファイルをディスクに移動してください。

3264

(ERR^CRFILECODE)

メッセージの説明

原因

PROCESSH の code-range ファイルが、オブジェクトまたは EDIT ファイルではありませんでした。

影響

要求した操作は中止されます。

リカバリ

コード範囲を EDIT 形式のファイルに移動するか、オブジェクト形式のファイルを使用してコード範囲を指定してください。

3265

(ERR^CRFILE)

メッセージの説明

原因

PROCESSH の code-range ファイルでオープンエラーまたは読み取りエラーが発生しました。

影響

要求した操作は中止されます。

リカバリ

ファイル名をチェックし、ファイルが読み取り可能（FUP COPY）であることを確認してください。必要な変更を行い、操作を再度実行してください。

3266

(ERR^CRFMT)

メッセージの説明

原因

PROCESSH の code-range ファイルの形式にエラーがありました。

影響

要求した操作は中止されます。

リカバリ

形式を修正して、操作を再度実行してください。

3267

(ERR^NOCRS)

メッセージの説明

原因

PROCESSH の code-range ファイルにコード範囲が指定されていないか、指定されたコード領域に既存のコード範囲が存在していません。

影響

要求した操作は中止されます。

リカバリ

ファイルに少なくとも 1 つのコード範囲を追加するか、コード領域指定を修正して、操作を再度実行してください。

3268

(ERR^TOOMANYCRS)

メッセージの説明

原因

PROCESSH の code-range ファイルが 500 を超えるコード範囲を指定しました。

影響

要求した操作は中止されます。

リカバリ

コード範囲が 500 以下になるようにファイルから削除して、操作を再度実行してください。

3269

(ERR^CRTABLES)

メッセージの説明

原因

PROCESSH の code-range テーブルがオーバーフローしました。データファイルが壊れていたか、MEASFH に問題があります。

影響

要求した操作は中止されます。

リカバリ

測定が行われた後でデータファイルが変更された場合は、測定を繰り返して、操作を再度実行してください。それ以外の場合は、データファイルを保存して、Hewlett Packard Enterprise の担当者にお問い合わせください。

3270

(ERR^EMPTYDATAFILE)

メッセージの説明

原因

データが含まれていないデータファイルに対して読み取りアクセスが要求されました。

影響

要求した操作は中止されます。

リカバリ

測定データが含まれている別のデータファイルを使用するか、このファイルを使用して測定を開始してください。

3271

(ERR^CONTABFULL)

メッセージの説明

原因

測定用に構成されている PROCESSH のコード範囲が多すぎて、CONTAB パラメーターがいっぱいになっています。

影響

要求した操作は中止されます。

リカバリ

コード範囲の数を減らして（または、いくつかのコード領域を取り除いて）、PROCESSH の測定を再構成してください。

3272

```
(ERR^INVALID^TAPE^OP)
```

メッセージの説明

原因

テープデータファイルに対して無効な要求が行われました。

影響

要求した操作は中止されます。

リカバリ

この要求はディスクファイルに対してのみ行うことができます。ディスクファイルを指定して、操作を再度実行してください。

3273

```
(ERR^TRAP^OCCURRED)
```

メッセージの説明

原因

MEASFH プロセスがトラップされました。

影響

MEASFH プロセスが突然停止して、saveabend ファイルが作成されます。

リカバリ

トラップ番号が 12（%14）の場合は、MEASFH 作業ファイルが作成されたディスクを確認してください。ディスクがいっぱいか非常に断片化している場合は、MEASCOM SWAPVOL コマンドを使用するか、TACL RUN コマンドで SWAP パラメーターを使用して、MEASFH 作業ファイルが別のディスク上に作成されるように要求してください。エクステントを作成できないことによってトラップが発生しない場合は、CPU がメモリ不足です。その場合は、より多くのメモリが搭載された CPU で MEASFH プロセスを実行してみてください。トラップ 12 以外のトラップの場合は、表示される情報をコピーし、サービスプロバイダーに問題を報告してください。

3274

```
(ERR^UNKNOWNMSG)
```

メッセージの説明

原因

MEASFH バージョンに互換性がありません。MEASFH プロセスが、理解できない要求を受信しました。要求が現在の MEASFH PVU に実装されていないか、要求が無効なタイプです。

影響

MEASFH プロセスは要求を無視します。

リカバリ

要求が有効な場合は、新しい要求をサポートしている MEASFH ファイルを使用してください。それ以外の場合は、有効な要求を指定してください。

3275

(ERR^WRONGRELEASE)

メッセージの説明

原因

互換性のない MEASFH バージョンが指定されました。

影響

要求した操作は中止されます。

リカバリ

正しい MEASFH バージョンを指定して、コマンドを再試行してください。

3276

(ERR^MEAS^NEVER^STARTED)

メッセージの説明

原因

このデータファイルで測定が正しく開始されなかった（MEASMON によってエラー 3410（ERRM^INSTALL）が返された）か、START 時刻がデータファイルに書き込まれませんでした。

影響

データファイルに有効なデータが含められません。データファイルに対するそれ以上の操作はエラーになる可能性があります。

リカバリ

別の測定を開始してみてください。エラー 3410（ERRM^INSTALL）に対する EMS ログをチェックしてください。START 時刻をデータファイルに書き込むのに十分な大きさのデータファイルを使用してください。

3277

(ERR^DATAFILE^CORRUPT)

メッセージの説明

原因

データファイルを追加しているときに MEASFH が無効なデータレコードを検出しました。

影響

データファイルは無効なレコードまでしか分析されません。

リカバリ

対処方法はありません。MEASFH は破損したレコード以降のデータファイルを分析できません。

3278

(ERR^CONTAB^NOT^COMPLETED)

メッセージの説明

原因

MEASREADCONF 呼び出しが MEASFH より前の PVU です。

影響

MEASREADCONF は、MEASFH からの不完全な CONTAB パラメーターを処理できない可能性があります。これは大規模な測定の構成を読み取ることができません。

リカバリ

C20 以降の PVU の MEASREADCONF 呼び出しを使用してください。

3279

```
(ERR^CONVERTING^TO^RISC)
```

メッセージの説明

原因

SC または SL の EDIT ファイルを処理しているときに、MEASFH が TNS アドレスを TNS/R アドレスに変換できませんでした。

影響

測定は構成されません。

リカバリ

EDIT ファイル内の各 TNS アドレスが、構成中のコード領域に対して有効であることを確認してください。そうするには、SYSTEM.SYS*nn* の CONFLIST をチェックしてください。Measure は 0 から%177777 までの任意の番号の TNS アドレスを受け入れますが、コード領域が高速化された場合は、NonStop オペレーティングシステムによって TNS/R アドレスにマッピングされたアドレスだけが EDIT ファイルに受け入れられます。

3280

```
(ERR^FH^OLD^STRUCT^FILE)
```

メッセージの説明

原因

レコードサイズがデータファイル内のサイズと異なっています。

影響

レコードは構造化ファイルに書き込まれませんでした。

リカバリ

既存の構造化ファイルを削除して MEASFH に作成させるか、別のサブボリュームを使用してください。MEASCOM を使用する場合は、VOLUME コマンドを使用してください。必ず適切な DDL ディクショナリを再生成してください。

3281

```
(ERR^FH^CTR^NAME^NOT^FOUND)
```

メッセージの説明

原因

IF^ITEM 構造体で渡されたカウンター名が見つかりませんでした。名前のスペルが間違っているか、処理中の PVU に存在しません。

影響

操作は実行されませんでした。

リカバリ

有効なカウンター名を渡してください。

3282

(ERR^FH^BY^CTR^ONLY)

メッセージの説明

原因

IF^ITEM 構造体のカウンター名は有効ですが、IF 項目に使用できません。

影響

操作は実行されませんでした。

リカバリ

別のカウンターの名前を渡してください。

3283

(ERR^RESIZESEGMENT)

メッセージの説明

原因

MEASFH プロセスが、要求された操作の実行に必要なサイズの拡張セグメントを取得できませんでした。

影響

要求した操作は中止されます。

リカバリ

より多くの利用可能なメモリがある CPU を指定して、コマンドを再試行してください。

3284

(ERR^ALLOCATE^SEG)

メッセージの説明

原因

MEASFH プロセスが、要求された操作の実行に必要なサイズの拡張セグメントを取得できませんでした。

影響

要求した操作は中止されます。

リカバリ

より多くの利用可能なメモリがある CPU を指定して、コマンドを再試行してください。

3285

(ERR^NOT^TNDM)

メッセージの説明

原因

MEASFH プロセスがそれ以上 COFF ファイルを読み取れませんでした。

影響

操作は実行されませんでした。

リカバリ

適切な COFF ファイルを指定して、MEASCOM ADD PROCESSH コマンドを再試行してください。

3286

(ERR^EDIT^NOT^ASCEND)

メッセージの説明

原因

EDIT ファイル内のネイティブアドレスが昇順ではなかったため、MEASFH プロセスが要求された操作を実行できませんでした。

影響

要求した操作は中止されます。

リカバリ

EDIT ファイルに変更を加えて、コマンドを再試行してください。

3287

(ERR^NO^POOLSPACE)

メッセージの説明

原因

MEASFH が要求された操作を実行できませんでした。MEASMON プロセスは、この測定の開始に必要なメモリ領域を取得できません。

影響

操作は実行されませんでした。

リカバリ

測定を複数の測定に分割して、コマンドを再試行してください。

3288

(ERR^NO^GINFO)

メッセージの説明

原因

MEASFH プロセスがそれ以上 COFF ファイルを読み取れませんでした。

影響

操作は実行されませんでした。

リカバリ

COFF ファイルが部分的にリンクされています。ネイティブモードリンカー（NLD）を使用して-r オプションを指定せずに渡すことで COFF ファイルを完全にリンクしてから、リンカープロセスの出力を使って操作を再度実行してください。

3289

(ERR^DATAFILE^INCOMPLETE)

メッセージの説明

原因

1 つまたは複数の CPU で、CID、カウンター領域、タイマーセル、またはすべてが不足していました。その結果、データファイルが不完全です。

影響

データファイルに測定用の完全な情報が含まれません。

リカバリ

指定するエンティティの数を減らして、測定を再試行してください。詳しくは、EMS ログを参照してください。

3290

(ERR^VERSION^NOT^SUPPORTED)

メッセージの説明

原因

Measure PVU がサポートされていません。

影響

データファイルは読み取られません。

リカバリ

有効な Measure PVU を指定して、操作を再度実行してください。

3291

(ERR^NO^EGINFO)

メッセージの説明

原因

MEASFH がそれ以上 ELF ファイルを読み取れませんでした。

影響

操作は実行されませんでした。

リカバリ

ELF ファイルが部分的にリンクされています。ネイティブモードリンカー（NLD）を使用して-r オプションを指定せずに ELF ファイルを渡してから、リンカープロセスの出力を使って操作を再度実行してください。

3292

(ERR^CANNOTMAKEJOURNAL)

メッセージの説明

原因

測定のシャットダウン時にまだジャーナルセグメントを構築している最中だったデータファイルに対して MEASOPEN が呼び出されたか、MEASFH プロセスでジャーナルセグメントを確立するには一時ファイル領域が不十分な状態で MEASCONFIGURE が呼び出されました。

影響

データファイルが開かれるか、測定が開始されますが、OSS または ANSI SQL ジャーナルセグメントまたは名前変換なしです。

リカバリ

データファイルを追加するときは、MEASCLOSE および MEASOPEN 呼び出しを使用してデータファイルを閉じてから再度開き、構築が完了した後にジャーナルセグメントを接続してください。測定を開始す

るときは、一時ファイル作成のための適切なセキュリティとファイルに適した容量（最大 64MB）を確認してください。

3293

(ERR^NOT^REPORTFORMATNOTSTRUCTURED)

メッセージの説明

原因

LIST OSSNAMES または LIST EXTNAMES が REPORT FORMAT STRUCTURED を設定せずに発行されました。

影響

OSSNAMES または EXTNAMES ファイルにレコードが書き込まれません。

リカバリ

LIST OSSNAME または LIST EXTNAMES コマンドを発行する前に、REPORT FORMAT STRUCTURED を指定してください。

3294

(ERR^CONFIGFILECONFLICT)

メッセージの説明

原因

測定の構成で、同じオブジェクトの同じコード領域範囲を構成するために 2 つのファイルが指定されました。無効な指定のため、MEASFH は測定の構成を完了できませんでした。

影響

操作は実行されませんでした。

リカバリ

2 つのファイルのいずれかを指定から削除してください。

3295

(ERR^SQL^API^INTERNAL)

メッセージの説明

原因

バッファー領域の不足のため、MEASURE API 内部チェックが失敗しました。

影響

ANSI SQL 名の操作は実行できませんでした。

リカバリ

Hewlett Packard Enterprise の担当者にお問い合わせください。

3296

(ERR^SQLMX^MAP^PROCESS)

メッセージの説明

原因

マッピングプロセスが開始されていなかったため、SQL/MX マッピング API への呼び出しが失敗しました。

影響

ANSI SQL 名の操作は実行できませんでした。

リカバリ

SQL/MX と TMF がインストールされていて、アクティブであることを確認してください。その場合は、Hewlett Packard Enterprise の担当者にお問い合わせください。

3297

```
(ERR^SQLMX^MAP^DUPCONNECTION)
```

メッセージの説明

原因

このアプリケーションプロセスに対する既存の SQL/MX マッピングセッションがすでにあります。

影響

新しいマッピングセッションは開始されません。

リカバリ

単に MEAS_SQL_MAP_STOP を呼び出した場合は、待機してから、もう一度やり直してください。それ以外の場合、メッセージはおそらく無視することができ、すでに存在している *connectionInfo* を使用して適切な API を呼び出すことができます。

3298

```
(ERR^SQLMX^MAP^INIT)
```

メッセージの説明

原因

SQL/MX マッピングセッションを開始できませんでした。

影響

新しいマッピングセッションは開始されません。

リカバリ

SQL/MX と TMF がインストールされていて、アクティブであることを確認してください。その場合は、Hewlett Packard Enterprise の担当者にお問い合わせください。

3299

```
(ERR^LCT^CHANGE)
```

メッセージの説明

原因

測定中にシステム時刻が変更されました。通常このエラーは、夏時間の移行時に発生します。

影響

後処理の結果が正しくない可能性があります。

リカバリ

リカバリはできません。測定を再度実行してください。

3300

(ERRM^INVALIDMSG)

メッセージの説明

原因

要求のタイプまたは長さが無効でした。

影響

要求した操作は中止されます。

リカバリ

タイプまたは長さを修正して、要求を再試行してください。

3301

(ERRM^BADMEASNUM)

メッセージの説明

原因

要求で指定された測定番号が、要求されたサービスに対して無効でした。

影響

要求した操作は中止されます。

リカバリ

正しい測定要求番号を指定して、操作を再度実行してください。

3302

(ERRM^SECURITYVIOLATION)

メッセージの説明

原因

リクエスターに要求された操作を実行するための適切なセキュリティ権限がありませんでした。

影響

要求した操作は中止されます。

リカバリ

要求を確認して、操作を再度実行してください。

3303

(ERRM^OPENOVERFLOW)

メッセージの説明

原因

MEASMON を開くプロセスの数が上限を超えています。

影響

要求した操作は中止されます。

リカバリ

可能性のあるオープナーのプロセスが終了するのを待ってから、操作を再度実行してください。これらのオープナーは、MEASCOM などの Measure アプリケーションである可能性があります。

3304

(ERRM^INTERNAL)

メッセージの説明

原因

内部エラーが発生しました。

影響

要求した操作は中止されます。

リカバリ

Hewlett Packard Enterprise の担当者にお問い合わせください。

3305

(ERRM^DATAFILEINUSE)

メッセージの説明

原因

別の測定によってすでに使用されていたデータファイルを使用しようとしました。

影響

要求した操作は中止されます。

リカバリ

別のデータファイルを指定してください。

3306

(ERRM^MEASTABLEFULL)

メッセージの説明

原因

64 を超える測定を構成しようとしました。

影響

要求した操作は中止されます。

リカバリ

いくつかの測定が停止するのを待ってから、操作を再度実行してください。

3307

(ERRM^NOCONTABSPAC)

メッセージの説明

原因

エンティティ記述子が含まれている MEASMON CONTAB プールがいっぱいでした。

影響

要求した操作は中止されます。

リカバリ

いくつかの測定が停止するのを待ってから、測定を再試行してください。

3308

(ERRM^INVALIDINTERVALTIME)

メッセージの説明

原因

時間間隔が無効です。1 秒未満または 1 年より長くすることはできません。

影響

要求した操作は中止されます。

リカバリ

間隔の値を修正して、要求を再試行してください。

3309

(ERRM^INVALIDSTARTSTOPTIME)

メッセージの説明

原因

測定の制御要求で、停止時間が開始時間よりも小さかったか、アクティブな測定の開始時間を変更しようとしたか、その他のいくつかの無効な開始または停止時間の要求がありました。

影響

要求した操作は中止されます。

リカバリ

開始および停止時間の値を修正して、要求を再試行してください。

3310

(ERRM^INVALIDDATAFILE)

メッセージの説明

原因

間違ったデータファイルタイプが指定されました。

影響

要求した操作は中止されます。

リカバリ

ファイルコードが 175 の非構造化ディスクファイルを指定してください。

3311

(ERRM^INVALIDCONTAB)

メッセージの説明

原因

CONTAB 形式が正しくありませんでした。

影響

要求した操作は中止されます。

リカバリ

CONTAB 形式を修正して、操作を再度実行してください。

3312

(ERRM^DATAFILEOPENFAILED)

メッセージの説明

原因

MEASMON でデータファイルを開こうとして失敗しました。

影響

要求した操作は中止されます。

リカバリ

データファイルの現在の状態をチェックして、適切な処置を取ってください。

3313

(ERRM^ADOPTFAILED)

メッセージの説明

原因

MEASMON で既存の MEASCTL または MEASIP プロセスを使用しようとして失敗しました。

影響

要求した操作は中止されます。

リカバリ

システムオペレーターにお問い合わせください。

3314

(ERRM^ALREADYOPEN)

メッセージの説明

原因

呼び出し側のプロセスがすでに MEASMON を開いていました。

影響

要求した操作は中止されます。

リカバリ

呼び出し側が MEASMON プロセスを開くことができるのは、1 回だけです。これはすでに開いています。呼び出し可能プロシージャーに渡された MEASCB 構造体を確認してください。呼び出し側で変更されないようにしてください。

3315

(ERRM^START^MEASSUBSYS)

メッセージの説明

原因

MEASMON のバックアップが開始しませんでした。1 つまたは複数の実行中の CPU に、アクティブな MEASCTL または MEASIP プロセス、またはその両方が含まれていません。

影響

MEASURE サブシステムが完全には動作しません。

リカバリ

コンソールメッセージによって示される問題を修正してください。その後、MEASURE サブシステムを停止し、サブシステムを再起動する前に MEASMON、MEASCTL、MEASIP のすべてのプロセスが停止していることを確認してから、再起動してください。

3316

```
(ERRM^MEMLOCKFAILURE)
```

メッセージの説明

原因

LOCKMEMORY プロシージャーへの呼び出しが失敗しました。MEASMON がデータファイルにアクセスするために必要なセキュリティチェックを実行できませんでした。

影響

測定は開始されませんでした。

リカバリ

要求を再度実行してください。問題が解決しない場合、システムオペレーターにお問い合わせください。

3317

```
(ERRM^INVALIDBINADDR)
```

メッセージの説明

原因

PROCESSH エンティティに無効なコード範囲が指定されました。

影響

測定は実行されません。

リカバリ

有効なコード範囲アドレスを指定して、再試行してください。

3318

```
(ERRM^INVALIDBINPART)
```

メッセージの説明

原因

PROCESSH 構成が MEASMON に送信されませんでした。MEASMON は測定を構成できませんでした。

影響

測定は開始されませんでした。

リカバリ

要求を再度実行してください。問題が解決されない場合は、Hewlett Packard Enterprise の担当者にお問い合わせください。

3319

(ERRM^NOBINADDRSPACE)

メッセージの説明

原因

MEASCTL プロセスでコード範囲アドレス用のプール領域がいっぱいです。

影響

測定は実行されません。

リカバリ

いくつかの測定を停止してから、測定をもう一度開始してみてください。

3320

(ERRM^BACKUP^DATAFILEOPENFAILED)

メッセージの説明

原因

バックアップの MEASMON でデータファイルを開こうとして失敗しました。

影響

要求した操作は中止されます。

リカバリ

ファイルシステムエラー番号を確認して、適切な処置を取ってください。

3351

(ERR^OSSJOURNAL^INCOMPLETE)

メッセージの説明

原因

MID の拡張中に MEASFH の OSS ジャーナルセグメントがオーバーフローしました。

影響

接続されているジャーナルセグメントにはすべての名前変換は含まれません。

リカバリ

測定の構成で使用するエンティティを減らしてください。

3352

(ERR^SQLJOURNAL^INCOMPLETE)

メッセージの説明

原因

MID の拡張中に MEASFH の SQL ジャーナルセグメントがオーバーフローしました。

影響

接続されているジャーナルセグメントにはすべての名前変換は含まれません。

リカバリ

測定の構成で使用するエンティティを減らしてください。

3400

(ERRM^READLINKFAILED)

メッセージの説明

原因

プロセッサーに十分な物理メモリがなかったため、要求全体を読み取ることができませんでした。

影響

応答ファイルにデータが含まれません。

リカバリ

メモリを増設してください。

3401

(ERRM^BUFFERTOOSMALL)

メッセージの説明

原因

要求を完了するにはリクエスターのバッファーが小さすぎました。

影響

応答ファイルにデータが含まれません。

リカバリ

応答に対してより多くの領域を割り当てて、操作を再度実行してください。

3402

(ERRM^CANNOTACCESS)

メッセージの説明

原因

現在の状態が原因で、エンティティにアクセスできませんでした。デバイスに対して入力された CPU、グループ、モジュール、スロット番号、SCSI ID、SAC 名、lun、パーティション番号、パス、または CLIM 名の値が正しくない可能性があります。SCF から正しい値を見つけて、コマンドを再度実行してください。

影響

応答にデータが含まれません。

リカバリ

デバイスにアクセスできない場合は、システムオペレーターにお問い合わせください。

3403

(ERRM^ILLEGALDESC)

メッセージの説明

原因

要求のエンティティ記述子が無効でした。

影響

応答にデータが含まれません。

リカバリ

エンティティ記述子を修正して、要求を再試行してください。

3404

(ERRM^NOTMEASURING)

メッセージの説明

原因

測定番号（MEASNUM）パラメーターがエンティティを測定しなかったか、エンティティが存在しません。

影響

応答にデータが含まれません。

リカバリ

エンティティが測定の CONTAB パラメーターに存在することを確認してください。

3405

(ERRM^REPLYTOOBIG)

メッセージの説明

原因

応答が MEASCTL 応答バッファーに収まりませんでした。これは、READACTIVE 要求に対する応答で返されるユーザー定義カウンターが多すぎるときに発生する可能性があります。

影響

応答にデータが含まれません。

リカバリ

一部のユーザー定義カウンターを除外してください。

3406

(ERRM^ILLEGALREQ)

メッセージの説明

原因

要求が無効でした。

影響

応答にデータが含まれません。

リカバリ

エンティティ記述子のタイプを確認して修正し、実際の要求を確認して、操作を再度実行してください。

3407

(ERRM^ILLEGALMEASNUM)

メッセージの説明

原因

要求内の測定番号（MEASNUM）が有効な測定番号ではなかったか、この測定番号を持つアクティブな測定がありませんでした。

影響

応答にデータが含まれません。

リカバリ

正しい測定番号を指定してください。

3408

(ERRM^MEASNUMINUSE)

メッセージの説明

原因

すでにアクティブな測定を開始しようとしました。

影響

操作は中止されます。

リカバリ

Hewlett Packard Enterprise の担当者にお問い合わせください。

3409

(ERRM^NOCONTABSPACE)

メッセージの説明

原因

MEASCTL CONTAB プールがいっぱいでした。

影響

操作は中止されます。

リカバリ

その他のいくつかの測定が停止するのを待ってから、操作を再度実行してください。

3410

(ERRM^INSTALL)

メッセージの説明

原因

MEASCTL でリソースの問題が発生しました。

影響

操作は中止されます。

リカバリ

いくつかのその他の測定が停止するのを待つか、測定エンティティの数を減らしてから、操作を再度実行してください。

3411

(ERRM^DFOPENFAILED)

メッセージの説明

原因

データファイルを開くことができませんでした。

影響

操作は中止されます。

リカバリ

現在開いている状態のデータファイルとそのファイルセキュリティを確認してください。適切なデータファイルで操作を再度実行してください。

3412

(ERRM^DFFULL)

メッセージの説明

原因

使用中のデータファイルが EOF に達しました。

影響

測定は中止されます。

リカバリ

より大きいディスクを使用するか、より大きいデータファイルを使用してください。

3413

(ERRM^DFFAILED)

メッセージの説明

原因

現在のデータファイルへの書き込みが失敗しました。

影響

測定は中止されます。

リカバリ

現在の測定を再開してください。

3414

(ERRM^LOCKMSGCTRLBUF)

メッセージの説明

原因

LOCKMEMORY プロシージャーへの呼び出しが失敗しました。

影響

測定は中止されます。

リカバリ

要求を再度実行してください。問題が解決しない場合、システムオペレーターにお問い合わせください。

3415

(ERRM^SQLSPACEOVERFLOW)

メッセージの説明

原因

SQL 制御ブロック領域が使用できません。

影響

測定は中止されます。

リカバリ

SQL 測定を再開してください。

3416

(ERRM^INVALIDRECLEN)

メッセージの説明

原因

構成レコードの単語の長さが極端に長くなっています。

影響

測定は中止されます。

リカバリ

測定を再開してください。

3420

(ERRM^NOCIDENTRY)

メッセージの説明

原因

測定を開始しているときに、指定された CPU の MEASCTL プロセスが内部カウンター領域を使い果たしました。

影響

一部のエンティティは測定されません。

リカバリ

システムでのすべての測定に対して、必要なエンティティのみを構成することによって、測定対象の一時的なエンティティの数を減らしてください。たとえば、構成でワイルドカード（*）指定を使用しないようにしてください。必要のないエンティティを除外するには、DELETE コマンドを使用してください。

3421

(ERRM^NOCOUNTERSPACE)

メッセージの説明

原因

測定を開始しているときに、指定された CPU の MEASCTL プロセスが内部カウンター領域を使い果たしました。

影響

一部のエンティティは測定されません。

リカバリ

システムでのすべての測定に対して、必要なエンティティのみを構成することによって、測定対象の一時的なエンティティの数を減らしてください。たとえば、構成でワイルドカード（*）指定を使用しないようにしてください。必要のないエンティティを除外するには、DELETE コマンドを使用してください。

3430

(ERRM^ILLEGALINDEX)

メッセージの説明

原因

SQLSTMT エンティティに対する MEASREADACTIVE または MEAS_READACTIVE_呼び出しに､不正な SQL 文のインデックス番号が指定されました。

影響

操作は中止されます。

リカバリ

指定された番号が目的の SQL プロセスに対して正しい文のインデックス番号であることを確認してください。

3431

(ERR^CANNOT^ACCESS)

メッセージの説明

原因

現在の状態が原因で、エンティティにアクセスできません。

影響

データファイル内でいくつかの SQLPROC、SQLSTMT、または OSSNS レコードが欠落します。

リカバリ

指定された測定の適用範囲を減らして、操作を再度実行してください。

3432

(ERR^DESCNOTFOUND^INCONTAB)

メッセージの説明

原因

contab で記述子が見つかりませんでした。

影響

データファイル内でいくつかの SQLPROC、SQLSTMT、または OSSNS レコードが欠落します。

リカバリ

指定された測定の適用範囲を減らして、操作を再度実行してください。

3433

(ERR^MEASUREMENT^NOTACTIVE)

メッセージの説明

原因

エンティティ測定がアクティブではありません。

影響

データファイル内でいくつかの SQLPROC、SQLSTMT、または OSSNS レコードが欠落します。

リカバリ

操作を再試行してください。

3434

(ERR^GETPOOL)

メッセージの説明

原因

バッファー内の領域を取得できません。

影響

データファイル内でいくつかの SQLPROC、SQLSTMT、または OSSNS レコードが欠落します。

リカバリ

指定された測定の適用範囲を減らして、操作を再度実行してください。

3435

(ERR^SQLCB^INCONSISTENCY)

メッセージの説明

原因

測定中の SQL 制御ブロックの不整合。

影響

データファイル内でいくつかの SQLPROC または SQLSTMT レコードが欠落します。

リカバリ

Hewlett Packard Enterprise の担当者にお問い合わせください。

3436

(ERR^SQLCTR^INCONSISTENCY)

メッセージの説明

原因

測定中の SQL カウンターアドレスの不整合。

影響

データファイル内でいくつかの SQLPROC または SQLSTMT レコードが欠落します。

リカバリ

Hewlett Packard Enterprise の担当者にお問い合わせください。

3437

(ERR^SQL^INCONSISTENCY)

メッセージの説明

原因

測定中の SQL 構造の不整合。

影響

データファイル内でいくつかの SQLPROC または SQLSTMT レコードが欠落します。

リカバリ

Hewlett Packard Enterprise の担当者にお問い合わせください。

3440

(ERRM^SQLSTMT^MAPPOOL^OVERFLOW)

メッセージの説明

原因

SQLSTMT 領域がいっぱいです。

影響

データファイル内でいくつかの SQLPROC または SQLSTMT レコードが欠落するか、アクティブな要求が空の外部レコードを返します。

リカバリ

SQLSTMT 測定を停止して領域を解放し、操作を再度実行してください。

3441

(ERRM^SQLSPACE^OVERFLOW)

メッセージの説明

原因

1 つまたは複数の CPU で、SQL 制御ブロック領域、SQLSTMT マップ領域、またはその両方が不足していました。

影響

データファイル内でいくつかの SQLPROC または SQLSTMT エンティティレコードが欠落します。その結果、データファイルが不完全です。

リカバリ

指定された測定の適用範囲を減らして、操作を再度実行してください。

サブシステムファイル

次のリストには、Measure のパフォーマンスモニターを構成するファイルについて説明します。

MEASCOM プロセス	MEASCOM は Measure サブシステムコマンドインタープリターです。コマンドの説明については、**MEASCOM コマンド**(36 ページ)を参照してください。通常、MEASCOM プロセスは TACL から開始され、開始したユーザーは Measure サブシステムとの対話型セッションを持続できます。スーパーグループユーザー（ユーザーグループ 255 のメンバー）は、MEASCOM を使用して Measure サブシステムを起動できます。その後、どのユーザーも MEASCOM を使用して、測定を構成、実行、および確認できます。同時セッションのほか、セッション内の同時測定も許可されます。MEASCOM は$SYSTEM.SYS*nn* に格納されます。
MEASFH プロセス	MEASFH プロセスは、測定を実行したり、結果として得られるデータファイルを後から調べるたびに、呼び出し可能なプロシージャーで作成され、1 つの MEASFH プロセスが 1 つのユーザー指定のデータファイルに対して、必要に応じてフォーマット、データの追加、データの取得などの管理を行います。あるバージョンで生成されたデータファイルは同じバージョンの MEASFH プロセスを使用した場合にのみ確認できるという点で、MEASFH はリリースに依存します。MEASFH は$SYSTEM.SYS*nn* に格納されます。H01 PVU では、デフォルトのレコード形式はレガシーから ZMS に変更しました。したがって、G シリーズ以前の RVU では、ZMS 形式のレコードおよびレポートにのみ表示されるフィールドは、ZMS 形式のレポートモードの場合でも「IF」句では使用できません。H シリーズ以降の RVU では、Legacy 形式のレコードおよびレポートにのみ表示されるフィールドは、Legacy 形式のレポートモードの場合でも「IF」句では使用できます。
MEASMON プロセス	サブシステムの監視および調整プロセスとして機能しながら、単一の MEASMON プロセスは、Measure サブシステムが起動するときに作成されます。MEASMON は、各 CPU に MEASCTL プロセスを作成し、これらのプロセスに測定構成要求を送信して情報を収集しながら、これらのプロセスのアクティビティを調整します。MEASMON は$SYSTEM.SYS*nn* に格納されます。
MEASCTL プロセス	MEASMON は、Measure サブシステムが起動するときに、CPU ごとに 1 つの MEASCTL プロセスを作成します。MEASCTL プロセスのジョブは、カウンター領域の説明を処理し、カウンターの管理を担当する適切なサブシステムを関与させることです。MEASCTL により実行されたタスクには、測定が開始したときにカウンターを初期化するタスクや、データファイルにカウンターをコピーするタスクがあります。MEASCTL は、カウンターを表示するために、オンライン調査に対するユーザー要求も取得します。MEASCTL は$SYSTEM.SYS*nn* に格納されます。
MEASIP プロセス	MEASIP は Measure 割り込みプロセスです。PROCESSH 測定値のサンプリングを行います。MEASMON プロセスは、Measure サブシステムが起動するときにローカルシステムの各 CPU で実行する 1 つまたは複数の MEASIP プロセスを作成します。このプロセスは、H シリーズ以前の RVU には存在していません。MEASIP は $SYSTEM.SYS**nn** に格納されます。
MEASDDLS ファイル	MEASDDLS ファイルには、測定結果を表示するコマンドおよびプロシージャーによって生成されるオプションの構造化ファイルのレコード定義（DDL）が含まれています。これらのレコード定義を使用して、データをプログラムで使用するための、または Enform 製品を使用したユーザー開発のレポートの基礎として使用するための言語固有のレコード宣言ファイルを作成できます。MEASDDLS は$SYSTEM.SYS*nn* に格納されます。

表は続く

MEASDDLF	MEASDDLF ファイルには、FORTRAN DDL 定義が含まれています。MEASDDLF は $SYSTEM.SYS*nn* に格納されます。
MEASDDLB	MEASDDLB ファイルには、COBOL85 DDL 定義が含まれています。MEASDDLB は $SYSTEM.SYS*nn* に格納されます。
MEASDDLZ ファイル	MEASDDLZ ファイルにより、アプリケーションは、影響を最小限に抑えて ZMS 形式のデータレコードにアクセスできます。ZMS 形式のレコード構造体は、レコードの各コンポーネントに命名レベルを追加します。Legacy 形式には、この追加の命名レベルはありません。たとえば、Legacy 形式のレコードのフィールド cpu.dispatches は、ZMS 形式のレコード構造体の zmscpu.ctr.dispatches と同じです。既存のアプリケーションおよび ENFORM レポートをレガシー命名形式から ZMS 形式の命名形式に変換することが、レコード参照の重要な編集になります。また、プログラムで参照される可能性のあるフィールドと、2 つのサイズで表示された同等のフィールドの推測される組み合わせ（process.sent-bytes と process.sent-bytes-f など）の削除も伴います。このプロセスを簡素化するには、MEASDDLZ ファイルを使用します。 MEASDDLZ は、Legacy 形式のレコード命名規則を使用して、ZMS 形式のカウンター幅とレコード内の位置を示します。レコードテンプレート名では、Legacy 形式の名前を使用します（たとえば、ZMSPROC ではなく PROCESS）。組み合わされたフィールドは再定義により参照されます。
MEASDECS ファイル	MEASDECS ファイルには、Measure 呼び出し可能プロシージャーで使用される構造体宣言とリテラル値定義が含まれます。MEASDECS は$SYSTEM.SYS*nn* に格納されます。**C および C++言語の使用上の注意**(670 ページ)を参照してください。
MEASCHMA ファイル	MEASCHMA ファイルには、C および TAL の構造体宣言が含まれます。この出力は MEASDECS ファイルの出力に似ています。**C および C++言語の使用上の注意**(670 ページ)を参照してください。
MEASIMMU ファイル	MEASIMMU ファイルには、Measure のエラーメッセージと、MEASCOM HELP コマンドでアクセスできるオンラインヘルプテキストが含まれます。MEASIMMU は $SYSTEM.SYS*nn* に格納されます。

注記:

MEASCTL、MEASFH、MEASIP、および MEASMON は、アクティブなシステムサブボリューム（SYS*nn*）上に置かれている必要があります。残りの Measure ファイルは、アクティブなサブボリュームに置く必要はありません。

C および C++言語の使用上の注意

フィールドがユニオン内にあるかどうかを確認するには、MEASCHMA および MEASDECS ファイルの C 宣言を参照してください。フィールドがユニオン内にある場合は、フィールドを使用するコードを書くときにユニオン名を含めてください。

Measure データを使用する既存の C または C++プログラムは、Measure の MEASCHMA または MEASDECS の新しい PVU の C 構造体宣言を使用してもコンパイルしない場合があります。現在ユニオン内にあるフィールドへの参照は、参照にユニオン名が含まれるように変更する必要があります。

プロセス識別番号

D シリーズ、G シリーズ以降の RVU を実行しているシステム上のプロセス識別番号（PIN）は、次の 2 つの範囲に分けられます。

- 0~254 の low PIN 範囲
- 256 から CPU 用にサポートされる最大数までの High PIN 範囲

PIN 255 は合成プロセス ID にのみ使用されます。PIN 255 について詳しくは、*Guardian Application Conversion Guide* を参照してください。

C シリーズの RVU 上のプロセスは low PIN で実行します。High PIN で実行するプロセスは、C シリーズの RVU を実行しているシステム上に存在するプロセスおよびファイルとは通信できません。D シリーズや G シリーズ以降のプロセスで C シリーズプロセスにアクセスできるようにするには、D シリーズや G シリーズ以降のプロセスを low PIN で実行する必要があります。

表 7: Measure 製品の PIN 情報

プロセス	High PIN で実行可	デフォルトで High PIN に設定
MEASCOM	いいえ	なし
MEASMON	はい	はい
MEASCTL	はい	はい
MEASFH	はい（Measure H02 以降の PVU で）	はい（Measure H02 以降の PVU で）

D シリーズや G シリーズ以降の RVU を実行しているシステムで Measure 製品を実行する場合は、PIN 範囲を考慮する必要はありません。ただし、C シリーズの RVU を実行しているシステム上の測定データファイルにアクセスするには、low PIN で Measure 製品を実行する必要があります。

low PIN で Measure プロセスを実行するには、TACL RUN コマンドで/HIGHPIN OFF/オプションを使用するか、TACL HIGHPIN 組み込み変数を OFF に設定します。

注記:

G シリーズ以降の RVU を実行しているシステムで D シリーズ以前の RVU からのデータファイルを検査するには、適切な MEASFH とともに、D*nn* PVU の MEASCOM を使用する必要があります。さまざまなデータファイルとの互換性をできるかぎり維持するには、常に最新の D*nn* PVU の MEASCOM を使用してください。

Measure データファイルツール（MEASFT）

Measure データファイルツール（MEASFT）によって、MEASCOM または Measure API による利用および表示のために測定データファイルを複数のデータファイルに分割できます。MEASFT は、多数のエンティティの測定や非常に短い間隔での測定から発生することがある大量のデータファイルを扱う場合に特に便利です。

MEASFT は H06.07 以降の RVU の一部ですが、Measure G10 以降の G シリーズの PVU によって作成された測定データファイルを分割するために使用できます。G シリーズのデータファイルを分割するには、最初に H シリーズ以降の RVU を実行するシステムにファイルを移動する必要があります。MEASFT を実行した後、出力データファイルを元のシステムに移動して処理および表示を行うか、適切な MEASFH オブジェクトファイルバージョンを指定する ADD コマンドの MEASFH オプションを使用して､H シリーズ以降の RVU を実行中のシステム上で出力データファイルを処理します。

注記:

Measure G12 以降の G シリーズの PVU では、ファイルを分割するために、G シリーズのデータファイルを H シリーズ以降の RVU が実行中のシステムに移動する必要がなくなりました。

MEASFT には 2 つの主要な機能があり、それぞれ以下のコマンドによって呼び出されます。

- SPLIT：このコマンドは、1 つの測定データファイルを複数の測定データファイルに分割します。ファイルの分割を、エンティティタイプ別に行うか、プロセッサー番号別に行うか、時間枠に基づいて行うか（H06.25/J06.14 以降の RVU のみ）を指定できます。測定の構成、START/STOP 測定レコード、および任意の該当する OSS および SQL ジャーナルデータは各出力データファイルにレプリケートされます。
- INFO：このコマンドは測定データファイルを読み込み、各エンティティタイプおよびプロセッサー番号に関連するデータで使用されるファイル容量を示すレポートを表示します。この情報は、データファイルを分割するための最も良い方法を決定するのに役立ちます。

MEASFT には、MEASFT コマンドの構文要件を表示する HELP コマンドもあります。

INFO コマンド

INFO コマンドは、データファイル内のどの程度のスペースが、各エンティティタイプおよびプロセッサータイプに関連付けられているかを示すレポートを生成します。ファイルの最適な分割基準を決定するために、この情報を使用することができます。

H06.25/J06.14 以降の RVU では、INFO コマンドはデータファイルの分割履歴レコードも表示します。分割履歴情報には、データファイルが作成された時間、分割を実行した MEASFT プログラムの MEASFT バナー、およびデータファイルを作成したコマンドラインが含まれています。さらに該当する場合、期間と間隔値が表示されます。

- データファイルが TIME 分割ファイル（SPLIT BY TIME コマンドによって作成されたデータファイル）または TIME 分割ファイルの子孫の場合、INFO コマンドは、元のデータファイルの期間のサブセットにあたる、データファイルの期間値も表示します。
- データファイルが間隔測定用であるか、間隔測定の子孫の場合、INFO コマンドは間隔値も表示します。

L シリーズでは、INFO コマンドは FABRIC エンティティの使用をリストします（データファイル内に存在する場合）。

構成、START/STOP レコード、およびジャーナルデータは別々に一覧表示され、エンティティおよびプロセッサーの使用統計情報に含まれていません。

構文

```
INFO filename
```

filename

検査対象の Measure データファイルを識別します。完全修飾または部分修飾の Guardian ファイル名を使用します。

例

- H07 Measure を実行するシステム上で作成された Measure データファイルは、SERVERNET の代わりに ENDPOINT が表示されます。

```
$MEAS1 KMAZAWA 5> measft info mdf5ea15
MEASURE File Tool - T9086L01 - (15NOV14) - \MEASCOG
(C) Copyright 2007-2014 Hewlett-Packard Development Company, L.P.
INFO MDF5EA15
\MEASCOG - Sys id: 77153
$MEAS1.KMAZAWA.MDF5EA15 (H07) File Size 4681768 bytes

Data file usage (in bytes) by Entity

Entity                Space       %
------              ----------  ------
CPU                     59192    1.26
DISC                   126816    2.71
ENDPOINT               104952    2.24
FABRIC                 480520   10.26
PROCESS               3458360   73.87
------              ----------  ------
Total                 4229840   90.35

Data file usage (in bytes) by Processor Number

CPU                   Space       %
------              ----------  ------
0                       887592   18.96
1                       792560    16.93
2                       494632    10.57
3                       499680    10.67
4                       388944    8.31
5                       388056    8.29
6                       389280    8.31
7                       389096    8.31

Data file usage for configuration is 451928 bytes (9.65%)

Measurement period of the original data file
  Start time: February 6, 2014 19:47:30
  Stop time: February 6, 2014 19:52:23
  Interval: 15 seconds
$MEAS1 KMAZAWA 6>
```

- H06.25/J06.14 以降の RVU では、INFO コマンドの表示にはデータファイルの分割履歴レコードが含まれています。次の例はデータファイル（MDFD00）を表示した例で、MDFALLI を TIME 別に 1 時間の PARTS に分割したもの、MDFA01 を CPU 別に分割したもの、MDFB01 を ENTITY 別に分割したもの、TIME 別に分割して 11:30 から始まる 15 分間のデータを抽出したものを列挙した結果です。TIME ベースの分割の SPLIT History エントリーには、分割されたファイルの時間間隔を表示する追加の行が表示されることに注意してください。

```
$PERF.CAPAPP 156> MEASFT INFO MDFD00
MEASURE File Tool - T9086J04 - (01AUG12) - \BLPROD
```

```
(C) Copyright 2007-2012 Hewlett-Packard Development Company, L.P.
INFO MDFD00
\BLPROD  - Sys id: 000001
$PERF.CAPAPP.MDFD00 (H06) File Size 24619732 bytes

Data file usage (in bytes) by Entity

Entity                Space       %
------              ----------  ------
FILE                  24617552   99.99
------              ----------  ------
Total                 24617552   99.99

Data file usage (in bytes) by Processor Number

CPU         Space       %
---      ----------  ------
 2         24617552   99.99

Data file usage for configuration is 2180 bytes (0.01%)

Time period of this Split data file
  From: March 16, 2012 11:30:00
  To:   March 16, 2012 11:45:00

Measurement period of the original data file
  Start time: March 16, 2012 10:26:09
  Stop time:  March 16, 2012 14:32:34
  Interval: 2 minutes

SPLIT History

Number of Splits is 4

  Entry 1, Timestamp April 10, 2012 17:01:23
   Banner: MEASURE File Tool - T9086J04 - (01AUG12) - \BLPROD
   CmdLin: SPLIT MDFALLI,MDFA*,BY TIME,INTO 1 HOUR PARTS
   Time period: 16 Mar 2012, 11:26:09 to 16 Mar 2012, 12:26:09
  Entry 2, Timestamp April 10, 2012 17:04:13
   Banner: MEASURE File Tool - T9086J04 - (01AUG12) - \BLPROD
   CmdLin: SPLIT MDFA01,MDFB*,BY CPU *
  Entry 3, Timestamp April 10, 2012 17:05:22
   Banner: MEASURE File Tool - T9086J04 - (01AUG12) - \BLPROD
   CmdLin: SPLIT MDFB02,MDFC*,BY ENTITY *
  Entry 4, Timestamp April 10, 2012 17:07:22
   Banner: MEASURE File Tool - T9086J04 - (01AUG12) - \BLPROD
   CmdLin: SPLIT MDFC04,MDFD*,BY TIME,FROM 11:30,TO 11:45
   Time period: 16 Mar 2012, 11:30:00 to 16 Mar 2012, 11:45:00
```

- H06.25/J06.14 の RVU では、INFO コマンドの表示にはデータファイルの分割履歴レコードが含まれていません。

```
$PERF.CAPAPP 28> MEASFT INFO MDATA5
MEASURE File Tool - T9086H01 - (01AUG06) - \YOSPRD
Copyright 2006 Hewlett-Packard Company
INFO MDATA5
$PERF.CAPAPP.MDATA5 (H01) File Size 133980160 bytes
```

```
Data file usage (in bytes) by Entity

Entity            Space       %
------         ----------  ------
CPU               133056    0.10
DISC             2031486    1.52
DISCOPEN        25678814   19.17
DISKFILE        17085880   12.75
FILE            35348390   26.38
NETLINE           173644    0.13
OSSCPU            155626    0.12
OSSNS              23394    0.02
PROCESS         41238946   30.78
SERVERNET       10585470    7.90
SQLPROC             9956    0.01
SQLSTMT            82328    0.06
TMF                29766    0.02
------         ----------  ------
Total          132576756   98.95

Data file usage (in bytes) by Processor Number

CPU         Space       %
---      ----------  ------
 0         28409802   21.20
 1         19602142   14.63
 2         41878388   31.26
 3         42686424   31.86

Start time: March 10, 2006 08:19:05
Stop time:  March 10, 2006 08:38:53
```

SPLIT コマンド

SPLIT コマンドは、指定したデータファイルを複数のデータファイルに分割します。選択したレコードから成る単一のファイルを元のデータファイルから生成する場合にも使用できます。

ファイルの分割を、エンティティタイプ別に行うか、プロセッサー番号別に行うか、時間枠に基づいて行うか（H06.25/J06.14 以降の RVU のみ）を指定できます。

データファイルを分割した後、MEASFT は、出力データファイルの名前、サイズ、および条件を示した、結果として得られる分割の概要を表示します。

構文

```
SPLIT filename [, out-file-set [, split-criteria] ]
```

filename

分割するデータファイルを指定します。部分修飾または完全修飾の Guardian ファイル名を指定できます（つまり、ノード、ボリューム、およびサブボリュームを指定することも省略することもできます）。

out-file-set

分割から作成されるファイルの名前を指定します。以下の方法のいずれかでセットを指定します。

```
*
name-prefix*
(name1[,name2]...)
name
```

out-file-set と *split-criteria* の両方を省略すると、MEASFT は、分割をできるだけ均等にする分割条件を使用して、入力データファイルを ZMSRF000 と ZMSRF001 の名前の 2 つのファイルに分割します。

アスタリスク（*）を指定すると、MEASFT は、ZMSRF*nnn* の形式の名前（ZMSRF000 から開始）で出力データファイルを作成します。作成される出力データファイルの数は、*split-criteria* によって異なります。*split-criteria* を指定しない場合、MEASFT は、**SPLIT コマンドの使用上の注意**(679 ページ)で説明されているように動作します。

アスタリスクで終わる文字列を指定する場合、MEASFT は、その値をファイル名プレフィックスと見なし、最初のファイルには「00」から始まる番号を付加します。作成される出力データファイルの数は、*split-criteria* によって異なります。*split-criteria* を指定しない場合、MEASFT は、**SPLIT コマンドの使用上の注意**(679 ページ)で説明されているように動作します。

ファイル名のカンマ区切りのリストをかっこで囲んで指定すると、MEASFT は作成するファイルにそれらの名前を使用します。少なくとも 1 つの名前を指定する必要があります。指定した名前の数が、*split-criteria* で指定された出力データファイルの数と異なる場合、MEASFT はエラーメッセージで終了します。

名前を 1 つしか指定しなかった場合は、*split-criteria* を指定して、単一の出力データファイルに含めるものを指定する必要があります。

name-prefix、*namen*、および *name* は部分修飾名か完全修飾名です。たとえば、次の指定はすべて有効です。

```
\node.$vol.subvol.*

subvol.pref*

(split1, \node.$vol.subvol.rsplit2, othersv.split3)
```

出力ファイル名に、分割するデータファイルの名前ではなく、既存の Measure データファイルの名前を使用した場合（つまり、指定した出力ファイル名が *filename* で指定したファイル名でない場合）、MEASFT は警告メッセージを表示し、ファイルに書き込む前にデータをパージします。分割する既存の Measure データファイルの名前と同じ名前を出力ファイル名に指定した場合、MEASFT はエラーメッセージを表示して終了します。

Measure データファイルではない既存のファイルの名前を出力ファイル名に指定した場合は、MEASFT はエラーメッセージを表示して終了します。

split-criteria

データファイルの分割方法を指定します。その構文は、次のいずれかになります。

```
BY ENTITY *
BY ENTITY (entity-list1[, entity-list2]...)

BY CPU *
BY CPU (cpu-list1[, cpu-list2]...)


BY TIME, FROM time [, { TO time | FOR duration } ]
BY TIME, TO time
BY TIME, FOR duration
BY TIME, INTO number [ SECOND | MINUTE | HOUR ] [ PARTS ]
```

BY ENTITY にアスタリスク（*）を続けると、MEASFT は、データファイルに存在するエンティティタイプごとに 1 つの出力データファイルを作成します。

BY ENTITY に 1 つまたは複数のエンティティリストを続けると、MEASFT はエンティティのリストごとに 1 つの出力データファイルを作成します。*entity-listn* の構文は次のとおりです。

```
entity-name [entity-name]...
```

つまり、*entity-name* のスペース区切りのリストです。*entity-name* に指定可能な値は次のとおりです。

CLUSTER
CPU
DEVICE
DISC または DISK
DISCOPEN または DISKOPEN
DISKFILE
ENDPOINT
FABRIC
FILE
LINE
NETLINE
OPDISK
OSSCPU
OSSNS
PROCESS
PROCESSH
SERVERNET
SQLPROC
SQLSTMT
SYSTEM
TERMINAL
TMF
USERDEF

最後の *entity-listn* はアスタリスク（*）にすることもできます。これは、残りのすべてのエンティティのデータを最後の出力データファイルに格納するように MEASFT に指示します。*entity-list1* だけを指定すると、単一の出力データファイルが作成されます。

BY CPU にアスタリスク（*）を続けると、MEASFT は、データファイルに存在するプロセッサー番号ごとに 1 つの出力データファイルを作成します。

BY CPU にプロセッサー番号の 1 つまたは複数のリストを続けると、MEASFT は、プロセッサー番号のリストごとに 1 つの出力データファイルを作成します。cpu-list*n* の構文は次のとおりです。

```
cpu-num [cpu-num]...
```

つまり、*cpu-num* のスペース区切りのリストです。

最後の *cpu-listn* はアスタリスク（*）にすることもできます。これは、残りのすべてのプロセッサーデータが最後の出力データファイルに格納されることを意味します。*cpu-list1* だけを指定した場合、単一の出力データファイルが作成され、リストにプロセッサー番号のデータが含められます。

H06.25/J06.14 以降の RVU のみで、BY TIME 分割条件に基づいてファイルを分割するかどうかを指定することもできます。

BY TIME, FROM *time* と指定すると、MEASFT は、指定した *time* から入力データファイル内に存在していたエンティティを含む出力データファイルを 1 つ作成します。BY TIME, FROM *time* には、オプションで TO *time* または FOR *duration* の指定を続けることができます。TO *time* を指定した場合、

TO *time* は時間枠の終了時間を定義します。FOR *duration* を指定した場合、FOR *duration* は時間枠の持続期間を定義します。

BY TIME, TO *time* と指定すると、MEASFT は、測定開始時刻から指定の *time* までの間に入力データファイルに存在していたエンティティを含んだ出力データファイルを 1 つ作成します。

BY TIME, FOR *duration* と指定すると、MEASFT は、測定開始時刻から指定の *duration* の間に入力データファイルに存在していたエンティティを含んだ出力データファイルを 1 つ作成します

注記:

MEASFT は、G10 から H06 のバージョンの Measure データファイルをサポートします。ただし、BY TIME 分割条件は、H01 から H06 のバージョンのデータファイルでのみサポートされます。SPLIT BY TIME コマンドで必要な時間情報が H01 より前のバージョンには存在していなかったためです。

BY TIME 分割条件とともに FROM、TO、および FOR パラメーターを使用した場合、サポートされる出力ファイル（分割ファイル）は 1 つだけです。

time の構文は次のとおりです。

```
[ date , ] time-of-day
```

ここで

date

は次の形式です。

```
dd mmm yyyy
```

ここで

dd は日であり、1～31 の範囲の数字です。

mmm は月の最初の 3 文字で、たとえば JAN、MAR、OCT とします。

yyyy は 4 桁の年です。

date を省略した場合、測定が開始された日付が使用されます。

time-of-day

は次の形式です。

```
hh:mm[:ss ]
```

ここで

hh は時間です（0～23）。

mm は分です（0~59）。

ss は秒です（0~59）。

duration の構文は次のいずれかです。

```
n SECOND[S]
n MINUTE[S]
n HOUR[S]
```

ここで *n* は 1～32767 の範囲の整数です。

BY TIME, INTO *number* を指定すると、MEASFT は、入力データファイルに存在していたエンティティを含む複数の出力データファイルを作成します。BY TIME, INTO *number* には、オプションで SECOND、MINUTE、HOUR、または PARTS の指定を続けることができます。

SECOND、MINUTE、HOUR、PARTS の指定なしに BY TIME, INTO *number* を指定する場合、または BY TIME, INTO *number* PARTS を指定する場合は、*number* には、入力ファイルを分割した結果の分割出力データファイルの数を、2〜100 の値で指定します。

BY TIME, INTO *number* に SECOND キーワードを続ける場合は、*number* で、それぞれの分割出力データファイルに格納される入力データの秒数を定義します。BY TIME, INTO *number* に MINUTE キーワードを続ける場合は、*number* で、それぞれの分割出力データファイルに格納される入力データの分数を定義します。*number* に HOUR キーワードを続ける場合は、*number* で、それぞれの分割出力データファイルに格納される入力データの時間数を定義します。SECOND、MINUTE、または HOUR を続ける場合、*number* には 1 から 32767 の値を指定できます。時間単位を指定すると、最後の分割出力データファイル内のデータは、他のデータよりも期間が短くなります。

分割出力データファイルの最大数は 100 です。BY TIME, INTO の指定で、100 を超える分割出力データファイルの作成を指示している場合、100 の分割出力データファイルだけが作成され、残りのデータレコードは 100 番目の分割出力データファイルに含まれます。たとえば、2 時間の測定を 1 分間の部分に分割すると、最初の 99 の分割出力データファイルには 1 分間のデータが含まれ、100 番目のデータファイルには 21 分間のデータが含まれます。

SPLIT コマンドの使用上の注意

- 間隔データファイルで SPLIT BY TIME コマンドを実行すると、分割ファイルの時間間隔の開始時刻と終了時刻に許容係数が含められます。この許容範囲は、LIST コマンドに FROM および TO オプションを使用するときに MEASCOM が採用するデフォルトの許容範囲に似ています。たとえば、分割ファイルの開始時刻と終了時刻がそれぞれ 5:00 と 6:00 である場合、および測定間隔が 10 分である場合、MEASFT は、分割ファイルにどのデータレコードを含めるかを決定するときに、実際には 4:55 の開始時刻と 6:05 の終了時刻を使用します。この許容範囲は、分割ファイルからの Measure レポートが元のファイルからの同じレポートと同一になるように追加されました。ただし、ビジー状態のシステムでは、MEASCTL または DP2 プロセスは、タイムリーに測定データファイルに書き込めない場合があります。さらに、CPU 間に（測定間隔に比べて）大幅なクロックスキューが生じることがあります。これらの状況によって、元のデータファイルと分割ファイル MEASCOM LIST 出力との間の正確性に影響する「エッジ効果」が作成されることがあります。正確さを確保するため、ユーザーは、対象の時間間隔よりもおよそ 1 つの間隔分だけ長くなるように時間間隔の境界を指定する必要があります。前出の例では、5:00 から 6:00 の間で信頼できる MEASCOM LIST 出力を取得するには、ユーザーは、分割ファイルを作成するときに、MEASFT SPLIT BY TIME コマンドで 4:50 の FROM 時刻と 6:10 の TO 時刻（または同等のオプション）を指定する必要があります。
- SPLIT 表示の「Time period」行には、入力データファイルで扱われる期間が反映されます。元の測定データファイルでは、測定の開始時刻と停止時刻です。TIME 分割データファイル、または TIME 分割データファイルの下位にあるファイルでは、「Time period」行にはデータファイルの時間間隔が反映されます。
- SPLIT 表示の「Time period」列には、作成された TIME 分割データファイルそれぞれの時間間隔が表示されます。ファイル内のすべてのデータレコードには、その期間内のタイムスタンプが必要です。間隔データファイルの場合、有効な時間間隔は、表示されている時間間隔の両側で間隔の半分だけ広くなります。
- SPLIT 表示の「%」列の数値は、入力データファイルに対する出力データファイルのサイズです。入力データファイルのすべてのデータレコードが一連の出力データファイルで分割されている場合、「%」の合計値は最大 100%まで追加可能になると予想できますが、その可能性は低くなります。構成レコード（すべてのジャーナルを含む）は各分割出力データファイルにコピーされます （さらに、入力データファイルが間隔データファイルの場合、時間間隔の境界の近くにあるタイムスタンプ付きのデータレコード（上記の「Time Period」列に関する注記を参照）は、隣接する両側の分割ファイルに表示されます）。入力データファイルが元のデータファイル（つまり MEASCOM によって作成されたファイル）である場合、そのファイルにはデータブロックの多くに余分なバイトが含まれます。MEASFT は、データレコードを分割出力データファイルに書き込む前に余分なバイトを削除するため、「%」の合計値が 100%を上回ったり下回ることがあります。

- SPLIT BY TIME コマンドを使用して分割したデータファイルのレポートまたはプロットを生成する場合は、MEASCOM LIST *entity-type*、LISTALL *entity-type*、または LIST PLOT コマンドで（TIME 分割ファイルデータファイルの時間間隔内で）FROM および TO オプションを指定して、元のデータファイルに対して行われた Measure のレポートおよびプロットと、TIME 分割データファイルに対して行ったものとの正確さを確保する必要があります。
- *out-file-set* としてアスタリスク（*）を指定し、*split-criteria* パラメーターを省略した場合、MEASFT は、Measure データファイルサイズを 1024 MB（1,073,741,824 バイト）で割り、端数を切り捨てて、出力データファイルの数を選択します。この数か 2 のいずれか大きい方を、出力データファイルの数として使用します。続いて、分割ができるだけ均等になるようにエンティティをグループ化します。次に例を示します。

```
$PERF.CAPAPP 113> MEASFT SPLIT BIGDATA,*
MEASURE File Tool - T9086H01 - (01AUG06) - \YOSPRD
Copyright 2006 Hewlett-Packard Company
SPLIT BIGDATA,*
File $PERF.CAPAPP.BIGDATA was split into the following files:
File                                    Size       %    Entity(s)
---------------------------------- ----------  ------
---------------------
$PERF.CAPAPP.ZMSRF000                 27391786    2.69  CPU,DISC,NETLINE,
                                                        SYSTEM,TMF,SQLPROC,

SQLSTMT,OSSCPU,OSSNS
$PERF.CAPAPP.ZMSRF001                 51509214    5.06  DISKFILE
$PERF.CAPAPP.ZMSRF002                 86759610    8.52  SERVERNET
$PERF.CAPAPP.ZMSRF003                145165294   14.26  DISCOPEN
$PERF.CAPAPP.ZMSRF004                165726444   16.28  FILE
$PERF.CAPAPP.ZMSRF005                190483848   18.71  PROCESS
$PERF.CAPAPP.ZMSRF006                337090908   33.11  PROCESSH
```

- BY TIME オプションの後でアスタリスク（*）を指定した場合、MEASFT はエラーメッセージ 3262 を表示します。

例

- 2012 年 3 月 16 日の 11:15 から 15 分間に発生するデータファイル MDFALLI からレコードを抽出し、新しいデータファイル MIN15 に書き込みます。「Time period」行は分割される入力データファイルの時間間隔です。「Time Period」列は、作成された各時間分割ファイルの時間間隔です。「%」列の数値は、入力データファイルに対する出力データファイルのサイズです。

```
$PERF.CAPAPP 63> MEASFT SPLIT MDFALLI,MIN15,BY TIME,FROM 11:15,FOR 15
MINUTES
MEASURE File Tool - T9086J04 - (01AUG12) - \BLPROD
(C) Copyright 2007-2012 Hewlett-Packard Development Company, L.P.
SPLIT MDFALLI,MIN15,BY TIME,FROM 11:15,FOR 15 MINUTES
File $PERF.CAPAPP.MDFALLI
Time period: 16 Mar 2012, 10:26:09 to 16 Mar 2012, 14:32:34
File                                    Size     %   Time Period
---------------------------------- ---------- ------
--------------------------
$PERF.CAPAPP.MIN15                   96185940  10.68 from 16 Mar 2012,
11:15:00
                                                     to   16 Mar 2012,
11:30:00
```

- Measure データファイル MDFALLI を MDFA*nn* というデータファイルに分割します。これらはそれぞれ 1 時間分の Measure データを含みます。このケースでは、入力データファイルには 4 時間を越えるデータが含まれているため、5 つの出力データファイルが作成されます。

```
$PERF.CAPAPP 149> MEASFT SPLIT MDFALLI,MDFA*,BY TIME,INTO 1 HOUR PARTS
MEASURE File Tool - T9086J04 - (01AUG12) - \BLPROD
(C) Copyright 2007-2012 Hewlett-Packard Development Company, L.P.
SPLIT MDFALLI,MDFA*,BY TIME,INTO 1 HOUR PARTS
File $PERF.CAPAPP.MDFALLI
Time period: 16 Mar 2012, 10:26:09 to 16 Mar 2012, 14:32:34
File                                    Size     %   Time Period
--------------------------------- ---------- ------
-------------------------
$PERF.CAPAPP.MDFA00                 255728012  28.40 from 16 Mar 2012,
10:26:09
                                                     to   16 Mar 2012,
11:26:09
$PERF.CAPAPP.MDFA01                 278494900  30.93 from 16 Mar 2012,
11:26:09
                                                     to   16 Mar 2012,
12:26:09
$PERF.CAPAPP.MDFA02                 190677148  21.18 from 16 Mar 2012,
12:26:09
                                                     to   16 Mar 2012,
13:26:09
$PERF.CAPAPP.MDFA03                 182919124  20.31 from 16 Mar 2012,
13:26:09
                                                     to   16 Mar 2012,
14:26:09
$PERF.CAPAPP.MDFA04                  25952028   2.88 from 16 Mar 2012,
14:26:09
                                                     to   16 Mar 2012,
14:32:34
```

- 出力データファイル名または分割条件を指定せずに Measure データファイル MDATA5 を分割します。MEASFT は、できるだけ均等にするために双方向の分割を作成しようとします。このケースでは、1 つのファイルに PROCESS レコードおよび FILE レコードを配置し、別のファイルに他のすべてのレコードを配置します。

```
$PERF.CAPAPP 7> MEASFT SPLIT MDATA5
MEASURE File Tool - T9086H01 - (01AUG06) - \YOSPRD
Copyright 2006 Hewlett-Packard Company
SPLIT MDATA5
File $PERF.CAPAPP.MDATA5 was split into the following files:
File                                    Size       %    Entity(s)
---------------------------------- ----------  ------
---------------------
$PERF.CAPAPP.ZMSRF000                 55990206   41.79  CPU,DISCOPEN,DISC,

NETLINE,TMF,SQLPROC,
                                                       SQLSTMT,SERVERNET,

DISKFILE,OSSCPU,OSSNS
$PERF.CAPAPP.ZMSRF001                 76588122   57.16  PROCESS,FILE
```

- Measure データファイル MDATA5 をエンティティごとに分割し、各エンティティが別々の出力データファイルに含まれるようにします。MEASFT が出力データファイルの名前を指定します。

```
$PERF.CAPAPP 8> MEASFT SPLIT MDATA5,*,BY ENTITY *
MEASURE File Tool - T9086H01 - (01AUG06) - \YOSPRD
Copyright 2006 Hewlett-Packard Company
```

```
SPLIT MDATA5,*,BY ENTITY *
File $PERF.CAPAPP.MDATA5 was split into the following files:
File                                    Size       %    Entity(s)

---------------------------------- ----------  ------
--------------------
$PERF.CAPAPP.ZMSRF000                   133842    0.10  CPU
$PERF.CAPAPP.ZMSRF001                 41239732   30.78  PROCESS
$PERF.CAPAPP.ZMSRF004                 35349176   26.38  FILE
$PERF.CAPAPP.ZMSRF005                 25679600   19.17  DISCOPEN
$PERF.CAPAPP.ZMSRF006                  2032272    1.52  DISC
$PERF.CAPAPP.ZMSRF009                   174430    0.13  NETLINE
$PERF.CAPAPP.ZMSRF013                    30552    0.02  TMF
$PERF.CAPAPP.ZMSRF014                    10742    0.01  SQLPROC
$PERF.CAPAPP.ZMSRF015                    83114    0.06  SQLSTMT
$PERF.CAPAPP.ZMSRF017                 10586256    7.90  SERVERNET
$PERF.CAPAPP.ZMSRF018                 17086666   12.75  DISKFILE
$PERF.CAPAPP.ZMSRF019                   156412    0.12  OSSCPU
$PERF.CAPAPP.ZMSRF020                    24180    0.02  OSSNS
```

- Measure データファイル MDATA5 をエンティティごとに分割し、FILE データが SFFILE という名前のファイルに含まれ、残りのデータが SFREST に含まれるようにします。

```
$PERF.CAPAPP 10> MEASFT SPLIT MDATA5,(SFFILE,SFREST),BY ENTITY (FILE,*)
MEASURE File Tool - T9086H01 - (01AUG06) - \YOSPRD
Copyright 2006 Hewlett-Packard Company
SPLIT MDATA5,(SFFILE,SFREST),BY ENTITY (FILE,*)
File $PERF.CAPAPP.MDATA5 was split into the following files:

File                                    Size       %    Entity(s)
---------------------------------- ----------  ------
---------------------
$PERF.CAPAPP.SFFILE                   35349176   26.38  FILE
$PERF.CAPAPP.SFREST                   97229152   72.57
CPU,PROCESS,DISCOPEN,
                                                       DISC,NETLINE,TMF,
                                                       SQLPROC,SQLSTMT,
                                                       SERVERNET,DISKFILE,
                                                       OSSCPU,OSSNS
```

- Measure データファイル MDATA5 をエンティティごとに分割し、CPU データおよび PROCESS データが SFFILE1 という名前のファイルに含まれ、SERVERNET データが SFFILE2 に含まれるようにします。

```
$PERF.CAPAPP 14> MEASFT SPLIT MDATA5,(SFFILE1,SFFILE2),
                              BY ENTITY (CPU PROCESS,SERVERNET)
MEASURE File Tool - T9086H01 - (01AUG06) - \YOSPRD

Copyright 2006 Hewlett-Packard Company

SPLIT MDATA5,(SFFILE1,SFFILE2),BY ENTITY (CPU PROCESS,SERVERNET)
File $PERF.CAPAPP.MDATA5 was split into the following files:
File                                    Size       %    Entity(s)
---------------------------------- ----------  ------
---------------------
$PERF.CAPAPP.SFFILE1                  41372788   30.88  CPU,PROCESS
```

```
$PERF.CAPAPP.SFFILE2                    10586256    7.90  SERVERNET
```

- Measure データファイル MDATA5 をプロセッサー番号ごとに分割し、各プロセッサーのデータが別々の出力データファイルに含まれるようにします。出力データファイルの名前は「MCPU」で始まります。

```
$PERF.CAPAPP 16> MEASFT SPLIT MDATA5,MCPU*,BY CPU *
MEASURE File Tool - T9086H01 - (01AUG06) - \YOSPRD
Copyright 2006 Hewlett-Packard Company
SPLIT MDATA5,MCPU*,BY CPU *
File $PERF.CAPAPP.MDATA5 was split into the following files:
File                                    Size       %    CPU(s)
----------------------------------- ----------  ------
---------------------
$PERF.CAPAPP.MCPU00                   28410588   21.21  0
$PERF.CAPAPP.MCPU01                   19602928   14.63  1
$PERF.CAPAPP.MCPU02                   41879174   31.26  2
$PERF.CAPAPP.MCPU03                   42687210   31.86  3
```

HELP コマンド

HELP コマンドは、公開済みコマンドに関するヘルプ情報を表示します。

構文

```
HELP | empty
```

MEASFT コマンドのヘルプテキストを表示するには、「MEASFT HELP」と入力するか、「MEASFT」とだけ入力します。

MEASFT のエラーメッセージと警告メッセージ

MEASFT を実行すると、次のいずれかのメッセージがコンソールに表示されることがあります。次の例は、標準的なエラーおよび警告の形式を示しています。

```
MEASURE File Tool - T9086H01 - (01AUG06) - \YOSPRD
Copyright 2006 Hewlett-Packard Company

SPLIT KMEAS3, ,BY CPU *
              ^
*** Error 3232: SPLIT name Resolution failed, File System Error = #13

$PERF KMZMFT 45> MEASFT SPLIT KMEAS2,*,BY ENTITY (CPU TERMINAL,*)

MEASURE File Tool - T9086H01 - (01AUG06) - \YOSPRD
Copyright 2006 Hewlett-Packard Company

SPLIT KMEAS2,*,BY ENTITY (CPU TERMINAL,*)

*** Warning 2007: No data for this specified entity, TERMINAL

File $PERF.KMZMFT.KMEAS2 was split into the following files:

File                                Size       %    Entity(s)
------------------------------ ----------  ------  ----------------------
```

```
$PERF.KMZMFT.ZMSRF000              118098    0.60  CPU
$PERF.KMZMFT.ZMSRF001            18679336   95.33  PROCESS,PROCESSH,
                                                   SERVERNET
```

メッセージが構文エラーを示す場合は、構文エラーが検出された場所に曲折アクセント符号（^）が表示されます。

MEASFT 2000（WARN_SPLIT_FILE_NOT_CREATED）

Specified file, *out-file*, not created because of lack of data records.

メッセージの説明

原因

ファイル内に配置するデータレコードが MEASFT で見つからなかったため、指定された出力データファイルが作成されませんでした。

影響

SPLIT コマンドは続行しますが、指定された出力ファイルは作成されません。

リカバリ

指定された出力ファイルと対応する分割条件をコマンドラインから削除してください。

MEASFT 2001（WARN_SPLIT_FILE_PURGED）

Specified output filename, *out-file*, already existed, previous data was overwritten.

メッセージの説明

原因

（直接またはワイルドカードで間接的に）指定された出力ファイル名はすでに存在しており、分割されるファイルである入力データファイルと同じではありません。

影響

SPLIT コマンドは続行し、以前のデータが上書きされます。

リカバリ

上書きされたファイルのデータを回復することはできません。今後は既存のファイルと競合しない出力ファイル名を指定してください。

MEASFT 2002（WARN_NO_XRTIMES_AT_EOF）

No Stop Measurement record, file possibly truncated/corrupted.

メッセージの説明

原因

入力データファイルに Stop Measurement レコードがありません。ファイルは切り捨てられたコピーであるか、ファイルが破損しています。

影響

INFO または SPLIT コマンドは続行します。

リカバリ

レコードが欠落した原因を確認し、これが重大な問題であるかどうか判断してください。

MEASFT 2003（WARN_BAD_SVNET_ENT_LEN）

```
Bad length in ENDPT/SVNET, DISC or DEVICE entity, entity-name, length
#length, record #number
```

メッセージの説明

原因

ENDPOINT/SERVERNET、DISC、または DEVICE エンティティ *entity-name* のレコードが、予期しない長さ *length* です。レコード番号は *number* です。

影響

INFO または SPLIT コマンドは続行します。

リカバリ

この警告について Hewlett Packard Enterprise の担当者に問い合わせ、原因となったデータファイルを保存してください。

MEASFT 2004（WARN_BAD_ENT_LEN）

```
Bad length in entity, entity-name, length #length, record #number
```

メッセージの説明

原因

エンティティ *entity-name* のレコードが、予期しない長さ *length* です。レコード番号は *number* です。

影響

INFO または SPLIT コマンドは続行します。

リカバリ

この警告について Hewlett Packard Enterprise の担当者に問い合わせ、原因となったデータファイルを保存してください。

MEASFT 2005（WARN_TOO_MANY_SPLIT_FILES）

```
More output files specified than entities present; # files lowered from mm to
nn
```

メッセージの説明

原因

Measure データファイルには、指定された分割ファイルよりも少ないエンティティについてのデータレコードが含まれているため、MEASFT は指定された分割を実行できません。この警告は、ユーザーが出力ファイルの明確な数を指定したが、分割条件を指定しなかったため、MEASFT に分割条件を決定させた場合にのみ発生します。

影響

SPLIT コマンドは続行します。MEASFT は余分な出力ファイル名を無視します。

リカバリ

指定された出力データファイルの数を減らしてください。

MEASFT 2006（WARN_NO_SPLIT_FILES_CREATED）

```
No output files were created, no records satisfied split criteria.
```

メッセージの説明

原因

Measure データファイルのすべてのレコードがいずれの分割条件も満たさなかったため、出力ファイルが作成されていません。

影響

SPLIT コマンドは続行します。

リカバリ

レコードが欠落した原因を確認し、これが重大な問題であるかどうか判断してください。

MEASFT 2007（WARN_ENTITY_HAS_NO_DATA）

`No data for this specified entity, entity`

メッセージの説明

原因

分割条件で指定されたエンティティのデータレコードが見つかりませんでした。

影響

SPLIT コマンドは続行します。指定された出力データファイルで表されたエンティティのデータがデータファイルに含まれていない場合（たとえば、出力ファイルに PROCESS エンティティおよび DISK エンティティのデータが含まれる予定であるが、分割されるデータファイルに、これらのどちらのエンティティのレコードも含まれない場合）、MEASFT は、その出力ファイルが作成されなかったことを示すメッセージ 2000 も表示します。いずれかの出力データファイルに含まれるエンティティのデータがデータファイルに格納されていない場合、MEASFT は、出力データファイルが作成されなかったことを示すメッセージ 2006 も表示します。

リカバリ

データが欠落した原因を確認し、これが重大な問題であるかどうか判断してください。

MEASFT 2008（WARN_REMAINING_ENTITY_HAS_NO_DATA）

`Output file for the remaining entities (*) was not created due to lack of data.`

メッセージの説明

原因

SPLIT コマンドでワイルドカード（*）で指定された *entity-list* を持つ最後の出力ファイルのデータレコードが見つかりませんでした。

影響

SPLIT コマンドは続行します。指定された出力ファイルは作成されません。

リカバリ

データが欠落した原因を確認し、これが重大な問題であるかどうか判断してください。

MEASFT 2009（WARN_CPU_HAS_NO_DATA）

`No data for this specified CPU, nn`

メッセージの説明

原因

SPLIT コマンドの分割条件で指定された CPU *nn* で、データレコードが検出されませんでした。

影響

SPLIT コマンドは続行します。出力ファイルに表示される任意の CPU のデータレコードがデータファイルに格納されない場合、メッセージ 2000 も表示されます。

リカバリ

データが欠落した原因を確認し、これが重大な問題であるかどうか判断してください。

MEASFT 2010（WARN_REMAINING_CPU_HAS_NO_DATA）

```
Output file for the remaining CPUs (*) was not created due to lack of data.
```

メッセージの説明

原因

SPLIT コマンドでワイルドカード（*）で指定された *cpu-list* を持つ最後の出力ファイルでデータレコードが見つかりませんでした。

影響

SPLIT コマンドは続行します。指定された出力ファイルは作成されません。

リカバリ

データが欠落した原因を確認し、これが重大な問題であるかどうか判断してください。

MEASFT 2011（WARN_SPLIT_FILE_NOT_CREATED_2）

```
An output file was not created due to lack of data.
```

メッセージの説明

原因

そのファイルの分割条件を満たすデータレコードが MEASFT で見つからなかったため、出力ファイル名は作成されませんでした。この警告は常にメッセージ 2007 の後に続き、出力ファイルセット内でアスタリスクまたはアスタリスクの後ろのプレフィックスが指定された場合のみ発生します。

影響

SPLIT コマンドは続行します。指定された出力ファイルは作成されません。

リカバリ

このメッセージに関連付けられた出力ファイルの分割条件を削除してください。問題の原因となった分割条件を把握するには、前に発生したメッセージ 2007 を参照してください。

MEASFT 2032（WARN_MISSING_JOURNAL）

journal-type Journal missing。

メッセージの説明

原因

予想されるジャーナルが検出されませんでした。*journal-type* は OSS または SQL を使用できます。

影響

コマンドは続行します。

リカバリ

Measure データファイルのファイルサイズをチェックして、ファイルが切り捨てられたかどうかを確認してください。切り捨ては、非常に大きなファイルを転送するときに生じることがあります。ファイルが切

り捨てられていない場合、破損している可能性があります。これらのどちらのケースも該当しない場合、Hewlett Packard Enterprise の担当者にこのメッセージを報告してください。

MEASFT 2033（WARN_READ_INVALID_DATA）

```
An invalid record was found in Measure data file
```

メッセージの説明

原因

Measure データファイルに無効なレコードがあります。データファイルが収集されたディスクがいっぱいになったか、MEASCOM SWAP ボリュームがいっぱいになったため、レコードが無効になった可能性があります。

影響

コマンドは、無効なレコードを Measure データファイルの EOF としてマークして続行します。データファイル内の残りのレコードは無視されます。

リカバリ

Measure データファイルが収集されたディスクのサイズをチェックするか、測定システムの MEASCOM SWAP ボリュームのサイズをチェックしてください。これらがいっぱいの場合、ファイルを他のディスクに再配置するか、ファイルを削除して空き容量を増やし、測定を再実行してください。

MEASFT 3000（ERR_FNAME_RESOLVE）

```
Measfile Name Resolution failed, File System Error = #nn
```

メッセージの説明

原因

Measure データファイル名が指定されましたが、FILENAME_RESOLVE_で解決できませんでした。

影響

SPLIT コマンドまたは INFO コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、指定された Measure データファイル名が解決できなかった理由を確認してください。ファイル名の構文が正しくないことが最も可能性の高い原因です。

MEASFT 3001（ERR_FNAME_UNRESOLVE）

```
Measfile Name UnResolution failed, File System Error = #nn
```

メッセージの説明

原因

Measure データファイル名が指定されましたが、FILENAME_UNRESOLVE_で未解決にできませんでした。

影響

SPLIT コマンドまたは INFO コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、指定された Measure データファイル名が未解決にできなかった理由を確認してください。このエラーは、FILENAME_RESOLVE_によってファイル名がすでに解決されているため、発生する可能性は高くありません。

MEASFT 3002（ERR_OPEN_MEASFILE）

```
Measfile Open failed, File System Error = #nn
```

メッセージの説明

原因

Measure データファイル名が指定されましたが、開くことができませんでした。

影響

SPLIT コマンドまたは INFO コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、指定された Measure データファイルを開くことができなかった理由を確認してください。考えられる原因は、データファイルが見つからないか、別のユーザーによって使用中か、またはユーザーから読み取りアクセスできるようにセキュリティ保護されていないことです。

MEASFT 3005（ERR_CREATE_SPLIT_FILE）

```
Creation of output file, datafile, failed, File System Error = #nn
```

メッセージの説明

原因

出力ファイル *datafile* を作成できませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、指定された出力ファイルを作成できなかった理由を確認してください。考えられる原因は、ボリューム名のスペルが間違っているか、サブボリュームがユーザーから書き込みアクセスできるようにセキュリティ保護されていないか、ボリュームがいっぱいであることです。

MEASFT 3006（ERR_OPEN_SPLIT_FILE）

```
Open of output file, datafile, failed, File System Error = #nn
```

メッセージの説明

原因

出力ファイル *datafile* を開くことができませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、指定された出力ファイルを開くことができなかった理由を確認してください。出力ファイルが正しく作成されたばかりであるため、このエラーが発生する可能性は高くありません。

MEASFT 3007（ERR_FILE_SETPOSITION）

```
FILE_SETPOSITION_ failed, File System Error = #nn
```

メッセージの説明

原因

Measure データファイルにジャーナルセクションがある場合、データファイルの位置をファイルの先頭またはファイルの終わり付近に変更できませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、Measure データファイルの位置を変更できなかった理由を確認してください。

MEASFT 3008（ERR_WRITEX_SPLIT_FILE）

```
Write of output file, datafile, failed, File System Error = #nn
```

メッセージの説明

原因

MEASFT は出力ファイル *datafile* への書き込み中にエラーが発生しました。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。作成された出力ファイルは不完全になります。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、エラーの原因を確認してください。

MEASFT 3009（ERR_CLOSE_SPLIT_FILE）

```
Close of output file, datafile, failed, File System Error = #nn
```

メッセージの説明

原因

MEASFT は出力ファイル *datafile* のクローズ中にエラーが発生しました。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。作成された出力ファイルは不完全になる可能性が高くなります。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、エラーの原因を確認してください。

MEASFT 3010（ERR_FILE_SETPOS_SPLIT_FILE）

```
FILE_SETPOSITION_ of output file, datafile, failed, File System Error = #nn
```

メッセージの説明

原因

MEASFT で、先頭レコードのジャーナルポインターを更新するために出力ファイル *datafile* をファイルの先頭に再配置するときにエラーが発生しました。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。作成された出力ファイルには正しいジャーナルポインターがありません。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、エラーの原因を確認してください。

MEASFT 3011（ERR_READX_SPLIT_FILE）

`Read of output file, datafile, failed, File System Error = #nn`

メッセージの説明

原因

MEASFT で、出力ファイル *datafile* の先頭レコードを読み取ってレコード内のジャーナルポインターを更新しようとしたときに、エラーが発生しました。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。作成された出力ファイルには正しいジャーナルポインターがありません。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、エラーの原因を確認してください。

MEASFT 3012（ERR_WRUPDX_SPLIT_FILE）

`Writeupdate of output file, datafile, failed, File System Error = #nn`

メッセージの説明

原因

MEASFT で、出力ファイル *datafile* の先頭レコードに書き込みを行ってレコード内のジャーナルポインターを更新しようとしたときに、エラーが発生しました。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。作成された出力ファイルには正しいジャーナルポインターがない可能性があります。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、エラーの原因を確認してください。

MEASFT 3013（ERR_READX_MEASFILE）

`Measfile READX failed, File System Error = #nn`

メッセージの説明

原因

MEASFT で Measure データファイルのジャーナルレコードの読み取り中にエラーが発生しました。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。作成された出力ファイルは完全ではありません。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、エラーの原因を確認してください。

MEASFT 3014（ERR_FILE_NOT_FOUND）

`Measure data file not found.`

メッセージの説明

原因

MEASFT で Measure データファイルが見つかりませんでした。

影響

SPLIT コマンドまたは INFO コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

Measure データファイルの正しい場所を入力してください。

MEASFT 3016（ERR_FILE_GETINFO）

`FILE_GETINFOLISTBYNAME_ of` *`datafile`* `failed, File System Error = #`*`nn`*`.`

メッセージの説明

原因

MEASFT で指定されたデータファイルの情報を取得できませんでした。

影響

SPLIT コマンドまたは INFO コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、エラーの原因を確認してください。

MEASFT 3017（ERR_FILE_BAD_TYPE）

`Measure file not unstructured, file type encountered #`*`n`*

メッセージの説明

原因

指定された Measure データファイルは非構造化されていません。

影響

SPLIT コマンドまたは INFO コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

Measure データファイルは非構造化ファイルであるのに対して、指定されたファイルはファイルタイプ *n* です。正しいファイル名を入力してください。

MEASFT 3018（ERR_FILE_BAD_FILECODE）

`Measure file code not 175, file code encountered #`*`n`*

メッセージの説明

原因

指定された Measure データファイルにファイルコード 175 がありません。

影響

SPLIT コマンドまたは INFO コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

Measure データファイルはファイルコード 175 を持つのに対して、指定されたファイルはファイルコード *n* です。正しいファイル名を入力してください。

MEASFT 3019（ERR_SPLIT_FILE_NAME_ERR）

Specified output filename, *outfile*, is the same as the file to be split.

メッセージの説明

原因

（直接またはワイルドカードで間接的に）指定された出力ファイル名は、分割されるファイルである入力データファイルと同じです。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

入力データファイル名と競合しない出力ファイル名を再指定してください。

MEASFT 3020（ERR_CREATE_SPLIT_FILE_DUP）

Specified output filename, *outfile*, already exists and is not a measure data file.

メッセージの説明

原因

（直接またはワイルドカードで間接的に）指定された出力ファイル名はすでに存在しており、Measure データファイルではありません。MEASFT では、既存の Measure データファイルではないファイルは上書きされません。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

既存のファイル名と競合しない出力ファイル名を再指定してください。

MEASFT 3021（ERR_PURGE_SPLIT_FILE）

Purging output filename, *datafile*, failed, File System Error = #*nn*

メッセージの説明

原因

出力ファイル *datafile* をパージできませんでした。同じ名前の出力ファイルが指定されている場合、MEASFT は既存の Measure データファイルを上書きします。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、指定された出力ファイルをパージできなかった理由を確認してください。

MEASFT 3022（ERR_SEG_ALLOCATE_）

SEGMENT_ALLOCATE_ error #*nn*.

メッセージの説明

原因

MEASFT は実行するための十分なメモリを割り当てできませんでした。

影響

コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

SEGMENT_ALLOCATE_エラーコード *nn* を使用して、メモリを割り当てできなかった理由を確認してください。MEASFT では、実行には約 3MB のメモリが必要です。

MEASFT 3023（ERR_USESEGMENT）

```
USESEGMENT error.
```

メッセージの説明

原因

MEASFT は実行するための十分なメモリにアクセスできませんでした。

影響

コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

このメッセージを Hewlett Packard Enterprise の担当者に報告してください。

MEASFT 3024（ERR_CRL_BAD_REC_TYPE）

```
Bad record type #mm in this context #nn
```

メッセージの説明

原因

MEASFT は予期しないレコードタイプを検出しました。

影響

コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

Measure データファイルが壊れているか、MEASFT に障害があります。このメッセージを Hewlett Packard Enterprise の担当者に報告し、このメッセージに表示されているレコードタイプ *mm* およびコンテキスト *nn* を伝えてください。

MEASFT 3025（ERR_CONTAB_TRLR_WRONG_CTXT）

```
Unexpected CONTAB^TRAILER record in MF_NORMAL context.
```

メッセージの説明

原因

MEASFT は、予期しないレコードタイプを検出しました。

影響

コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

Measure データファイルが壊れているか、MEASFT に障害があります。このメッセージを Hewlett Packard Enterprise の担当者に報告してください。

MEASFT 3027（ERR_BAD_RECORD_STATE）

```
Bad record state in record #number
```

メッセージの説明

原因

MEASFT は、レコード *number* で予期しないレコード状態を検出しました。

影響

コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

Measure データファイルが壊れているか、MEASFT に障害があります。このメッセージを Hewlett Packard Enterprise の担当者に報告してください。

MEASFT 3028（ERR_BAD_DFFIRST_REC）

```
Measure data file corrupt, 1st record wrong length, file version Xnn
```

メッセージの説明

原因

MEASFT は、バージョンが *Xnn* である Measure データファイル内の先頭レコードを認識しませんでした。

影響

コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

Measure データファイルが壊れているか、MEASFT に障害があります。このメッセージを Hewlett Packard Enterprise の担当者に報告してください。

MEASFT 3029（ERR_UNEXPECTED_NONCTR_REC）

```
Unexpected non-counter record #number
```

メッセージの説明

原因

MEASFT は、レコード *number* で予期しないレコードタイプを検出しました。

影響

コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

Measure データファイルが壊れているか、MEASFT に障害があります。このメッセージを Hewlett Packard Enterprise の担当者に報告してください。

MEASFT 3030（ERR_UNSUPPORTED_VERSION）

```
Unsupported Measure data file version, Xnn
```

メッセージの説明

原因

MEASFT は、Measure の *Xnn* バージョンによって作成された Measure データファイルバージョンを検出しました。

影響

コマンドは完了しません。MEASFT はエラーで終了します

リカバリ

MEASFT は、G10 より前の Measure バージョンで作成されたデータファイルをサポートしません。

MEASFT 3031（ERR_NO_DFFIRST_REC）

`Measure data file corrupt, 1st record not present.`

メッセージの説明

原因

MEASFT は、Measure データファイル内の先頭レコードを認識しませんでした。

影響

コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

Measure データファイルが壊れているか、MEASFT に障害があります。このメッセージを Hewlett Packard Enterprise の担当者に報告してください。

MEASFT 3033（ERR_REC_LENGTH_COMP_ERR）

`Record length could not be computed, record #`*`nnnnn`*

メッセージの説明

原因

MEASFT は、MEASFT の障害または Measure データファイルの破損が原因で、レコード番号 *nnnnn* の長さを計算できませんでした。

影響

コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

Measure データファイルが正しいと思われる場合は、Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 3034（ERR_REC_LENGTH_TOO_BIG）

`Record length too big, record #`*`nnnnn`*

メッセージの説明

原因

レコードの番号 *nnnnn* の長さが大きすぎます。このエラーは MEASFT の障害または Measure データファイルの破損が原因です。

影響

コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

Measure データファイルが正しいと思われる場合は、Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 3035（ERR_BAD_RECORD_TYPE）

```
Invalid record type, record #nnnnn
```

メッセージの説明

原因

MEASFT は、レコード番号 *nnnnn* で無効なレコードタイプを検出しました。Measure データファイルが破損しているか、指定された入力データファイルが Measure データファイルでないことが原因の可能性があります。

影響

コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

指定された Measure データファイルが正しいことを確認してください。

MEASFT 3036（ERR_READ_NO_DATA_ERR）

```
File error, read no data, File System Error = #nn, record #nnnnn
```

メッセージの説明

原因

MEASFT がデータを読み取らず、レコード番号 *nnnnn* でファイルシステムエラー *nn* を検出しました。Measure データファイルが破損しているか、指定された入力データファイルが Measure データファイルでないことが原因の可能性があります。

影響

コマンドは完了しません。MEASFT はエラーで終了します。いくつかの出力ファイルは作成されている可能性があります。

リカバリ

指定された Measure データファイルが正しいことを確認してください。

MEASFT 3038（ERR_NO_XRTIMES_RECORD）

```
No Start Measurement record in the input data file.
```

メッセージの説明

原因

SPLIT BY TIME コマンドが発行されましたが、入力データファイルに Start Measurement レコードがありません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

Measure データファイルが壊れているか、MEASFT に障害があります。このメッセージを Hewlett Packard Enterprise の担当者に報告してください。

MEASFT 3039（ERR_BAD_XRTIMES_TIMESTAMP）

```
Bad Start Measurement record in the input data file.
```

メッセージの説明

原因

SPLIT BY TIME コマンドが発行されましたが、Start Measurement レコードに不正なタイムスタンプがあります。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

Measure データファイルが壊れているか、MEASFT に障害があります。このメッセージを Hewlett Packard Enterprise の担当者に報告してください。

MEASFT 3040（ERR_COMPUTETIMESTAMP）

```
COMPUTETIMESTAMP error, mask = #mm.
```

メッセージの説明

原因

SPLIT BY TIME コマンドが発行されましたが、Start Measurement レコードのタイムスタンプによって COMPUTETIMESTAMP エラーが発生しました。これは内部 MEASFT エラーです。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

このメッセージを Hewlett Packard Enterprise の担当者に報告し、このメッセージに表示されているマスク *mm* を伝えてください。

MEASFT 3041（ERR_SEG_ALLOCATE64）

```
SEGMENT_ALLOCATE64_ error #nn, size #ss
```

メッセージの説明

原因

MEASFT は十分な 64 ビットメモリを割り当てできませんでした。

影響

コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

SEGMENT_ALLOCATE64_エラーコード*#nn* およびサイズ*#ss* を使用して、メモリを割り当てできなかった理由を確認してください。

MEASFT 3042（ERR_SEG_RESIZE64）

```
SEGMENT_RESIZE64_ error #nn, resize #news, oldsize #olds
```

メッセージの説明

原因

MEASFT は十分な 64 ビットメモリをサイズ変更できませんでした。

影響

コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

SEGMENT_RESIZE64_エラーコード*#nn*、新しいサイズ*#news*、および古いサイズ*#olds* を使用して、メモリのサイズ変更ができなかった理由を確認してください。

MEASFT 3043（ERR_SIG_HANDLER）

`Signal <signal>`

メッセージの説明

原因

シグナルがキャッチされました。

影響

開いている出力ファイルはパージされ、開いているファイルは閉じられ、MEASFT はエラーで終了します。

リカバリ

シグナルについて詳しくは、*オープンシステムサービスユーザーズガイド*を参照してください。

MEASFT 3200（ERR_PARSE_NOT_KEYWORD）

XXX is not a recognized keyword.

メッセージの説明

原因

コマンドラインの文字列 *XXX* は有効なキーワードではありません。

影響

コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

正しいキーワードまたはコマンドの構文を入力してください。

MEASFT 3201（ERR_PARSE_NO_FILENAME）

filename parameter is missing.

メッセージの説明

原因

Measure データファイルがコマンドラインで指定されませんでした。

影響

コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

Measure データファイル名を入力してください。

MEASFT 3202（ERR_PARSE_FILENAME_TOO_LONG）

filename parameter too long.

メッセージの説明

原因

コマンドラインで入力された Measure データファイル名が長すぎます。

影響

コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

正しい Measure データファイル名を入力してください。

MEASFT 3230（ERR_PARSE_SPLIT_NAME_TOO_LONG）

SPLIT *name* parameter too long.

メッセージの説明

原因

指定された出力ファイル名が長すぎます。このエラーは、8 バイトを超える非修飾出力ファイル名が入力されたときに発生します。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

正しい出力ファイル名を入力してください。

MEASFT 3231（ERR_PARSE_SPLIT_PREFIX_TOO_LONG）

SPLIT *name-prefix* parameter too long.

メッセージの説明

原因

指定された出力ファイル名のプレフィックスが長すぎます。最大のプレフィックスの長さは 6 バイトです。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

短い出力ファイルプレフィックスを入力してください。

MEASFT 3232（ERR_PARSE_SPLIT_FNAME_RESOLVE）

SPLIT *name* Resolution failed, File System Error = #*nn*

メッセージの説明

原因

指定された出力ファイル名が、FILENAME_RESOLVE_で解決できませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、エラーの原因を確認してください。

MEASFT 3233（ERR_PARSE_SPLIT_FNAME_UNRESOLVE）

```
SPLIT name UnResolution failed, File System Error = #nn
```

メッセージの説明

原因

指定された出力ファイル名が、FILENAME_UNRESOLVE_で未解決にできませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

表示されているファイルシステムのエラー番号 *nn* を使用して、エラーの原因を確認してください。このエラーは、FILENAME_RESOLVE_によって出力ファイル名がすでに解決されているため、発生する可能性は高くありません。

MEASFT 3234（ERR_PARSE_SPLIT_RPAREN_MISSING）

```
SPLIT file specs right parenthesis missing.
```

メッセージの説明

原因

SPLIT ファイル指定の項目が、左カッコで始まり、右カッコがありません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します

リカバリ

SPLIT ファイル指定の構文を修正してください。

MEASFT 3235（ERR_PARSE_SPLIT_BY_EXPECTED）

```
SPLIT 'BY' keyword missing.
```

メッセージの説明

原因

SPLIT 分割条件の項目は「BY」で始めなければなりません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します

リカバリ

SPLIT 分割条件の構文を修正してください。

MEASFT 3236（ERR_PARSE_SPLIT_BY_KW_EXPECTED）

```
SPLIT 'BY CPU' or 'BY ENTITY' expected.
```

メッセージの説明

原因

SPLIT 分割条件の項目は「BY CPU」または「BY ENTITY」で始めなければなりません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します

リカバリ

SPLIT 分割条件の構文を修正してください。

MEASFT 3237（ERR_PARSE_SPLIT_SC_RPAREN_MISSING）

```
SPLIT split criteria right parenthesis missing.
```

メッセージの説明

原因

SPLIT 分割条件の項目が、左カッコで始まり、右カッコがありません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します

リカバリ

SPLIT 分割条件の構文を修正してください。

MEASFT 3238（ERR_PARSE_SPLIT_SC_MISMATCH）

```
SPLIT file specs and split criteria mismatch.
```

メッセージの説明

原因

SPLIT ファイル指定と分割条件の項目は、指定された出力ファイルの数が一致しません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT ファイル指定または分割条件構文あるいはその両方を訂正して、それらが同じ数の出力ファイルを指定するようにしてください。

MEASFT 3239（ERR_PARSE_SPLIT_BAD_CPU_NUM2）

```
SPLIT criteria CPU number out of range.
```

メッセージの説明

原因

SPLIT コマンドの分割条件で指定された CPU が、範囲（0 から 15）の範囲外であるか、数値ではありません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

無効な CPU 番号を修正してください。

MEASFT 3240（ERR_PARSE_SPLIT_SC_CPU_WC_ERR1）

SPLIT criteria CPU '*' must be in last *cpu-list*.

メッセージの説明

原因

SPLIT コマンドの分割条件で指定されたワイルドカード（*）が、最後の *cpu-list* にありませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY CPU 項目を修正して、ワイルドカード *cpu-list* がリストの最後の項目になるようにしてください。

MEASFT 3241（ERR_PARSE_SPLIT_SC_CPU_WC_ERR2）

`SPLIT criteria CPU '*' must be only item in last` *`cpu-list`*`.`

メッセージの説明

原因

SPLIT コマンドの分割条件で指定されたワイルドカード（*）が、最後の *cpu-list* の唯一の項目ではありませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY CPU 項目を修正して、ワイルドカード *cpu-list* が最後のリストにあり、ワイルドカードがそのリスト内の唯一の項目になるようにしてください。

MEASFT 3242（ERR_PARSE_SPLIT_WC_ERR1）

`SPLIT criteria '*' can not be prepended to anything.`

メッセージの説明

原因

SPLIT コマンドの分割条件で指定されたワイルドカード（*）を、コマンドの先頭に付加することはできません。リストが指定されない場合は、コマンドラインの最後の項目でなければなりません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT 分割条件を修正して、ワイルドカードがコマンドラインの最後の要素になるようにしてください。

MEASFT 3243（ERR_PARSE_SPLIT_WC_ERR2）

`SPLIT criteria '*' requires that` *`out-file-set`* `be a '*'.`

メッセージの説明

原因

ワイルドカード（*）が SPLIT コマンドの分割条件で指定された場合、ファイルセットもワイルドカードである必要があります。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT 分割条件を修正して、ファイル指定と分割条件の両方にワイルドカードがあるか、両方ともワイルドカードがないようにしてください。

MEASFT 3244（ERR_PARSE_SPLIT_SC_EMPTY）

```
SPLIT criteria empty.
```

メッセージの説明

原因

SPLIT コマンドの分割条件のカッコ内に何も指定されませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

目的の SPLIT 分割条件を入力してください。

MEASFT 3245（ERR_PARSE_SPLIT_SC_ENT_WC_ERR1）

SPLIT criteria Entity '*' must be in last *entity-list*.

メッセージの説明

原因

SPLIT コマンドの分割条件で指定されたワイルドカード（*）が、最後の *entity-list* にありませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY ENTITY 項目を修正して、ワイルドカード *entity-list* が最後の項目になるようにしてください。

MEASFT 3246（ERR_PARSE_SPLIT_SC_ENT_WC_ERR2）

SPLIT criteria Entity '*' must be only item in last *entity-list*.

メッセージの説明

原因

SPLIT コマンドの分割条件で指定されたワイルドカード（*）が、最後の *entity-list* の唯一の項目ではありませんでした。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY ENTITY 項目を修正して、ワイルドカード *entity-list* が最後のリストにあり、ワイルドカードがそのリスト内の唯一の項目になるようにしてください。

MEASFT 3247（ERR_PARSE_NOT_ENT_KEYWORD）

XXX is not a recognized Entity.

メッセージの説明

原因

コマンドラインの文字列 *XXX* は有効なエンティティ名ではありません。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

正しいエンティティ名を入力してください。

MEASFT 3248（ERR_PARSE_SPLIT_ONE_FILE）

```
SPLIT criteria required (no wildcard) for a single output file.
```

メッセージの説明

原因

出力ファイル名が 1 つだけ指定されましたが、分割条件が指定されていません。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

指定した SPLIT コマンドに分割条件を追加してください。

MEASFT 3249（ERR_PARSE_FS_SYNTAX）

SPLIT *out-file-set* syntax error.

メッセージの説明

原因

ファイル指定でカッコを使用せずに複数の出力ファイル名が指定されました。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

SPLIT ファイル指定の構文を修正してください。

MEASFT 3250（ERR_PARSE_SPLIT_FS_WC_ERR）

SPLIT *out-file-set* filename in list cannot have an '*'.

メッセージの説明

原因

名前がカッコで囲まれている場合、SPLIT コマンドのファイル指定の項目でワイルドカードは使用できません。ワイルドカードは単独で、または出力ファイル名のプレフィックスとしてのみ使用できます。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

SPLIT ファイル指定の構文を修正してください。

MEASFT 3251（ERR_PARSE_SPLIT_BAD_CPU_NUM1）

```
SPLIT criteria CPU not numeric or out of range.
```

メッセージの説明

原因

SPLIT コマンドの分割条件で指定された CPU が、数値でないか範囲外です。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

エラー状態の CPU 番号を修正してください。

MEASFT 3252（ERR_PARSE_SPLIT_SC_BAD）

```
Syntax error after 'BY CPU/ENTITY', expected '*' or '('.
```

メッセージの説明

原因

「BY CPU」または「BY ENTITY」句の後に、ワイルドカードまたは左カッコ以外のものが存在します。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

SPLIT 分割条件の構文を修正してください。

MEASFT 3253（ERR_PARSE_SPLIT_MISSING_DEL）

```
SPLIT file specs missing a delimiter (',' or ')').
```

メッセージの説明

原因

SPLIT ファイル指定に区切り文字（カンマまたは右カッコ）がありません。

影響

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

SPLIT ファイル指定の構文を修正してください。

MEASFT 3254（ERR_PARSE_SPLIT_TIME_KW_EXP）

```
SPLIT 'FROM', 'TO', 'FOR' or INTO keyword expected.
```

メッセージの説明

原因

SPLIT BY TIME コマンドの後に、FROM、TO、FOR、または INTO キーワードが存在しなければなりません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY TIME 構文を修正してください。

MEASFT 3255（ERR_PARSE_SPLIT_BAD_FROM_TIME）

```
SPLIT FROM time value not valid.
```

メッセージの説明

原因

SPLIT BY TIME FROM *time* 値が無効です。*time* 構文は次のとおりです。

```
[ day month year, ] hr: min [ :sec ]
```

day、*month*、および *year* の各項目は、少なくとも 1 つのスペースで区切る必要があります。また、項目には有効な値が必要です。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY TIME FROM 構文を修正してください。

MEASFT 3256（ERR_PARSE_SPLIT_BAD_TO_TIME）

```
SPLIT TO time value not valid.
```

メッセージの説明

原因

SPLIT BY TIME TO *time* 値が無効です。*time* 構文は次のとおりです。

```
[ day month year, ] hr: min [ :sec ]
```

day、*month*、および *year* の各項目は、少なくとも 1 つのスペースで区切る必要があります。また、項目には有効な値が必要です。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY TIME TO 構文を修正してください。

MEASFT 3257（ERR_PARSE_SPLIT_BAD_TO_FROM_TIME）

```
SPLIT TO time value not greater than FROM time value.
```

メッセージの説明

原因

SPLIT BY TIME TO *time* 値は FROM *time* 値より大きくなければなりません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY TIME FROM または TO *time* 値を修正してください。

MEASFT 3258（ERR_PARSE_SPLIT_BAD_FOR_TIME）

```
SPLIT FOR time value not valid.
```

メッセージの説明

原因

SPLIT BY TIME FOR *duration* 値が無効です。*duration* 構文は次のとおりです。

number {SECOND[S] | MINUTES[S] | HOUR[S] }

number は 1 から 32767 までの間で指定でき、後に SECOND、MINUTE、または HOUR を続ける必要があります。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します

リカバリ

SPLIT BY TIME FOR 構文を修正してください。

MEASFT 3259（ERR_PARSE_SPLIT_BAD_INTO_OPT）

SPLIT INTO option value not valid.

メッセージの説明

原因

SPLIT BY TIME INTO 構文が無効です。INTO 構文は次のとおりです。

number [SECOND | MINUTE | HOUR][PARTS]

ここで、*number* は、後に SECOND、MINUTE、または HOUR が続く場合は 1 から 32767 を、後に値がないか PARTS が続く場合は 2 から 100 を指定できます。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY TIME INTO 構文を修正してください。

MEASFT 3260（ERR_PARSE_SPLIT_BAD_INTO_VAL）

SPLIT INTO option value range error, max PARTS is 100.

メッセージの説明

原因

PARTS の SPLIT BY TIME INTO *number* の値が範囲外です。有効な範囲は 2 から 100 です。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY TIME INTO *number* の値を修正してください。

MEASFT 3261（ERR_PARSE_SPLIT_TIME_FROM_KW_EXP）

SPLIT 'TO' or 'FOR' keyword expected.

メッセージの説明

原因

SPLIT BY TIME FROM オプションに続くキーワードが、TO または FOR ではありません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY TIME FROM 構文を修正してください。

MEASFT 3262（ERR_PARSE_SPLIT_TIME_WC）

```
SPLIT, BY TIME * command not allowed.
```

メッセージの説明

原因

SPLIT BY TIME コマンドではワイルドカードは使用できません。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

SPLIT BY TIME 構文を修正してください。

MEASFT 3300（ERR_CANNOT_SPLIT）

```
The specified measure file cannot be split by entity, try splitting by CPU.
```

メッセージの説明

原因

Measure データファイルには 1 つのエンティティのみのデータレコードが含まれているため、エンティティで分割することはできません。このエラーは分割条件が指定されていない場合のみ発生します。デフォルトでは、MEASFT はエンティティでデータファイルを分割しようとします。

原因

SPLIT コマンドは実行されません。MEASFT はエラーで終了します。

リカバリ

CPU で分割してみてください。

MEASFT 3301（ERR_CANNOT_SPLIT_BY_TIME）

```
Only H01 and later data files can be split by TIME, file version VNN.
```

メッセージの説明

原因

Measure データファイルは H01 より前のデータファイルです。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

なし。

MEASFT 4000（ERR_PROCHAND_GETMINE）

```
PROCESSHANDLE_GETMINE_ Error = #nn
```

メッセージの説明

原因

MEASFT は PROCESSHANDLE_GETMINE_でエラー *nn* が発生しました。

影響

SPLIT コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise に問い合わせてください。

MEASFT 4001（ERR_PROCHAND_DECOMP）

```
PROCESSHANDLE_DECOMPOSE_ File System Error = #nn
```

メッセージの説明

原因

MEASFT が、PROCESSHANDLE_DECOMPOSE_でファイルシステムエラー *nn* を検出しました。

影響

SPLIT コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise に問い合わせてください。

MEASFT 4002（ERR_NOMEMORY_RECBUF）

```
Not enough memory, bytes remaining = nnnn
```

メッセージの説明

原因

MEASFT はメモリ不足で、*nnnn* バイトが残っていることを報告しました。この状態は MEASFT の不具合です。

影響

SPLIT コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4003（ERR_NOMEMORY_SPLITBUF）

```
Not enough memory for output buffer, bytes remaining = nnnn
```

メッセージの説明

原因

MEASFT はメモリ不足で、*nnnn* バイトが残っていることを報告しました。この状態は MEASFT の不具合です。

影響

SPLIT コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4004（ERR_NUM_SPFILES_ERR1）

```
Internal error with variable, Num_spfiles = nn
```

メッセージの説明

原因

MEASFT で、内部変数 Num_spfiles の内部エラーが発生し、内部変数の値は *nn* であると報告されています。テキストは、エラー 4005（ERR_NUM_SPFILES_ERR2）のものと同じですが、根拠となるエラーが異なります。

影響

SPLIT コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4005（ERR_NUM_SPFILES_ERR2）

```
Internal error with variable, Num_spfiles = nn
```

メッセージの説明

原因

MEASFT で、内部変数 Num_spfiles の内部エラーが発生し、内部変数の値は *nn* であると報告されています。テキストは、エラー 4004（ERR_NUM_SPFILES_ERR1）のものと同じですが、根拠となるエラーが異なります。

影響

SPLIT コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4007（ERR_BAD_SPF_EOF）

```
Output file EOF bad, expected = nnnnnn
```

メッセージの説明

原因

MEASFT に内部エラーが発生し、出力ファイルの EOF が誤って計算されました。これは 5〜10 桁の数値である *nnnnnn* となることが予期されていました。

影響

コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4010（ERR_BAD_MF_CONTEXT）

```
Bad Measfile_context #nn
```

メッセージの説明

原因

MEASFT の内部論理エラーです。

影響

コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4012（ERR_PARSE_KW）

```
Internal error, bad keyword index = #nn
```

メッセージの説明

原因

コマンドラインの解析中の MEASFT の内部論理エラーです。この状態は MEASFT の不具合です。

影響

コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4013（ERR_DIDNT_GET_INFO_DATA）

```
Internal error, didn't get the INFO data.
```

メッセージの説明

原因

INFO または SPLIT コマンドでの MEASFT の内部論理エラーです。この状態は MEASFT の不具合です。

影響

コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4014（ERR_RB_STATE_ERROR）

```
Bad record buffer state, record #nnnnn
```

メッセージの説明

原因

レコード番号 *nnnnn* で発生した、INFO または SPLIT コマンドでの MEASFT の内部論理エラーです。この状態は MEASFT の不具合です。

影響

コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4015（ERR_BAD_JOURNAL_SELECTION）

```
Internal error, bad Journal selection = #n
```

メッセージの説明

原因

INFO または SPLIT コマンドでの MEASFT の内部論理エラーです。この状態は MEASFT の不具合です。

影響

コマンドは実行されません。MEASFT は異常終了します。

リカバリ

Hewlett Packard Enterprise の担当者に問い合わせてください。

MEASFT 4016（ERR_NOMEMORY_TIME128）

```
Not enough memory for TIME128 array, bytes remaining = #nn.
```

メッセージの説明

原因

メモリ割り当ての内部エラーです。これは MEASFT の不具合です。

影響

SPLIT コマンドは完了しません。MEASFT はエラーで終了します。

リカバリ

このメッセージを Hewlett Packard Enterprise の担当者に報告し、このメッセージに表示されている番号 *nn* を伝えてください。

MEASFT 4018（ERR_TIME128_ENTRY_NOT_FOUND）

```
TIME128 entry is not found in TIME128 array.
```

メッセージの説明

原因

エンティティレコード数が TIME128 の最大項目を超過しました。

影響

SPLIT ファイルは不完全です。MEASFT はエラーで終了します。

リカバリ

最初に CPU 分割または ENTITY 分割を行い、その分割ファイルで TIME 分割を行ってください。

MEASFT 4019（ERR_TIME128_ARRAY_FULL）

```
TIME128 array is full.
```

メッセージの説明

原因

TIME128 配列に TIME128 項目が検出されず、配列がいっぱいです。

影響

SPLIT ファイルは不完全です。MEASFT はエラーで終了します。

リカバリ

最初に CPU 分割または ENTITY 分割を行い、その分割ファイルで TIME 分割を行ってください。

新規情報と更新内容の古いエントリー

H06.25/J06.14 RVU に関する新規情報と更新内容（523324-017）

- H06.25/J06.14 以降の RVU で CID テーブルの上限が 1,024,000 CID に増加したことを示すように、START MEASSUBSYS に関する**すべての PVU および RVU での START MEASSUBSYS の使用上の注意**(150 ページ)を修正しました。
- PROCESS_CREATE_エラー情報について記述するために、**H シリーズ以降の PVU および RVU についての START MEASSUBSYS の使用上の注意**(151 ページ)を START MEASSUBSYS に追加しました。
- STATUS MEASSUBSYS に関する **STATUS MEASSUBSYS の使用上の注意**(158 ページ)でサポート対象の CID テーブルの値を改訂し、1,024K の新しい CID テーブルサイズを表示した新しい例を **STATUS MEASSUBSYS**(157 ページ)に追加しました。
- より明瞭になるように、**CPU**(194 ページ)および **PROCESS**(358 ページ)のエンティティの PAGE-SIZE-BYTES に関連した定義と使用上の注意を改訂しました。
- NATIVE-BUSY-TIME の参照を修正するために、CPU エンティティに関する **CPU-BUSY-TIME**(201 ページ)の説明を改訂しました。
- **DISC**(226 ページ)エンティティの Legacy 形式と ZMS 形式の両方のカウンターフィールド DDL 定義で C サブフィールドの説明を改訂しました。Legacy 形式のレコードでは C サブフィールドで 4 つの値をサポートし、ZMS 形式のレコードでは C サブフィールドで 8 つの値をサポートします。
- Measure G09 以降の PVU では、/G で始まる OSS ファイルのパス名の指定がサポートされていないことを示すように、**DISCOPEN**(249 ページ)、**DISKFILE**(259 ページ)、**FILE**(284 ページ)、**PROCESS**(358 ページ)、**PROCESSH**(387 ページ)、**SQLPROC**(414 ページ)、**SQLSTMT**(421 ページ)、および **USERDEF**(449 ページ)のエンティティの「パス名」または「pname」の説明を改訂しました。
- 以下のカウンターフィールドを **OSSCPU エンティティの DDL レコード（ZMS 形式）**(335 ページ)に追加し、それらの定義を **OSSCPU エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義**(339 ページ)に追加します。
  - AF-INET-SOCKET-BUF-BYTES
  - AF-INET-SOCKET-BUF-BYTES-LIMIT
  - AF-INET-SOCKET-OPENS
  - AF-INET-SOCKET-OPENS-LIMIT
  - AF-UNIX-SOCKET-BUF-BYTES
  - AF-UNIX-SOCKET-BUF-BYTES-LIMIT
  - AF-UNIX-SOCKET-OPENS
  - AF-UNIX-SOCKET-OPENS-LIMIT
  - DIR-OPENS
  - DIR-OPENS-LIMIT
  - DIR-STREAMS
  - DIR-STREAMS-LIMIT
  - DISK-CACHE-BUF-BYTES
  - DISK-CACHE-BUF-BYTES-LIMIT
  - DISK-OPENS
  - DISK-OPENS-LIMIT
  - NFS-OPENS
  - NFS-OPENS-LIMIT
  - OSS-OPENS
  - OSS-OPENS-LIMIT
  - PIPE-FIFO-BUF-BYTES

  - PIPE-FIFO-BUF-BYTES-LIMIT
  - PIPE-FIFO-OPENS
  - PIPE-FIFO-OPENS-LIMIT
  - PXS-BUF-BYTES
  - PXS-BUF-BYTES-LIMIT
  - PXS64-BUF-BYTES
  - PXS64-BUF-BYTES-LIMIT
  - TTY-OPENS
  - TTY-OPENS-LIMIT
- 以下の OSSCPU カウンター値が、OSS ファイルシステムで使用される無制限のオープンリソースのソフト制限を表しており、128,000 の OSS-OPENS-LIMIT 値（OSS SCF インターフェイスでは ALL_OPENS リソース制限とも呼ばれます）に達しない限り、これらのソフト制限は超過可能であるという注記を **OSSCPU エンティティの使用上の注意**(349 ページ)に追加しました。
  - DIR-OPENS-LIMIT
  - DIR-STREAMS-LIMIT
  - DISK-OPENS-LIMIT
  - NFS-OPENS-LIMIT
- 以下のカウンターフィールドを **PROCESS エンティティの DDL レコード（ZMS 形式）**(360 ページ)に追加し、それらの定義を **PROCESS エンティティの共通および ZMS 形式のみのカウンターフィールドの DDL 定義**(370 ページ)に追加し、その使用上の留意事項を **H シリーズ以降の PROCESS エンティティの使用上の注意**(384 ページ)に追加しました。
  - CURRENT-PRIORITY-END
  - CURRENT-PRIORITY-START
  - INITIAL-PRIORITY-END
  - INITIAL-PRIORITY-START
- PROCESS_CREATE_エラー情報を追加するように、**MEASMONCONTROL**(499 ページ)の説明を改訂しました。
- CID テーブルの上限を 1,024,000 に設定するために使用できる新しい DEFINE 文および DEFINEADD 文について説明した **MEASMONCONTROL の使用上の注意**(501 ページ)を MEASMONCONTROL に追加しました。
- CID テーブルサイズが 1,024,000 であるときの新しい設定値で **MEASMONSTATUS**(501 ページ)の説明を更新しました。
- **エンティティ記述子**(545 ページ)でのエンティティ記述子の該当する RVU 情報（必要な場合）を改訂しました。
- 時間枠に基づいてデータファイルを分割する場合にも SPLIT コマンドを使用できるようになったことを示すように、**Measure データファイルツール（MEASFT）**(672 ページ)の紹介テキストを更新しました。
- コマンドが分割した履歴レコードを表示するようになったことを示すように、**INFO コマンド**(672 ページ)の説明を更新しました。これらのレコードを表示するコマンドの例を追加しました。
- 例を含め、**SPLIT コマンド**(675 ページ)の分割条件に BY TIME オプションを追加しました。間隔データファイルに対して SPLIT BY TIME コマンドを指定するときに許容係数を含めることに関する使用上の注意を追加しました。SPLIT BY TIME の表示での時間間隔行、時間間隔列、および%列の解釈に関する使用上の注意を追加しました。TIME 分割データファイルでの MEASCOM LIST コマンドの使用に関する使用上の注意を追加しました。
- 以下のエラーメッセージを **MEASFT のエラーメッセージと警告メッセージ**(683 ページ)に追加しました。3038～3040、3254～3262、3301、4016。

H06.24/J06.13 RVU の新規情報と更新内容（523324-016）

- **LISTACTIVE entity-type**(104 ページ)、**CPU**(194 ページ)、および **PROCESS**(358 ページ)で、アクティブなカウンター測定に関する情報を改訂しました。
- 測定が終了する前に OSS ファイルが削除された場合、OSS ジャーナルに変換がキャプチャされないことがあるという情報を含めるように、**OSS ジャーナルセグメント**(182 ページ)を改訂しました。
- CPU エンティティの ZMSCPU-ID 構造体の RESERVED-1 フィールドを FLAGS フィールドに変更しました。また、この新しい **FLAGS**(200 ページ) ID フィールドの定義を追加しました。
- TNS/E システムの CPU エンティティの **CPU-BUSY-TIME**(201 ページ)カウンターの説明を改訂しました。
- 有効な IPU/CPU の数が CPU 測定中に大幅に増加した場合の Measure の動作について記述した注記を、**J シリーズの CPU エンティティの使用上の注意**(212 ページ)に追加しました。
- PROCESS エンティティの ZMSPROC-ID 構造体の RESERVED-1 フィールドを FLAGS フィールドに変更しました。また、この新しい **FLAGS**(366 ページ) ID フィールドの定義を追加しました。
- 有効な IPU/CPU の数が PROCESS 測定中に大幅に増加した場合の Measure の動作について記述した注記を、**J シリーズ PROCESS エンティティの使用上の注意**(384 ページ)に追加しました。
- エンティティ記述子の定義を、**Measure 呼び出し可能プロシージャー**(458 ページ)の **MEASCONFIGURE**(469 ページ)から新しい章である**エンティティ記述子**(545 ページ)に移しました。
- エラー 3402（ERRM^CANNOTACCESS）を引き起こす動作と推奨の対応を明確にした MEAS_READACTIVE_MANY_に関する新しい使用上の注意を、**使用上の注意**(523 ページ)に追加しました。

H06.23/J06.12 RVU に関する新規情報と更新内容（523324-015）

- Measure 製品のバージョンを `H06` と `J04` に更新しました。

  Measure H06/J04 PVU は当初、H06.22/J06.11 RVU でリリースされましたが、RVU には Measure リファレンスマニュアルは不要でした。

CLIM 接続デバイスのパーティション番号のサポート

- CLIM 接続デバイスの新しいパーティションパラメーターのサポートを示すように、すべての使用上の注意を改訂し、**MEASCOM コマンド**(36 ページ)で ADD、INFO、INFO MEASUREMENT、LIST、LIST PLOT、および LISTALL コマンドのいくつかの例を示しました。
- *partition* パラメーターのサポートと、*sac* パラメーターでの CLIM 接続デバイスに対する複数の名前のサポートにおける違いを示すように、**LISTACTIVE エンティティ指定の特殊なケース**(108 ページ)、**DEVICE**(108 ページ)、および **DISC**(109 ページ)を改訂しました。DEVICE と DISC がそれぞれ別のサブセクションを持つようになりました。DISC エンティティはこれらの新しいパーティション関連のパラメーターの変更をサポートしますが、DEVICE エンティティはサポートしません。また、DEVICE エンティティは *path* パラメーターをサポートしなくなったことに注意してください。
- *path* パラメーターがサポートされなくなったことを示すように、**DEVICE**(214 ページ)エンティティの構文と使用上の注意を改訂しました。
- **DISC**(226 ページ)

  エンティティ構文と使用上の注意を、*partition* パラメーターのサポートと、*sac* パラメーターでの CLIM 接続デバイスに対する複数の名前のサポートを示すように改訂しました。
- 新しい *partition* フィールドと新しい PLPT フラグ、MEAS_MULTIPLE_CLIMS を示すように、**DEVICE^CLIM^DESC**(550 ページ)を改訂しました。
- パーティションパラメーターを参照するように、**エラーメッセージ**(597 ページ)内のエラー 3122 のテキストを改訂しました。
- 3124、3125、および 3126 のエラーメッセージを**エラーメッセージ**(597 ページ)に追加しました。
- パーティションパラメーターを参照するように、**エラーコード**(627 ページ)内のエラーコード 3402 のテキストを改訂しました。

ロックページカウンターのサポート

- 新しい LOCKED-PAGES-END および LOCKED-PAGES-START カウンターが **CPU**(194 ページ)エンティティの説明に追加されました。変更内容は、ZMS 形式の DDL レコード、フィールド定義、および使用上の注意の更新です。
- 新しい LOCKED-PAGES-END および LOCKED-PAGES-START カウンターが **PROCESS**(358 ページ)エンティティの説明に追加されました。変更内容は、ZMS 形式の DDL レコード、フィールド定義、および使用上の注意の更新です。

その他

- 新しいサブセクションの**測定の開始と停止の際の考慮事項**(33 ページ)を **Measure の概要**(29 ページ)に追加しました。
- **MEASCOM コマンド**(36 ページ)でのさまざまな構文および構造上の不整合を修正しました。
- ZMS 形式の DDL レコードを強調し、フィールド定義の検索を機能強化するように、**エンティティおよびカウンター**(167 ページ)の構造を再編成しました。その他の構文、定義、およびフォーマットの問題を修正しました。
- CPU エンティティについて **PROCESS-OVHD**(205 ページ)の定義を改訂しました。
- 指定した名前が Guardian 名であることを示すように、**OSSNS**(350 ページ)エンティティの説明で **process-name** の定義を更新しました。
- H シリーズ以降の RVU で ALLTIME を指定する場合には、システムライブラリである *code-space* を指定する必要があることを示すように、**PROCESSH エンティティのエンティティ指定構文**(388 ページ)の ALLTIME 定義を改訂しました。
- 測定が SQL アプリケーションの実行全体に及ぶことを確認するように勧める推奨を**すべての SQLSTMT エンティティの使用上の注意**(434 ページ)に追加しました。
- EXTDECS と **Measure 呼び出し可能プロシージャー**(458 ページ)の内容とのさまざまな不整合を修正しました。
- **Measure 呼び出し可能プロシージャー**(458 ページ)のその他のプロシージャーに、エラーコードを追加しました。
- 欠落している記述子テンプレート構造体を **MEASCONFIGURE**(469 ページ)に追加し、目次で見つけられるように記述子名を小見出しにしました。
- **MEASCONFIGURE**(469 ページ)から OSSNS^DESC の説明を削除しました。
- OSSNS エンティティが PROCESS^DESC を使用することを示すように、**エンティティおよびカウンター**の OSSNS エントリーを改訂しました。
- OSSCPU エンティティのサポートを参照するように **CPU^DESC**(547 ページ)を改訂しました。
- OSSNS エンティティのサポートを示すように **PROCESS^DESC**(571 ページ)を改訂しました。
- 欠落している *ip^addr*、*port*、および *ip^wildcard^flags* フィールドを **MEASREAD_DIFF_**(509 ページ)と **MEASWRITE_DIFF_**(538 ページ)に追加しました。