Hewlett Packard
Enterprise

2020 年 9 月 27 日

PostgreSQL 13 新機能検証結果（GA）

日本ヒューレット・パッカード株式会社
篠田典良

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

Hewlett Packard
Enterprise

目次

目次 2
1. 本文書について 5
1.1 本文書の概要 5
1.2 本文書の対象読者 5
1.3 本文書の範囲 5
1.4 本文書の対応バージョン 5
1.5 本文書に対する質問・意見および責任 6
1.6 表記 6
2. PostgreSQL 13 における変更点概要 7
2.1. 大規模環境に対応する新機能 7
2.2. 信頼性向上に関する新機能 7
2.3. 運用性を向上させる新機能 8
2.4. 将来の新機能に対する準備 8
2.5. 非互換 10
2.5.1. configure 10
2.5.2. createuser 10
2.5.3. CSV ログ・フォーマット 10
2.5.4. Extension 10
2.5.5. opaque 11
2.5.6. OpenSSL の必要バージョン 11
2.5.7. pg_regress 11
2.5.8. psql のデフォルト・プロンプト 11
2.5.9. to_date/to_timestamp 12
2.5.10. リカバリー中のプロモーション 12
2.5.11. パーティション・キー 12
2.5.12. 待機イベント 12
2.5.13. SIMILAR TO ESCAPE 13
3. 新機能解説 14
3.1. アーキテクチャの変更 14
3.1.1. システムカタログの変更 14
3.1.2. データ型 19
3.1.3. ディスク・ベースのハッシュ集約 20
3.1.4. インクリメンタル・ソート 21
3.1.5. バックアップ・マニフェスト 22

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.1.6. パーティション・テーブル 24
3.1.7. 自動 VACUUM のログ 27
3.1.8. 待機イベント 27
3.1.9. libpq 接続文字列 29
3.1.10. libpq 関数 30
3.1.11. Hook 30
3.1.12. 列トリガー 30
3.1.13. ローカル接続のキー 30
3.1.14. Trusted Extension 31
3.1.15. レプリケーション・スロット 33
3.1.16. テキスト検索 33
3.2. SQL 文の拡張 35
3.2.1. ALTER NO DEPENDS ON 35
3.2.2. ALTER STATISTICS SET STATISTICS 35
3.2.3. ALTER TABLE 36
3.2.4. ALTER TYPE 38
3.2.5. ALTER VIEW 39
3.2.6. CREATE DATABASE 40
3.2.7. CREATE INDEX 41
3.2.8. CREATE TABLE 42
3.2.9. CREATE TABLESPACE 42
3.2.10. DROP DATABASE FORCE 43
3.2.11. EXPLAIN ANALYZE 43
3.2.12. INSERT 47
3.2.13. JSON 48
3.2.14. MAX/MIN pg_lsn 48
3.2.15. ROW 48
3.2.16. SELECT FETCH FIRST WITH TIES 49
3.2.17. VACUUM PARALLEL 49
3.2.18. オペレーター<-> 50
3.2.19. 関数 50
3.3. パラメーターの変更 55
3.3.1. 追加されたパラメーター 55
3.3.2. 変更されたパラメーター 57
3.3.3. デフォルト値が変更されたパラメーター 58
3.4. ユーティリティの変更 59

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.4.1. dropdb 59
3.4.2. pg_basebackup 59
3.4.3. pg_dump 60
3.4.4. pg_rewind 62
3.4.5. pg_verifybackup 63
3.4.6. pg_waldump 65
3.4.7. psql 66
3.4.8. reindexdb 68
3.4.9. vacuumdb 69
3.4.10. その他 69
3.5. Contrib モジュール 70
3.5.1. adminpack 70
3.5.2. auto_explain 70
3.5.3. dict_int 71
3.5.4. ltree 72
3.5.5. pageinspect 72
3.5.6. pg_stat_statements 74
3.5.7. postgres_fdw 76
3.5.8. bool_plperl 77
参考にした URL 78
変更履歴 79

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

Hewlett Packard
Enterprise

1. 本文書について

1.1 本文書の概要

本文書は GA 版が公開されているオープンソース RDBMS である PostgreSQL 13 (13.0) の主な新機能について検証した文書です。

1.2 本文書の対象読者

本文書は、既にある程度 PostgreSQL に関する知識を持っているエンジニア向けに記述しています。インストール、基本的な管理等は実施できることを前提としています。

1.3 本文書の範囲

本文書は PostgreSQL 12 (12.4) と PostgreSQL 13 (13.0)の主な差分を記載しています。原則として利用者が見て変化がわかる機能について調査しています。すべての新機能について記載および検証しているわけではありません。特に以下の新機能は含みません。

- バグ解消
- 内部動作の変更によるパフォーマンス向上
- レグレッション・テストの改善
- psql コマンドのタブ入力による操作性改善
- pgbench コマンドの改善
- ドキュメントの改善、ソース内の Typo 修正
- 動作に変更がないリファクタリング

1.4 本文書の対応バージョン

本文書は以下のバージョンとプラットフォームを対象として検証を行っています。

表 1 対象バージョン

種別	バージョン
データベース製品	PostgreSQL 12.4 (比較対象) PostgreSQL 13 (13.0) GA (2020/9/21 20:51:14)
オペレーティング・システム	Red Hat Enterprise Linux 7 Update 5 (x86-64)
Configure オプション	--with-llvm --with-openssl --with-perl --with-python

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

Hewlett Packard
Enterprise

1.5 本文書に対する質問・意見および責任

本文書の内容は日本ヒューレット・パッカード株式会社の公式見解ではありません。また内容の間違いにより生じた問題について作成者および所属企業は責任を負いません。本文書で検証した仕様が変更される場合があります。本文書に対するご意見等ありましたら作成者 篠田典良（Mail: noriyoshi.shinoda@hpe.com）までお知らせください。

1.6 表記

本文書内にはコマンドや SQL 文の実行例および構文の説明が含まれます。実行例は以下のルールで記載しています。

表 2 例の表記ルール

表記	説明
#	Linux root ユーザーのプロンプト
$	Linux 一般ユーザーのプロンプト
太字	ユーザーが入力する文字列
postgres=#	PostgreSQL 管理者が利用する psql コマンド・プロンプト
postgres=>	PostgreSQL 一般ユーザーが利用する psql コマンド・プロンプト
下線部	特に注目すべき項目
<<以下省略>>	より多くの情報が出力されるが文書内では省略していることを示す
<<途中省略>>	より多くの情報が出力されるが文書内では省略していることを示す
<<パスワード>>	パスワードの入力を示す

構文は以下のルールで記載しています。

表 3 構文の表記ルール

表記	説明
斜体	ユーザーが利用するオブジェクトの名前やその他の構文に置換
[ ]	省略できる構文であることを示す
{ A | B }	A または B を選択できることを示す
…	旧バージョンと同一である一般的な構文

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

2. PostgreSQL 13 における変更点概要

PostgreSQL 13 には 200 以上の新機能が追加されました。代表的な新機能と利点について説明します。

2.1. 大規模環境に対応する新機能

大規模環境に適用できる以下の機能が追加されました。

□ パラレル処理の拡張

テーブルに複数のインデックスが作成されている場合、VACUUM 処理が並列に行われるようになりました。

□ B-Tree インデックスの重複排除

B-Tree インデックスでは重複排除処理が標準で動作するようになりました。ストレージ容量と I/O を削減することができます。

□ WAL 出力の抑制

wal_level パラメーターが minimal に設定されている場合に、WAL 出力が抑制される構文が増えました。

2.2. 信頼性向上に関する新機能

PostgreSQL 13 では信頼性を向上させるために以下の拡張が実装されました。

□ バックアップの信頼性向上

バックアップの整合性をチェックできるようになりました。pg_basebackup コマンド等によるベース・バックアップにはファイル単位のサイズとチェックサムが取得され、バックアップ先に保存されます。取得したバックアップの整合性は pg_verifybackup コマンドでチェックすることができます。

□ ストリーミング・レプリケーションの動的構成変更

ストリーミング・レプリケーションのスタンバイ・インスタンスで使用する各種パラメーターは動的に変更できるようになりました。この新機能により、プライマリ・インスタンス障害時にスタンバイ・インスタンスを再起動する必要がなくなりました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

2.3. 運用性を向上させる新機能

運用性を向上できる以下の機能が追加されました。

□ ディスク・ベースのハッシュ集約

ハッシュ・テーブルを操作する処理はメモリー使用量の上限を超えるとストレージ上で行われるようになりました。従来はメモリー使用量に上限が無かったため、OOM Killer の対象になることがありました。

□ パーティション・テーブルの論理レプリケーション

パーティション・テーブルの論理レプリケーションが実現できるようになりました。

□ モニタリング機能の強化

ANALYZE 文や pg_basebackup コマンドの実行状況をリアルタイムに確認できるカタログが追加されました。またキャッシュのヒット率や I/O の状況を確認できるようになりました。

□ 実行計画作成と WAL 出力量のトラッキング

実行計画作成時や SQL 文実行時の WAL 出力量がトラッキングできるようになりました。pg_stat_statements モジュールや EXPLAIN 文で確認できます。

□ 待機イベントの追加

監視できる待機イベントがいくつか追加されました。pg_stat_activity カタログで確認できます。

2.4. 将来の新機能に対する準備

PostgreSQL 13 では将来のバージョンで提供される機能の準備が進みました。

□ 64 ビット・トランザクション ID 対応

64 ビット・トランザクション ID を示す xid8 データ型と、このデータ型を扱う各種関数群が追加されました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

□　並列実行の拡張

COPY、INSERT、VACUUM 等のパラレル化を実装できるようにするため、パラレル・ワーカー・プロセス間でページロックとリレーション拡張ロックを競合させて排他的に処理するようになりました。

□　PUBLICATION の拡張

テーブル以外のオブジェクトを PUBLICATION に追加できる準備が行われました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

2.5. 非互換

PostgreSQL 13 は PostgreSQL 12 から以下の仕様が変更されました。

2.5.1. configure

configure コマンドから--disable-float4-byval オプションが削除されました。また libxml2 ライブラリの検索に pkg-config コマンドを利用するようになりました。

2.5.2. createuser

createuser コマンドから--adduser オプション、--no-adduser オプションが削除されました。

2.5.3. CSV ログ・フォーマット

CSV 形式のログ（log_destination='csvlog'）の末尾にバックエンド・タイプが追記されるようになりました。

例 1 CSV ログ（抜粋）

```
2020-09-24 23:27:12.495 JST,,,30744,,5ec6194a.7818,1,,2020-05-21 15:01:46
JST,,0,LOG,00000,"database system was shut down at 2020-05-21 15:01:29
JST",,,,,,,,,"","startup"

2020-09-24 23:27:12.495 JST,,,30734,,5ec61949.780e,6,,2020-05-21 15:01:45
JST,,0,LOG,00000,"database system is ready to accept
connections",,,,,,,,,"","postmaster"

2020-09-24 23:27:12.495
JST,"postgres","postgres",30753,"[local]",5ec61953.7821,1,"SELECT",2020-05-21
15:01:55 JST,3/2,0,ERROR,42P01,"relation ""notexists"" does not
exist",,,,,,"SELECT * FROM notexists;",15,,"psql","client backend"
```

2.5.4. Extension

Extension における"unpackaged" の指定は廃止されました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

2.5.5. opaque

疑似データ型 opaque は削除されました。このデータ型は PostgreSQL 7.3 以前からの互換性のために使用されていました。

2.5.6. OpenSSL の必要バージョン

OpenSSL 1.0.0 以下はサポート対象外になりました。OpenSSL 1.0.1 以上が必要です。この修正に伴い、パラメーターssl_min_protocol_version の最小値が TLSv1 から TLSv1.2 に変更されました。

2.5.7. pg_regress

pg_regress コマンドから--load-language オプションが削除されました。

2.5.8. psql のデフォルト・プロンプト

psql コマンドのデフォルト・プロンプト設定である PROMPT1 と PROMPT2 変数に、トランザクションの状態を示す%x が含まれるようになりました。

表 4 デフォルト設定の変更

変数名	PostgreSQL 12	PostgreSQL 13	備考
PROMPT1	"%/%R%# "	"%/%R%x%# "	
PROMPT2	"%/%R%# "	"%/%R%x%# "	
PROMPT3	">> "	">> "	変更無し

例 2 デフォルト・プロンプト

```
postgres=> BEGIN ;
BEGIN
postgres=*> -- In the active transaction
postgres=*> ERROR ;
ERROR: syntax error at or near "ERROR"
LINE 1: ERROR
        ^
postgres=!> -- In the failed transaction
postgres=!> ROLLBACK ;
ROLLBACK
postgres=>
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

2.5.9. to_date/to_timestamp

フォーマット文字列 TM はロケールに応じて出力内容が変化するようになりました。従来は無視されていました。

例 3 to_date 関数の実行

```
postgres=> SET lc_time='ru_RU' ;
SET
postgres=> SELECT to_date('01 фев 2020', 'DD TMMON YYYY') ;
  to_date
------------
 2020-02-01
(1 row)
```

2.5.10. リカバリー中のプロモーション

従来はリカバリーの一時停止中にプロモーションが行われた場合、一時停止状態が継続していました。PostgreSQL 13 ではリカバリーの一時停止中にプロモーションが行われた場合にはプロモーションを優先します。

2.5.11. パーティション・キー

抽象型を返す関数はパーティション・キーの値として使用できなくなりました。この仕様は PostgreSQL 12.2 以前のバージョンにバックポートされました。

例 4 抽象型を返すパーティションキー値

```
postgres=> CREATE TABLE part1 (c1 INT, c2 INT)
        PARTITION BY RANGE(((c1, c2))) ;
ERROR:  partition key column 1 has pseudo-type record
```

2.5.12. 待機イベント

待機イベントの名前が大幅に変更されました。pg_stat_activity カタログの event_name 列や、pg_locks カタログの locktype 列の出力文字列が変更されました[1]。

[1] 「3.1.8 待機イベント」参照

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

2.5.13. SIMILAR TO ESCAPE

SIMILAR TO … ESCAPE NULL 句は NULL を返します。従来のバージョンでは ESCAPE NULL 句は無視されていました。

例 5 PostgreSQL 12 の動作

```
postgres=> SELECT 'ABC' SIMILAR TO 'ABC' ESCAPE NULL ;
 ?column?
----------
 t
(1 row)
```

例 6 PostgreSQL 13 の動作

```
postgres=> ¥pset null null
Null display is "null".
postgres=> SELECT 'ABC' SIMILAR TO 'ABC' ESCAPE NULL ;
 ?column?
----------
 null
(1 row)
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

Hewlett Packard
Enterprise

3. 新機能解説

3.1. アーキテクチャの変更

3.1.1. システムカタログの変更

以下のシステムカタログが変更されました。

表 5 追加されたシステムカタログ

カタログ名	説明
pg_shmem_allocations	共有メモリーの内訳を参照できます。
pg_stat_progress_analyze	ANALYZE 文の実行状況を参照できます。
pg_stat_progress_basebackup	ベースバックアップの実行状況を参照できます。
pg_stat_slru	SLRU キャッシュの統計情報を参照できます。

表 6 削除されたシステムカタログ

カタログ名	説明
pg_pltemplate	プロシージャ言語のテンプレート

表 7 情報スキーマ（information_schema）内の削除されたビュー

カタログ名	説明
sql_languages	SQL 文の標準準拠レベル、オプション、方言
sql_packages	標準パッケージの一覧
sql_sizing_profiles	定義されたサイズ情報の一覧

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

表 8 列が追加されたシステムカタログ

カタログ名	追加列名	データ型	説明
pg_available_extension_versions	trusted	boolean	Extension が一般ユーザー登録できるか
pg_publication	pubviaroot	boolean	パーティションの更新をルート・テーブルに変換するか
pg_replication_slots	wal_status	text	WAL ファイルの状態
	safe_wal_size	bigint	危険無く書き込める WAL バイト数
pg_stat_activity	leader_pid	integer	パラレル・クエリーのリーダーPID
pg_stat_{all|sys|user}_tables	n_ins_since_vacuum	bigint	前回の VACUUM から INSERT されたタプル数
pg_stat_wal_receiver	written_lsn	pg_lsn	書き込まれたがまだフラッシュされていない LSN
	flushed_lsn	pg_lsn	フラッシュされた LSN
pg_statistic_ext	stxstattarget	integer	SET STATISTIC value
pg_trigger	tgparentid	oid	親トリガーの OID

表 9 列が削除されたシステムカタログ

カタログ名	削除列名	説明
pg_stat_wal_receiver	received_lsn	written_lsn と flushed_lsn に分割されました。

表 10 出力内容が変更されたシステムカタログ

カタログ名	説明
pg_locks	locktype 列に出力される値 speculative token が spectoken に変更されました。
pg_stat_ssl	クライアント接続以外のプロセス情報が削除されました。
pg_stat_gssapi	クライアント接続以外のプロセス情報が削除されました。

変更されたシステムカタログから、主なカタログの詳細を以下に記載します。

□ pg_shmem_allocations カタログ

pg_shmem_allocations カタログは共有メモリー内の内訳を確認することができます。ただしダイナミック共有メモリーの領域は含みません。このカタログを参照できるのは SUPERUSER 属性を持つユーザーだけです。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

表 11 pg_shmem_allocations カタログ

列名	データ型	説明
name	text	メモリー領域の名前、null は未使用領域
off	bigint	開始位置からのオフセット
size	bigint	確保量
allocated_size	bigint	パディングを含む確保量

例 7 pg_shmem_allocations カタログの検索

```
postgres=# SELECT * FROM pg_shmem_allocations ;
                name                |    off    |   size    | allocated_size
------------------------------------+-----------+-----------+----------------
 Buffer IO Locks                    | 140660096 |    524288 |         524288
 Buffer Descriptors                 |   5393792 |   1048576 |        1048576
 Backend SSL Status Buffer          | 146585088 |     42312 |          42368
 Async Queue Control                | 147119360 |      2492 |           2560
 Wal Sender Ctl                     | 147112832 |      1280 |           1280
 AutoVacuum Data                    | 147104384 |      5368 |           5376
 PROCLOCK hash                      | 143136000 |      2904 |           2944
 FinishedSerializableTransactions   | 146097664 |        16 |            128
 XLOG Ctl                           |     53504 |   4208272 |        4208384
 Shared MultiXact State             |   5392640 |      1028 |           1152
<<<以下省略>>>
```

□ pg_stat_progress_analyze カタログ

pg_stat_progress_analyze カタログは ANALYZE 文の実行状況を確認することができます。このカタログは一般ユーザーでも参照できますが、一般ユーザーでは他のユーザーのコマンド実行状況は確認できません。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

表 12 pg_stat_progress_analyze カタログ

列名	データ型	説明
pid	integer	バックエンドのプロセス ID
datid	oid	接続先データベースの OID
datname	name	接続先データベース名
relid	oid	ANALYZE 文を実行しているテーブル OID
phase	text	実行フェーズ
sample_blks_total	bigint	サンプリングされる総ブロック数
sample_blks_scanned	bigint	サンプリング済のブロック数
ext_stats_total	bigint	拡張統計数
ext_stats_computed	bigint	計算された拡張統計数
child_tables_total	bigint	子テーブル数
child_tables_done	bigint	ANALYZE 実行済の子テーブル数
current_child_table_relid	oid	ANALYZE 実行中の子テーブル OID

例 8 pg_stat_progress_analyze カタログの検索

```
postgres=# SELECT * FROM pg_stat_progress_analyze ;
-[ RECORD 1 ]---------+--------------------
pid                   | 30932
datid                 | 13578
datname               | postgres
relid                 | 16388
phase                 | computing statistics
sample_blks_total     | 54055
sample_blks_scanned   | 54055
ext_stats_total       | 0
ext_stats_computed    | 0
child_tables_total    | 0
child_tables_done     | 0
current_child_table_relid | 0
```

□ pg_stat_progress_basebackup カタログ

pg_stat_progress_basebackup カタログには pg_basebackup コマンドのようなツールによるバックアップ中の実行状況を確認することができます。このカタログは一般ユーザーでも参照できますが、接続ユーザー以外では pid 列のみ参照できます。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

表 13 pg_stat_progress_basebackup カタログ

列名	データ型	説明
pid	integer	WAL sender のプロセス ID
phase	text	実行フェーズを示す文字列
backup_total	bigint	総バックアップ量
backup_streamed	bigint	ストリーミング・データ量
tablespaces_total	bigint	総テーブルスペース数
tablespaces_streamed	bigint	ストリーミング・テーブルスペース数

例 9 pg_stat_progress_basebackup カタログの検索

```
postgres=# SELECT * FROM pg_stat_progress_basebackup ;
-[ RECORD 1 ]--------+-------------------------
pid                  | 31080
phase                | streaming database files
backup_total         | 467913216
backup_streamed      | 302026752
tablespaces_total    | 1
tablespaces_streamed | 0
```

□ pg_stat_slru カタログ

pg_stat_slru カタログは SLRU キャッシュの利用状況を確認できます。SLRU キャッシュ領域ごとにキャッシュされたページへのアクセスに関する統計が表示されます。

表 14 pg_stat_slru カタログ

列名	データ型	説明
name	text	SLRU 名
blks_zeroed	bigint	ゼロに初期化されたブロック数
blks_hit	bigint	キャッシュにヒットしたブロック数
blks_read	bigint	読み込みを行ったブロック数
blks_written	bigint	書き込みを行ったブロック数
blks_exists	bigint	キャッシュに存在をチェックされたブロック数
flushes	bigint	フラッシュされたダーティ・ブロック数
truncates	bigint	トランケートされたブロック数
stats_reset	timestamp with time zone	統計情報がリセットされた日時

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

カウンターの値はインスタンスが再起動しても維持されます。カウンター値をリセットするためには pg_stat_reset_slru 関数[2]を実行します。

例 10 pg_stat_slru カタログの検索

```
postgres=# SELECT name, blks_hit, blks_read, blks_written FROM pg_stat_slru ;
      name       | blks_hit | blks_read | blks_written
-----------------+----------+-----------+--------------
 CommitTs        |        0 |         0 |            0
 MultiXactMember |        0 |         0 |            0
 MultiXactOffset |        0 |         0 |            0
 Notify          |        0 |         0 |            0
 Serial          |        0 |         0 |            0
 Subtrans        |        0 |         0 |            0
 Xact            |       14 |         0 |            0
 other           |        0 |         0 |            0
(8 rows)
```

3.1.2. データ型

以下のデータ型が追加されています。

□ regcollation

Collation 名を示す regcollation 型が追加されました。同時に text 型から regcollation 型に変換する to_regcollation 関数が追加されました。

例 11 to_regcollation 関数の実行

```
postgres=> SELECT to_regcollation('"POSIX"') ;
 to_regcollation
-----------------
 "POSIX"
(1 row)
```

[2] 「3.2.1.9 関数」参照

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

□ xid8

64 ビットのトランザクション ID を示す xid8 型が追加されました。xid8 型を利用する関数群が追加されています。従来の 32 ビット長トランザクション ID を返す関数は互換性の維持のために残されていますが、将来は削除される可能性があります。

表 15 64 ビット・トランザクション ID を利用する関数

旧関数名	新関数名
txid_current	pg_current_xact_id
txid_current_if_assigned	pg_current_xact_id_if_assigned
txid_current_snapshot	pg_current_snapshot
txid_snapshot_xip	pg_snapshot_xip
txid_snapshot_xmax	pg_snapshot_xmax
txid_snapshot_xmin	pg_snapshot_xmin
txid_visible_in_snapshot	pg_visible_in_snapshot
txid_status	pg_xact_status

□ pg_snapshot

pg_snapshot 型は特定の時点でのトランザクション ID の可視性（xmin, xmax, xip_list）に関する情報を格納します。

□ その他の抽象データ型

その他に以下の抽象データ型が追加されました。

表 16 追加された抽象データ型

データ型	説明
anycompatible	関数が任意のデータ型を受入れ、自動的に昇格
anycompatiblearray	関数が任意の配列型を受入れ、自動的に昇格
anycompatiblenonarray	関数が任意の配列以外の型を受入れ、自動的に昇格
anycompatiblerange	関数が任意の範囲型を受入れ、自動的に昇格
table_am_handler	テーブル・アクセス・メソッドのハンドラー

3.1.3. ディスク・ベースのハッシュ集約

GROUP BY 句や DISTINCT 句を実行する際にメモリー中にハッシュ・テーブルを作成する場合があります。PostgreSQL 13 ではハッシュ・テーブルがワーク・メモリーに格納できない場合、ストレージ・ベースのハッシュ集約（Hash Aggregation）を実行します。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

従来のバージョンではメモリーに格納できる前提でワーク・メモリーを作成していたため、予期せぬメモリー使用量の増加が発生する可能性がありました。

例 12 ディスク・ベースのハッシュ集約

```
postgres=> EXPLAIN ANALYZE SELECT COUNT(*) FROM data1 GROUP BY c1 ;
                                QUERY PLAN
-------------------------------------------------------------------------------
 HashAggregate  (cost=4304055.00..5029055.00 rows=10000000 width=14) (actual
time=2703.993..16089.774 rows=10000000 loops=1)
    Group Key: c1
    Planned Partitions: 4  Batches: 87381  Memory Usage: 109kB  Disk Usage:
268256kB
    ->  Seq Scan on data1  (cost=0.00..154055.00 rows=10000000 width=6) (actual
time=0.025..606.455 rows=10000000 loops=1)
  Planning Time: 0.039 ms
  Execution Time: 16407.060 ms
 (6 rows)
```

3.1.4. インクリメンタル・ソート

インクリメンタル・ソートは複数列のソートに使用するリソースを削減する方法です。すでにソートされている列に加えて別の列のソートが必要になった場合に使われます。この機能は enable_incremental_sort パラメーターにより制御できます。デフォルト値は on です。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 13 インクリメンタル・ソート

```
postgres=> EXPLAIN ANALYZE SELECT * FROM (SELECT * FROM data1 ORDER BY c1) t
ORDER BY c1, c2;
                                    QUERY PLAN
-----------------------------------------------------------------------------
Incremental  Sort    (cost=0.49..853808.44  rows=10000000  width=12)  (actual
time=0.063..2253.672 rows=10000000 loops=1)
   Sort Key: data1.c1, data1.c2
   Presorted Key: data1.c1
   Full-sort Groups: 312500  Sort Method: quicksort  Average Memory: 26kB  Peak
Memory: 26kB
   ->   Index  Only  Scan  using  idx1_data1  on  data1   (cost=0.43..303808.43
rows=10000000 width=12) (actual time=0.045..891.341 rows=10000000 loo
ps=1)
         Heap Fetches: 0
 Planning Time: 0.258 ms
 Execution Time: 2495.751 ms
(8 rows)
```

3.1.5. バックアップ・マニフェスト

pg_basebackup コマンド等で作成されるベースバックアップには整合性のチェックを行うことができるようになりました。データベース・クラスター内の各ファイル単位にサイズとチェックサムが計算されます。

□ チェックサム

チェックサムの計算方法は SHA224、SHA256、SHA384、SHA512、CRC32C から選択できます。CRC32C がデフォルト値です。

□ マニフェスト・ファイル

バックアップ・マニフェストはバックアップ先ディレクトリ内に backup_manifest として保存されます。JSON フォーマットのファイルです。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 14 backup_manifest ファイル

```
$ pg_basebackup -D back
$ cat back/backup_manifest
{ "PostgreSQL-Backup-Manifest-Version": 1,
"Files": [
{ "Path": "backup_label", "Size": 226, "Last-Modified": "2020-09-24 06:20:15
GMT", "Checksum-Algorithm": "CRC32C", "Checksum": "0e45028a" },
{ "Path": "global/1262", "Size": 8192, "Last-Modified": "2020-09-24 05:50:10
GMT", "Checksum-Algorithm": "CRC32C", "Checksum": "6a2131f8" },
<<途中省略>>
"WAL-Ranges": [
{ "Timeline": 1, "Start-LSN": "0/2D000028", "End-LSN": "0/2D0001E8" }
],
"Manifest-Checksum":
"8832de70e00fd7612125364088a945ff5cacca3befb4004eea255e2b498a0ca6"}
```

tar 形式のバックアップを行った場合、マニフェスト・ファイルは tar ファイルには含まれません。

例 15 tar ファイル形式のバックアップにおけるマニフェスト・ファイル

```
$ pg_basebackup --format=tar -D back
$ ls back
backup_manifest  base.tar  pg_wal.tar
$
```

□ 整合性の確認

取得したバックアップの整合性を再検査するために pg_verifybackup コマンドが提供されています。pg_verifybackup コマンドにはバックアップ先ディレクトリ名を指定します。[3]

[3] 「3.4.6 pg_verifybackup」参照

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 16 pg_verifybackup コマンドの実行

```
$ pg_verifybackup back
backup successfully verified
```

3.1.6. パーティション・テーブル

パーティション・テーブルには以下の機能が追加されました。

□ ロジカル・レプリケーション対応

ロジカル・レプリケーション環境の PUBLICATION にパーティション・テーブルを追加できるようになりました。旧バージョンでは下記の例の CREATE PUBLICATION 文は失敗していました。レプリケーションを成功させるには SUBSCRIPTION 側にも同一構造のテーブルが必要です。

例 17 パーティション・テーブルを PUBLICATION に追加

```
postgres=> CREATE TABLE part1(c1 NUMERIC PRIMARY KEY, c2 VARCHAR(10)) PARTITION
        BY RANGE(c1) ;
CREATE TABLE
postgres=> CREATE TABLE part1v1 PARTITION OF part1 FOR VALUES FROM (0)
        TO (100000) ;
CREATE TABLE
postgres=> CREATE TABLE part1v2 PARTITION OF part1 FOR VALUES FROM (100000)
        TO (200000) ;
CREATE TABLE
postgres=> CREATE PUBLICATION pub1 FOR TABLE part1 ;
CREATE PUBLICATION
postgres=> ¥dRp+ pub1
                            Publication pub1
 Owner | All tables | Inserts | Updates | Deletes | Truncates | Via root
-------+------------+---------+---------+---------+-----------+----------
 demo  | f          | t       | t       | t       | t         | f
Tables:
    "public.part1"
```

デフォルトでは PUBLICATION はパーティションに対する更新情報をパーティション・テーブルへの更新に変換せずに SUBSCRIPTION に送信します。SUBSCRIPTION 側で非パーティション・テーブルに対してレプリケーションを行う場合は PUBLICATION の属性 publish_via_partition_root を off に設定します。

例 18 SUBSCRIPTION 側では通常のテーブル

```
postgres=> ALTER PUBLICATION pub1 SET (publish_via_partition_root = on) ;
ALTER PUBLICATION
postgres=> ¥dRp+ pub1
                            Publication pub1
 Owner | All tables | Inserts | Updates | Deletes | Truncates | Via root
-------+------------+---------+---------+---------+-----------+----------
 demo  | f          | t       | t       | t       | t         | t
Tables:
    "public.part1"

# SUBSCRIPTION 側
postgres=> CREATE TABLE part1(c1 NUMERIC PRIMARY KEY, c2 VARCHAR(10)) ;
CREATE TABLE
postgres=# CREATE SUBSCRIPTION sub1 CONNECTION 'host=remhost1 dbname=postgres
user=postgres password=<<パスワード>>' PUBLICATION pub1 ;
NOTICE:  created replication slot "sub1" on publisher
CREATE SUBSCRIPTION
```

□ BEFORE INSERT トリガー

ROW レベルの BEFORE INSERT トリガーを設定できるようになりました。ただし、タプルが格納されるパーティションを変更することはできません。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 19 パーティション・テーブルに対する CREATE TRIGGER 文

```
postgres=> CREATE TABLE part1 (c1 INT, c2 INT, c3 VARCHAR(10))
               PARTITION BY LIST(c1) ;
CREATE TABLE
postgres=> CREATE TABLE part1v1 PARTITION OF part1 FOR VALUES IN (10) ;
CREATE TABLE
postgres=> CREATE OR REPLACE FUNCTION fnc1_part1() RETURNS TRIGGER
           LANGUAGE plpgsql AS $$
           BEGIN
               NEW.c3 = 'TRIGGER';
               RETURN NEW;
           END; $$ ;
CREATE FUNCTION
postgres=> CREATE TRIGGER trg1_part1 BEFORE INSERT ON part1
               FOR EACH ROW EXECUTE FUNCTION fnc1_part1() ;
CREATE TRIGGER
postgres=> INSERT INTO part1(c1, c2) VALUES (10, 20) ;
INSERT 0 1
postgres=> SELECT * FROM part1 ;
 c1 | c2 |   c3
----+----+---------
 10 | 20 | TRIGGER
(1 row)
```

□ より積極的な Partition Wise Join

パーティション構造が完全に同じでなくてもパーティションワイズ結合が行えるようになりました。この機能はハッシュ・パーティション・テーブルでは動作しません。

□ テーブル全体を使ったパーティション化列

パーティション化列の定義に列全体を指定することができるようになりました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 20 列全体のパーティション指定

```
postgres=> CREATE TABLE part2(c1 INT, c2 INT) PARTITION BY LIST((part2)) ;
CREATE TABLE
postgres=> CREATE TABLE part2v1 PARTITION OF part2 FOR VALUES IN ('(1, 2)') ;
CREATE TABLE
postgres=> CREATE TABLE part2v2 PARTITION OF part2 FOR VALUES IN ('(2, 4)') ;
CREATE TABLE
```

3.1.7. 自動 VACUUM のログ

自動 VACUUM のログに WAL 統計が出力されるようになりました。

例 21 自動 VACUUM ログ

```
2020-09-24 15:59:21.872 JST [31752] LOG:  automatic vacuum of table
"postgres.public.data1": index scans: 1
        pages: 0 removed, 54055 remain, 0 skipped due to pins, 0 skipped frozen
        tuples: 10000000 removed, 0 remain, 0 are dead but not yet removable,
oldest xmin: 514
        buffer usage: 234548 hits, 119331 misses, 130854 dirtied
        avg read rate: 2.780 MB/s, avg write rate: 3.048 MB/s
        system usage: CPU: user: 3.24 s, system: 4.13 s, elapsed: 335.39 s
        WAL usage: 244232 records, 93518 full page images, 303298890 bytes
```

3.1.8. 待機イベント

pg_stat_activity カタログの wait_event 列に出力される待機イベントに以下の変更がありました。

表 17 追加された待機イベント

待機イベント	説明
BackupWaitWalArchive	アーカイブの作成待ち
RecoveryConflictSnapshot	VACUUM クリーンアップ中のリカバリ競合の解消待ち
RecoveryConflictTablespace	表スペース削除時のリカバリ競合の解消待ち
RecoveryPause	スタンバイ・インスタンスの昇格待ち
VacuumDelay	コストベースの VACUUM 遅延
ProcSignalBarrier	バックエンドによるバリアイベントの処理待ち

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

表 18 名前が変更された待機イベント

待機イベント（PostgreSQL 12）	待機イベント（PostgreSQL 13）
AsyncCtlLock	NotifySLRU
AsyncQueueLock	NotifyQueue
CLogControlLock	XactSLRU
ClogGroupUpdate	XactGroupUpdate
CommitTsControlLock	CommitTsSLRU
Hash/Batch/Allocating	HashBatchAllocate
Hash/Batch/Electing	HashBatchElect
Hash/Batch/Loading	HashBatchLoad
Hash/Build/Allocating	HashBuildAllocate
Hash/Build/Electing	HashBuildElect
Hash/Build/HashingInner	HashBuildHashInner
Hash/Build/HashingOuter	HashBuildHashOuter
Hash/GrowBatches/Allocating	HashGrowBatchesAllocate
Hash/GrowBatches/Deciding	HashGrowBatchesDecide
Hash/GrowBatches/Electing	HashGrowBatchesElect
Hash/GrowBatches/Finishing	HashGrowBatchesFinish
Hash/GrowBatches/Repartitioning	HashGrowBatchesRepartition
Hash/GrowBuckets/Allocating	HashGrowBucketsAllocate
Hash/GrowBuckets/Electing	HashGrowBucketsElect
Hash/GrowBuckets/Reinserting	HashGrowBucketsReinsert
MultiXactOffsetControlLock	MultiXactOffsetSLRU
MultiXactMemberControlLock	MultiXactMemberSLRU
OldSerXidLock	SerialSLRU
RecoveryWalAll	RecoveryWalStream
RecoveryWalStream	RecoveryRetrieveRetryInterval
SerializablePredicateLockListLock	SerializablePredicateList
SubtransControlLock	SubtransSLRU
speculative token	spectoken

イベント名の後端から Lock を削除する等、上記の表以外にもイベント名の変更が多数あります。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.1.9. libpq 接続文字列

以下の libpq 接続文字列が追加／変更されました。

□ channel_binding パラメーター

クライアント接続文字列にチャネル・バインディングを制御する channel_binding パラメーターが追加されました。指定できる値は以下の通りです。

表 19 channel_binding パラメーター設定値

設定値	説明
require	必須
prefer	クライアントが選択
disable	無効

SSL が有効になっている場合は prefer がデフォルト値です。SSL が無効にされている場合は disable がデフォルト値です。環境変数 PGCHANNELBINDING はこのパラメーターと同じ値を指定できます。

□ sslkey パラメーター

SSL 接続のキーファイルに ASN.1 DER フォーマットのファイルを指定できるようになりました。

□ ssl_min_protocol_version, ssl_max_protocol_version パラメーター

SSL/TLS プロトコルの最小バージョン（ssl_min_protocol_version）と最大バージョン（ssl_max_protocol_version）を指定します。使用できる値は TLSv1、TLSv1.1、TLSv1.2、TLSv1.3 です。ssl_min_protocol_version のデフォルト値は TLSv1.2 です。ssl_max_protocol_version の値が指定されない場合はバックエンドの値が使用されます。パラメーターで指定する代わりに環境変数 PGSSLMINPROTOCOLVERSION と PGSSLMAXPROTOCOLVERSION を使用することもできます。

□ sslpassword パラメーター

sslkey パラメーターで指定された秘密鍵のパスワードを指定します。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.1.10. libpq 関数

以下の libpq 関数が追加されました。

- BufferUsageAccumDiff
- TupleHashTableHash
- LookupTupleHashEntryHash
- PQsetSSLKeyPassHook_OpenSSL
- LogicalTapeSetExtend

3.1.11. Hook

以下のフック関数が追加されました。

□ TLS 初期化フック

TLS の初期化フックが提供されました。openssl_tls_init_hook にコールバック関数を指定します。

□ TRUNCATE フック

TRUNCATE 文に対する強制アクセス制御（MAC）を許可するためにフックが実行されます。Contrib モジュール sepgsql で TRUNCATE 文に対するアクセス制御を実施できます。

3.1.12. 列トリガー

ロジカル・レプリケーション環境のサブスクリプション側で列トリガーが実行されるようになりました。

3.1.13. ローカル接続のキー

インスタンスが使用する共有メモリーのキー情報は、従来接続待ちポート番号（パラメーターport）を元に決定していました。PostgreSQL 13 ではデータベース・クラスタの i ノード番号を使用するようになりました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 22 共有メモリーのキー番号

```
$ ls -lid data
33832487 drwx------. 20 postgres postgres 4096 Sep 24 08:11 data
$ ipcs -m

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x02043e27 196610     postgres   600        56         6
```

上記の例では、i ノード番号が 33,832,487 であり、16 進数に変換すると 0x02043e27 になります。

3.1.14. Trusted Extension

エクステンションのコントロール・ファイル（{extension}.control）に記述される属性に trusted が追加されました。この属性のデフォルト値は off です。この属性値に on が指定されたエクステンションは SUPERUSER 属性を持たないユーザーも CREATE EXTENSION 文を実行できます。ユーザーはデータベースに対して CREATE 権限を持つ必要があります。

例 23 plpgsql.control ファイルの内容

```
$ cat plpgsql.control
# plpgsql extension
comment = 'PL/pgSQL procedural language'
default_version = '1.0'
module_pathname = '$libdir/plpgsql'
relocatable = false
schema = pg_catalog
superuser = true
trusted = true
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 24 一般ユーザーで実行する CREATE EXTENSION 文

```
postgres=# CREATE USER demo PASSWORD '<<パスワード>>' ;
CREATE ROLE
postgres=# GRANT CREATE ON DATABASE postgres TO demo ;
GRANT
postgres=# ¥connect postgres demo
You are now connected to database "postgres" as user "demo".
postgres=> CREATE EXTENSION hstore ;
CREATE EXTENSION
```

以下のエクステンションの trusted 設定に on が指定されています。

- bool_plperl
- btree_gin
- btree_gist
- citext
- cube
- dict_int
- fuzzystrmatch
- hstore
- intarray.
- isn
- jsonb_plperl
- lo
- ltree
- pgcrypto
- pg_trgm
- plperl
- plpgsql
- seg
- tablefunc
- tcn
- tsm_system_rows
- tsm_system_time
- unaccent

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.1.15. レプリケーション・スロット

ストリーミング・レプリケーション環境のスタンバイ・インスタンスでパラメーターprimary_slot_name が指定されていない場合、プライマリー・インスタンス上にテンポラリ・レプリケーション・スロットを作成することができます。スタンバイ・インスタンスのパラメーターwal_receiver_create_temp_slot を on（デフォルト値 off）にする必要があります。

例 25 自動作成されたレプリケーション・スロットの情報

```
postgres=# SELECT * FROM pg_replication_slots ;
-[ RECORD 1 ]-------+--------------------
slot_name           | pg_walreceiver_4273
plugin              |
slot_type           | physical
datoid              |
database            |
temporary           | t
active              | t
active_pid          | 4273
xmin                |
catalog_xmin        |
restart_lsn         | 0/390005A0
confirmed_flush_lsn |
wal_status          | reserved
safe_wal_size       |
```

3.1.16. テキスト検索

テキスト検索の対応言語が増えました。PostgreSQL 12 では 22 言語でしたが、PostgreSQL 13 ではギリシャ語が増えています。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 26 テキスト検索設定

```
postgres=> ¥dF
               List of text search configurations
   Schema   |    Name    |              Description
------------+------------+-----------------------------------------
 pg_catalog | arabic     | configuration for arabic language
 pg_catalog | danish     | configuration for danish language
 pg_catalog | dutch      | configuration for dutch language
 pg_catalog | english    | configuration for english language
 pg_catalog | finnish    | configuration for finnish language
 pg_catalog | french     | configuration for french language
 pg_catalog | german     | configuration for german language
 pg_catalog | greek      | configuration for greek language << new
 pg_catalog | hungarian  | configuration for hungarian language
<<以下省略>>
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.2. SQL 文の拡張

ここでは SQL 文に関係する新機能を説明しています。

3.2.1. ALTER NO DEPENDS ON

ファンクション、インデックス、マテリアライズド・ビュー、トリガーに対する ALTER 文で、エクステンションに対する依存設定を削除できるようになりました。

構文

```
ALTER object_name NO DEPENDS ON EXTENSION extension_name
```

例 27 ALTER NO DEPENDS ON 文

```
postgres=> ALTER FUNCTION func1 DEPENDS ON EXTENSION cube ;
ALTER FUNCTION
postgres=> ALTER FUNCTION func1 NO DEPENDS ON EXTENSION cube ;
ALTER FUNCTION
```

3.2.2. ALTER STATISTICS SET STATISTICS

ANALYZE 実行時の統計収集ターゲットを指定します。設定できる値は 0～10,000 です。10,000 以上を指定した場合は 10,000 とみなされます。デフォルト値にリセットする場合は -1 を指定します。

構文

```
ALTER STATISTICS statistic_name SET STATISTICS new_target
```

指定した値は pg_statistic_ext カタログの stxstattarget 列または psql ユーティリティの ¥d コマンドで確認することができます。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 28 ALTER STATISTICS SET STATISTICS 文

```
postgres=> CREATE TABLE data1(c1 INT, c2 INT, c3 VARCHAR(10)) ;
CREATE TABLE
postgres=> CREATE STATISTICS stat1 ON c1, c2 FROM data1 ;
CREATE STATISTICS
postgres=> ALTER STATISTICS stat1 SET STATISTICS 10000 ;
ALTER STATISTICS
postgres=> SELECT stxname, stxstattarget FROM pg_statistic_ext
        WHERE stxname='stat1' ;
 stxname | stxstattarget
---------+--------------
 stat1   |         10000
(1 row)
postgres=> ¥d data1
                    Table "public.data1"
 Column |         Type          | Collation | Nullable | Default
--------+-----------------------+-----------+----------+---------
 c1     | integer               |           |          |
 c2     | integer               |           |          |
 c3     | character varying(10) |           |          |
Statistics objects:
    "public"."stat1" (ndistinct, dependencies, mcv) ON c1, c2 FROM data1;
STATISTICS 10000
```

3.2.3. ALTER TABLE

ALTER TABLE 文には以下の拡張が追加されました。

□ ALTER TABLE ALTER COLUMN DROP EXPRESSION

ALTER TABLE ALTER COLUMN DROP EXPRESSION 文を実行することで GENERATED ALWAYS 句を指定した列から自動作成の定義のみを削除できるようになりました。計算情報を削除するだけで、列値はそのまま維持されます。IF EXISTS 句を指定することもできます。

構文

```
ALTER TABLE table_name ALTER COLUMN column_name DROP EXPRESSION [IF EXISTS]
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 29 ALTER TABLE DROP EXPRESSION 文

```
postgres=> CREATE TABLE gen1(c1 INT, c2 INT, c3 INT GENERATED ALWAYS AS
        (c1 + c2) STORED) ;
CREATE TABLE
postgres=> ¥d gen1
                              Table "public.gen1"
 Column |  Type   | Collation | Nullable |              Default
--------+---------+-----------+----------+-----------------------------------
 c1     | integer |           |          |
 c2     | integer |           |          |
 c3     | integer |           |          | generated always as (c1 + c2)
stored

postgres=> INSERT INTO gen1(c1, c2) VALUES (100, 200) ;
INSERT 0 1
postgres=> ALTER TABLE gen1 ALTER COLUMN c3 DROP EXPRESSION IF EXISTS ;
ALTER TABLE
postgres=> ¥d gen1
                Table "public.gen1"
 Column |  Type   | Collation | Nullable | Default
--------+---------+-----------+----------+---------
 c1     | integer |           |          |
 c2     | integer |           |          |
 c3     | integer |           |          |

postgres=> SELECT * FROM gen1 ;
 c1  | c2  | c3
-----+-----+-----
 100 | 200 | 300
(1 row)
```

□ ALTER TABLE ALTER COLUMN SET STORAGE

列の STORAGE 属性を変更が、インデックスにも伝播するようになりました。この機能は最新の PostgreSQL 12、PostgreSQL 11 にもバックポートされました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 30 ALTER TABLE SET STORAGE 文

```
postgres=> CREATE TABLE attr1(c1 CHAR(10)) ;
CREATE TABLE
postgres=> ¥d+ attr1
                                        Table "public.attr1"
 Column |     Type      | Collation | Nullable | Default | Storage  | …
--------+---------------+-----------+----------+---------+----------+ …
 c1     | character(10) |           |          |         | extended | …
Access method: heap

postgres=> CREATE INDEX idx1_attr1 ON attr1(c1) ;
CREATE INDEX
postgres=> ALTER TABLE attr1 ALTER COLUMN c1 SET STORAGE EXTERNAL ;
ALTER TABLE
postgres=> ¥d+ idx1_attr1
                      Index "public.idx1_attr1"
 Column |     Type      | Key? | Definition | Storage  | Stats target
--------+---------------+------+------------+----------+--------------
 c1     | character(10) | yes  | c1         | external |
btree, for table "public.attr1"
```

3.2.4. ALTER TYPE

ALTER TYPE 文では SET 句による属性変更ができるようになりました。

構文

```
ALTER TYPE type_name SET (attribute = value)
```

変更できる属性は以下の通りです。CREATE TYPE 文で指定できる属性の一部になります。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

表 20　変更できる属性

属性名	説明
RECEIVE	外部表現を内部表現に変換する関数
SEND	内部表現からバイナリに変換する関数
TYPMOD_IN	修飾子のサポート関数
TYPMOD_OUT	修飾子のサポート関数
ANALYZE	統計情報の収集関数
STORAGE	可変長データの格納方法を指定する関数

例 31 ALTER TYPE SET 文

```
postgres=# ALTER TYPE box SET (SEND = myboxsend) ;
ALTER TYPE
```

3.2.5. ALTER VIEW

ビューに含まれる列名を変更できるようになりました。ALTER VIEW RENAME COLUMN 文を実行します。

構文

```
ALTER VIEW view_name RENAME COLUMN old_name TO new_name
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 32 ビューの列名変更

```
postgres=> CREATE VIEW customer_name AS SELECT c_customer_id,
        c_first_name || ',' || c_last_name c_full_name FROM customer ;
CREATE VIEW
postgres=> ¥d customer_name
                  View "public.customer_name"
    Column     |     Type      | Collation | Nullable | Default
---------------+---------------+-----------+----------+---------
 c_customer_id | character(16) |           |          |
 c_full_name   | text          |           |          |

postgres=> ALTER VIEW customer_name RENAME COLUMN c_full_name TO full_name ;
ALTER VIEW
postgres=> ¥d customer_name
                  View "public.customer_name"
    Column     |     Type      | Collation | Nullable | Default
---------------+---------------+-----------+----------+---------
 c_customer_id | character(16) |           |          |
 full_name     | text          |           |          |
```

3.2.6. CREATE DATABASE

CREATE DATABASE 文のオプションに LOCALE 句を指定できるようになりました。LC_CTYPE 句や LC_COLLATE 句とは同時に指定できません。

構文

```
CREATE DATABASE database_name [[WITH] LOCALE [=] 'locale_name']
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 33 LOCALE の指定

```
postgres=# CREATE DATABASE localedb1 WITH LOCALE='ja_JP.eucjp' ENCODING='eucjp'
        TEMPLATE=template0 ;
CREATE DATABASE
postgres=# ¥l localedb1
                                  List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    | Access privileges
-----------+----------+----------+-------------+-------------+-------------------
 localedb1 | postgres | EUC_JP   | ja_JP.eucjp | ja_JP.eucjp |
(1 row)
```

3.2.7. CREATE INDEX

B-Tree インデックスにページ内の重複排除機能が追加されました。この機能はデフォルトで有効になっています。この機能は無効にするには deduplicate_items 属性を off に指定します。

例 34 B-Tree インデックスの重複排除

```
postgres=> CREATE TABLE data1(c1 INT, c2 VARCHAR(10), c3 VARCHAR(10)) ;
CREATE TABLE
postgres=> INSERT INTO data1 VALUES (generate_series(1, 1000000), 'data1',
'data1') ;
INSERT 0 1000000
postgres=> CREATE INDEX idx1_dup ON data1(c2) WITH (deduplicate_items = off) ;
CREATE INDEX
postgres=> CREATE INDEX idx2_dedup ON data1(c3) WITH (deduplicate_items = on) ;
CREATE INDEX
postgres=> ¥d+ idx1_dup
                          Index "public.idx1_dup"
 Column |         Type          | Key? | Definition | Storage  | Stats target
--------+-----------------------+------+------------+----------+--------------
 c2     | character varying(10) | yes  | c2         | extended |
btree, for table "public.data1"
Options: deduplicate_items=off
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 35 重複排除の効果

```
postgres=> SELECT pg_size_pretty(pg_relation_size('idx1_dup')) ;
 pg_size_pretty
----------------
 21 MB
(1 row)
postgres=> SELECT pg_size_pretty(pg_relation_size('idx2_dedup')) ;
 pg_size_pretty
----------------
 6792 kB
(1 row)
```

3.2.8. CREATE TABLE

CREATE TABLE/ALTER TABLE 文ではいくつかの属性[4]を指定できるようになりました。

例 36 テーブルに対する追加属性の指定

```
postgres=> CREATE TABLE data1(c1 NUMERIC, c2 VARCHAR(10)) ;
CREATE TABLE
postgres=> ALTER TABLE data1 SET (toast.vacuum_index_cleanup = off) ;
ALTER TABLE
postgres=> ALTER TABLE data1 SET (autovacuum_vacuum_insert_threshold = 10000) ;
ALTER TABLE
postgres=> ALTER TABLE data1 SET (autovacuum_vacuum_insert_scale_factor = 0.1) ;
ALTER TABLE
```

3.2.9. CREATE TABLESPACE

テーブル空間の属性に maintenance_io_concurrency[5]を指定できるようになりました。

[4]「3.3.1 追加されたパラメーター」参照

[5]「3.3.1 追加されたパラメーター」参照

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 37 maintenance_io_concurrency 属性の指定

```
postgres=# CREATE TABLESPACE ts1 LOCATION '/home/postgres/ts1'
                WITH (maintenance_io_concurrency = 20) ;
CREATE TABLESPACE
postgres=# ALTER TABLESPACE ts1 SET (maintenance_io_concurrency = 30) ;
ALTER TABLESPACE
postgres=# ¥db+ ts1
                                                    List of tablespaces
 Name |  Owner   |      Location       | Access privileges |            Options
|  Size   | Description
------+----------+---------------------+-------------------+--------------------
------------+---------+-------------
 ts1    |  postgres  |  /home/postgres/ts1  |                                   |
{maintenance_io_concurrency=30} | 0 bytes |
(1 row)
```

3.2.10. DROP DATABASE FORCE

接続中のセッションが存在しても強制的にデータベースを削除できるようになりました。接続中のセッションは強制的にクローズされます。DROP DATABASE 文を発行するセッションが接続しているデータベースは削除できません。

構文

```
DROP DATABASE database_name [[WITH] (FORCE)]
```

例 38 DROP DATABASE 文

```
postgres=# DROP DATABASE demodb WITH (FORCE) ;
DROP DATABASE
postgres=# DROP DATABASE postgres WITH (FORCE) ;
ERROR:  cannot drop the currently open database
```

3.2.11. EXPLAIN ANALYZE

EXPLAIN ANALYZE 文には以下の拡張が実装されました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

□　キャッシュ情報の出力

BUFFERS を指定することで、実行計画作成時の共有バッファの状態が出力されるようになりました。

例 39 EXPLAIN (ANALYZE, BUFFERS)文

```
postgres=> EXPLAIN (ANALYZE, BUFFERS) SELECT * FROM data1 WHERE c1 = 10000 ;
                                    QUERY PLAN
-------------------------------------------------------------------------------
Index Scan using idx1_data1 on data1  (cost=0.42..8.44 rows=1 width=16) (actual
 time=0.029..0.030 rows=1 loops=1)
   Index Cond: (c1 = 10000)
   Buffers: shared hit=1 read=3
 Planning:
   Buffers: shared hit=17 read=1
 Planning Time: 1.380 ms
 Execution Time: 0.044 ms
(7 rows)
```

ANALYZE 句を指定していない場合はプランナーのバッファ利用量が出力されます。

例 40 EXPLAIN (BUFFERS)文

```
postgres=> EXPLAIN (BUFFERS) SELECT * FROM data1 WHERE c1=100 ;
                                QUERY PLAN
-------------------------------------------------------------------------
 Index Scan using idx1_data1 on data1  (cost=0.43..19.68 rows=4 width=16)
   Index Cond: (c1 = 100)
 Planning:
   Buffers: shared hit=93 read=3
(4 rows)
```

□　ソート情報の出力

パラレル・クエリーを実行した場合の EXPLAIN ANALYZE 文の出力結果が変更されました。JIT 機能の情報がワーカー単位に出力されるようになりました。またソート情報の出力方法が変更されました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 41 EXPLAIN ANALYZE 文の出力（JIT）

```
postgres=> EXPLAIN (ANALYZE, VERBOSE) SELECT * FROM data1 WHERE c1 < 10000 ;
                                                              QUERY PLAN
---------------------------------------------------------------------------------------------------------
 Gather     (cost=1000.00..162274.35   rows=10052   width=12)   (actual   time=122.039..434.906
rows=9999 loops=1)
   Output: c1, c2
   Workers Planned: 2
   Workers Launched: 2
   ->  Parallel Seq Scan on public.data1  (cost=0.00..160269.15 rows=4188 width=12) (actual
time=80.639..384.760 rows=3333 loops=3)
         Output: c1, c2
         Filter: (data1.c1 < '10000'::numeric)
         Rows Removed by Filter: 3330000
         Worker 0:  actual time=60.034..361.329 rows=185 loops=1
           JIT:
             Functions: 2
             Options: Inlining false, Optimization false, Expressions true, Deforming true
             Timing: Generation 1.994 ms, Inlining 0.000 ms, Optimization 1.503 ms, Emission
20.786 ms, Total 24.284 ms
         Worker 1:  actual time=60.056..361.358 rows=184 loops=1
           JIT:
             Functions: 2
             Options: Inlining false, Optimization false, Expressions true, Deforming true
             Timing: Generation 1.967 ms, Inlining 0.000 ms, Optimization 1.486 ms, Emission
20.803 ms, Total 24.256 ms
 Planning Time: 0.081 ms
<<以下省略>>
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 42 EXPLAIN ANALYZE 文の出力（ソート）

```
postgres=> EXPLAIN (ANALYZE, VERBOSE) SELECT * FROM data1 ORDER BY c1 ;
                                                                QUERY PLAN
-----------------------------------------------------------------------------------------------
 Gather    Merge        (cost=752323.83..1726047.95    rows=8345624    width=12)    (actual
 time=1471.139..4003.430 rows=10000000 loops=1)
   Output: c1, c2
   Workers Planned: 2
   Workers Launched: 2
   ->        Sort        (cost=751323.81..761755.84    rows=4172812    width=12)    (actual
time=1328.103..1849.320 rows=3333333 loops=3)
         Output: c1, c2
         Sort Key: data1.c1
         Sort Method: external merge  Disk: 74864kB
         Worker 0:  actual time=1466.829..1989.939 rows=3335746 loops=1
           Sort Method: external merge  Disk: 75096kB
         Worker 1:  actual time=1360.627..1885.937 rows=3338879 loops=1
           Sort Method: external merge  Disk: 75168kB
         ->  Parallel Seq Scan on public.data1  (cost=0.00..149837.12 rows=4172812 width=12)
(actual time=60.222..279.624 rows=3333333 loops=3)
               Output: c1, c2
               Worker 0:  actual time=58.974..280.404 rows=3335746 loops=1
               Worker 1:  actual time=58.948..277.968 rows=3338879 loops=1
 Planning Time: 0.056 ms
 Execution Time: 4261.051 ms
(18 rows)
```

□ WAL 情報の出力

EXPLAIN ANALYZE 文に WAL オプションを指定すると、生成された WAL の情報が出力されます。

例 43 EXPLAIN (ANALYZE, WAL)文の出力

```
postgres=> EXPLAIN (ANALYZE, WAL) DELETE FROM data1 WHERE c1 = 20000 ;
                                  QUERY PLAN
------------------------------------------------------------------------------
Delete on data1  (cost=0.00..17906.00 rows=1 width=6) (actual time=56.817..56.8
18 rows=0 loops=1)
   WAL: records=2 fpi=1 bytes=8237
   ->  Seq Scan on data1  (cost=0.00..17906.00 rows=1 width=6) (actual time=1.26
9..56.800 rows=1 loops=1)
         Filter: (c1 = 20001)
         Rows Removed by Filter: 999998
         WAL: records=1 fpi=1 bytes=8183
 Planning Time: 0.085 ms
 Execution Time: 56.843 ms
(8 rows)
```

3.2.12. INSERT

GENERATED ALWAYS 句を指定された IDENTITY 列に対して INSERT OVERRIDING USER VALUE 文による値の指定が許可されるようになりました。ただし、IDENTITY 列に指定した値は無視されます。

例 44 INSERT OVERRIDING USER VALUE 文

```
postgres=> CREATE TABLE gen1(c1 INT GENERATED ALWAYS AS IDENTITY, c2
VARCHAR(10)) ;
CREATE TABLE
postgres=> INSERT INTO gen1 OVERRIDING USER VALUE VALUES (100, 'data1') ;
INSERT 0 1
postgres=> SELECT * FROM gen1 ;
 c1 |  c2
----+-------
  1 | data1
(1 row)
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.2.13. JSON

以下の JSON 対応機能が追加されました。

□ Unicode エスケープの許可

サーバーの文字エンコードが UTF-8 以外の場合でも、文字リテラルと識別子に Unicode エスケープが許可されるようになりました。

□ jsonpath の datetime メソッド

文字列から日付／時刻型に変換する datetime メソッドが追加されました。

例 45 datetime メソッド

```
postgres=> SELECT
    jsonb_path_query('"27-09-2020"', '$.datetime("dd-mm-yyyy")') ;
 jsonb_path_query
------------------
 "2020-09-27"
(1 row)
```

3.2.14. MAX/MIN pg_lsn

pg_lsn 型に対して MAX/MIN 関数が使用できるようになりました。

例 46 MAX(pg_lsn)関数

```
postgres=# SELECT MIN(flush_lsn), MAX(flush_lsn) FROM pg_stat_replication ;
    min    |    max
-----------+-----------
 0/94FD1E0 | 0/94FD1E0
(1 row)
```

3.2.15. ROW

ROW 式から一部のデータ型でフィールドを直接抽出できるようになりました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 47 ROW 式から値の抽出

```
postgres=> SELECT (ROW(2, 3.1)).f1, (ROW(4, 5.1)).f2 ;
 f1 | f2
----+-----
  2 | 5.1
(1 row)
```

3.2.16. SELECT FETCH FIRST WITH TIES

SELECT FETCH FIRST 文に WITH TIES 句を指定できるようになりました。ROWS 句で指定したタプル数に同一値のタプルも含めて出力します。

例 48 SELECT FETCH FIRST WITH TIES 文

```
postgres=> SELECT * FROM data1 WHERE c1 < 4 ORDER BY c1
             FETCH FIRST 2 ROWS ONLY ;
 c1 |  c2
----+-------
  1 | data1
  1 | data1
(2 rows)

postgres=> SELECT * FROM data1 WHERE c1 < 4 ORDER BY c1
             FETCH FIRST 2 ROWS WITH TIES ;
 c1 |  c2
----+-------
  1 | data1
  1 | data1
  1 | data1
(3 rows)
```

3.2.17. VACUUM PARALLEL

インデックスに対する VACUUM 文がパラレル化できるようになりました。パラレル化はデフォルトで動作し、並列度はテーブルに作成されたインデックス数から決定されます。並列度を指定する場合は PARALLEL 句に 0～1024 の値を指定します。この機能は自動 VACUUM では動作しません。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 49 VACUUM 文の PARALLEL 指定

```
postgres=> VACUUM (PARALLEL 4, VERBOSE) data1 ;
INFO:  vacuuming "public.data1"
INFO:  launched 2 parallel vacuum workers for index vacuuming (planned: 2)
INFO:  scanned index "idx3_data1" to remove 1000 row versions by parallel vacuum
worker
DETAIL:  CPU: user: 0.39 s, system: 0.11 s, elapsed: 12.45 s
INFO:  scanned index "idx2_data1" to remove 1000 row versions by parallel vacuum
worker
DETAIL:  CPU: user: 0.39 s, system: 0.12 s, elapsed: 12.56 s
INFO:  scanned index "idx1_data1" to remove 1000 row versions
DETAIL:  CPU: user: 0.47 s, system: 0.17 s, elapsed: 53.26 s
INFO:  "data1": removed 1000 row versions in 1000 pages
<<以下省略>>
```

3.2.18. オペレーター<->

距離を示すオペレーター (<->) がいくつかのデータ型間で使用できるようになりました。また box_ops 演算子クラスで GiST インデックスや SP-GiST インデックスが利用できるようになりました。

例 50 オペレーター<->

```
postgres=> CREATE TABLE point_tbl(f1 point) ;
CREATE TABLE
postgres=> CREATE TABLE box_tbl(f1 box) ;
CREATE TABLE
postgres=> SELECT p.f1, b.f1, p.f1 <-> b.f1 AS dist_pb, b.f1 <-> p.f1 AS dist_bp
                FROM point_tbl p, box_tbl b ;
f1 | f1 | dist_pb | dist_bp
----+----+---------+---------
(0 rows)
```

3.2.19. 関数

以下の関数が追加／拡張されました。

□ gen_random_uuid

ランダムな UUID を生成する gen_random_uuid 関数が提供されました。

例 51 gen_random_uuid 関数

```
postgres=> SELECT gen_random_uuid() ;
            gen_random_uuid
--------------------------------------
 3759712a-e7f6-4348-b62c-85535c831e7e
(1 row)
```

□ 日付フォーマット

日付のフォーマットに SQL Standard 2016 で定義された FF1 から FF6 フォーマット・パターンが追加されました。

例 52 日付フォーマット

```
postgres=> SELECT TO_CHAR(current_timestamp, 'HH:MI:SS.FF6') ;
     to_char
-----------------
 01:39:56.077670
(1 row)
```

表 21 追加されたパターン

パターン	出力結果	備考
FF1	10 分の 1 秒	
FF2	100 分の 1 秒	
FF3	ミリ秒	
FF4	10 分の 1 ミリ秒	
FF5	100 分の 1 ミリ秒	
FF6	マイクロ秒	

パターン SSSSS は SSSS のエイリアスになりました。

□ min_scale, trim_scale

min_scale 関数は NUMERIC 型から小数点以下の末尾 0 を除く最小桁数を取得します。trim_scale 関数は NUMERIC 型から小数点以下の末尾 0 を除く値を取得します。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 53 min_scale, trim_scale 関数

```
postgres=> SELECT trim_scale(1.234000::numeric), min_scale(1.234000::numeric) ;
 trim_scale | min_scale
------------+-----------
      1.234 |         3
(1 row)
```

□ jsonb_set_lax

jsonb_set_lax 関数は null_value_treatment パラメーター（text 型）が追加された以外は jsonb_set 関数と同じです。このパラメーターには以下の値を指定できます。

表 22 null_value_treatment 設定値

値	説明	備考
use_json_null	NULL 値を返す	デフォルト値
raise_exception	例外をスローする	
delete_key	キーを削除する	
return_target	ターゲットを返す	

例 54 jsonb_set_lax 関数

```
postgres=> SELECT
        jsonb_set_lax('[{"f1":1,"f2":null},2,null,3]', '{0,f1}', null) ;
             jsonb_set_lax
-------------------------------------
 [{"f1": null, "f2": null}, 2, null, 3]
(1 row)
```

□ gcd, lcm

最小公倍数を返す lcm 関数と、最大公約数を返す gcd 関数が追加されました。同じ型の数値を２つパラメーターに指定します。両方のパラメーターが 0 の場合、これらの関数は 0 を返します。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 55 gcd, lcm 関数

```
postgres=> SELECT gcd(1071, 462), lcm(1071, 46) ;
 gcd |  lcm
-----+-------
  21 | 49266
(1 row)
```

□ get_bit / set_bit

bytea 型を指定する get_bit / set_bit 関数の offset パラメーターが integer 型から bigint 型に変更されました。

□ Unicode 正規化

UNICODE の正規化を行う normalize 関数と、正規化されているかをチェックする is_normalized 関数が実装されました。また指定されたテキストをチェックする is normalized 構文を利用できます。

構文

```
テキスト is [not] [form] normalized
```

form の部分は、NFC（デフォルト）、NFD、NFKC または NFKD を指定できます。

例 56 is normalized 構文

```
postgres=> SELECT '㍾' IS NFKD NORMALIZED ;
 is_normalized
---------------
 f
(1 row)
postgres=> SELECT normalize('㍾', NFKD) ;
 normalize
-----------
 令和
(1 row)
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

□ pg_stats_reset_slue 関数

pg_stat_slru カタログのカウンター値をリセットします。この関数のパラメーターに NULL を指定した場合、すべてのカウンターがリセットされます。pg_stat_slru カタログの name 列に出力されるカウンター名を指定すると、一部のカウンターがリセットされます。

例 57 pg_stat_reset_slru 関数

```
postgres=# SELECT pg_stat_reset_slru('subtrans') ;
pg_stat_reset_slru
--------------------


(1 row)
```

3.3. パラメーターの変更

PostgreSQL 13 では以下のパラメーターが変更されました。

3.3.1. 追加されたパラメーター

以下のパラメーターが追加されました。

表 23 追加されたパラメーター

パラメーター	説明（context）	デフォルト値
autovacuum_vacuum_insert_scale_factor	自動 VACUUM が実行される INSERT レコード割合（sighup）	0.2
autovacuum_vacuum_insert_threshold	自動 VACUUM が実行される INSERT レコード数（sighup）	1000
backtrace_functions	エラー発生時にバックトレース出力を行う関数名のリスト（superuser）	''
enable_incremental_sort	インクリメンタル・ソートを有効にする（user）	on
hash_mem_multiplier	ハッシュ操作で使用できるメモリーの最大値計算に使用する（user）	1
ignore_invalid_pages	リカバリー中に無効なページの参照を無視（postmaster）	off
log_min_duration_sample	サンプリングによる SQL 文のログ出力を行う時間（superuser）	-1
log_parameter_max_length	非エラーのメッセージのバインド値出力の最大長（superuser）	-1
log_parameter_max_length_on_error	エラー・メッセージのバインド値出力の最大長（user）	0
log_statement_sample_rate	サンプリングによる SQL 文のログ出力割合（superuser）	1.0
logical_decoding_work_mem	ロジカル・レプリケーション環境のデコード用ワーク・メモリー（user）	64MB
maintenance_io_concurrency	メンテナンス用に使用される I/O 同時実行数（user）	10

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

パラメーター	説明（context）	デフォルト値
max_slot_wal_keep_size	レプリケーション・スロットが保持する WAL の最大 MB サイズ（sighup）	-1
wal_receiver_create_temp_slot	wal receiver プロセスが一時レプリケーション・スロットを作成するか（sighup）	off
wal_skip_threshold	wal_level が 'minimal' の場合に fsync が実行される書き込み間隔（user）	2MB

□ バックトレースの出力

パラメーターbacktrace_functions にはエラーが発生した場合にトレース出力を行う関数名を指定します。ここで指定する関数は SQL 関数ではなく、ソースコード上の関数名です。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 58 backtrace_functions パラメーター

```
postgres=# SET backtrace_functions = 'pg_strtoint32' ;
SET
postgres=# SELECT int 'foobar' ;
ERROR:  invalid input syntax for type integer: "foobar"
LINE 1: SELECT int 'foobar' ;
postgres=# ¥q
$ tail data/log/postgresql.log
2020-09-24 18:36:42.452 JST [2532] BACKTRACE:
        postgres: postgres postgres [local] SELECT() [0x807d2e]
        postgres: postgres postgres [local] SELECT(int4in+0xd) [0x7d27ad]
        postgres: postgres postgres [local] SELECT(InputFunctionCall+0x67) [0x884d27]
        postgres:  postgres  postgres  [local]  SELECT(OidInputFunctionCall+0x31)
[0x884f31]
        postgres: postgres postgres [local] SELECT(coerce_type+0x361) [0x5785f1]
        postgres:  postgres  postgres  [local]  SELECT(coerce_to_target_type+0x9b)
[0x577d8b]
<<途中省略>>
        postgres:  postgres  postgres  [local]  SELECT(pg_analyze_and_rewrite+0x13)
[0x76ffa3]
        postgres: postgres postgres [local] SELECT() [0x770483]
        postgres: postgres postgres [local] SELECT(PostgresMain+0x1037) [0x7718f7]
        postgres: postgres postgres [local] SELECT() [0x483e46]
        postgres: postgres postgres [local] SELECT(PostmasterMain+0xe88) [0x6ff668]
        postgres: postgres postgres [local] SELECT(main+0x46a) [0x484a9a]
        /lib64/libc.so.6(__libc_start_main+0xf5) [0x7f19383693d5]
        postgres: postgres postgres [local] SELECT() [0x484b01]
2020-09-24 18:36:42.452 JST [2532] STATEMENT:  SELECT int 'foobar' ;
```

3.3.2. 変更されたパラメーター

以下のパラメーターは設定範囲や選択肢が変更されました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

表 24 変更されたパラメーター

パラメーター	変更内容
allow_system_table_mods	コンテキストが postmaster から superuser に変更されました。
effective_io_concurrency	内部ロジックによる計算値を使用せず、直接指定された値を使うように変更されました。
log_line_prefix	バックエンドタイプ（%b）が追加されました。
log_min_duration_statement	pg_settings ビューの short_desc 列値が変更されました。
max_files_per_process	最小値が 25 から 64 に変更されました。
primary_conninfo	コンテキストが postmaster から sighup に変更されました。
primary_slot_name	コンテキストが postmaster から sighup に変更されました。
ssl_passphrase_command	SUPERUSER のみ参照可能になりました。
track_activity_query_size	最大値がが 102,400 から 1,048,576 に変更されました。
wal_keep_size	wal_keep_segments から名前が変更されました。設定単位はファイル数からサイズに変更されました。

3.3.3. デフォルト値が変更されたパラメーター

以下のパラメーターはデフォルト値が変更されました。

表 25 デフォルト値が変更されたパラメーター

パラメーター	PostgreSQL 12	PostgreSQL 13	備考
server_version	12.4	13.0	
server_version_num	120004	130000	
ssl_min_protocol_version	TLSv1	TLSv1.2	

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.4. ユーティリティの変更

ユーティリティ・コマンドの主な機能強化点を説明します。

3.4.1. dropdb

--force オプション（または-f オプション）が追加されました。接続セッションが存在しても強制的にデータベースを削除します。接続中のセッションはクローズされます。

例 59 dropdb --force オプション

```
$ dropdb --force --echo demodb
SELECT pg_catalog.set_config('search_path', '', false);
DROP DATABASE demodb WITH (FORCE);
```

3.4.2. pg_basebackup

pg_basebackup コマンドには以下の拡張が実装されました。

□ --manifest-checksums オプション

チェックサムを作成するアルゴリズムを指定します。指定できるオプションは CRC32C、SHA224、SHA256、SHA384、SHA512 または NONE です。デフォルト値は CRC32C です。NONE を指定された場合、マニフェストにはチェックサムは含まれません。

□ --manifest-force-encode オプション

マニフェストに書き込まれるファイル名を 16 進数で出力します。デフォルトでは UTF-8 以外を使ったファイル名のみ 16 進数で表現されます。

□ --no-estimate-size オプション

pg_basebackup コマンドは--progress オプションを指定しなくてもデータサイズの見積もりを行うように変更されました。データ量の見積もりを計算しない場合にはオプション--no-estimate-size を指定します。このオプションは--progress とは同時に使用できません。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 60 pg_basebackup ---no-estimate-size オプション

```
$ pg_basebackup --verbose --no-estimate-size -D back
pg_basebackup: initiating base backup, waiting for checkpoint to complete
pg_basebackup: checkpoint completed
pg_basebackup: write-ahead log start point: 1/44000028 on timeline 1
pg_basebackup: starting background WAL receiver
pg_basebackup: created temporary replication slot "pg_basebackup_3822"
pg_basebackup: write-ahead log end point: 1/44000138
pg_basebackup: waiting for background process to finish streaming ...
pg_basebackup: syncing data to disk ...
pg_basebackup: renaming backup_manifest.tmp to backup_manifest
pg_basebackup: base backup completed
```

□ --no-manifest オプション
マニフェストを作成しません。

□ --no-verify-checksums オプション
チェックサムの検証を行いません。

3.4.3. pg_dump

pg_dump コマンドには以下の拡張が実装されました。

□ --include-foreign-data オプション
FOREIGN SERVER のデータをダンプする--include-foreign-data オプションが追加されました。このオプションには FOREIGN SERVER 名のパターンを指定します。このオプションは複数回指定することができます。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 61 pg_dump --include-foreign-data オプション

```
$ pg_dump -d demodb --include-foreign-data=svr1
--
-- PostgreSQL database dump
<<途中省略>>
CREATE FOREIGN TABLE public.foreign1 (
    c1 numeric,
    c2 character varying(10)
)
SERVER svr1;
<<途中省略>>
--
-- Data for Name: data1; Type: TABLE DATA; Schema: public; Owner: demo
--

COPY public.foreign1 (c1, c2) FROM stdin;
100     data1
¥.
<<以下省略>>
```

□ ALTER DEPEND ON EXTENSION 文の出力

ALTER object DEPENDS ON EXTENSION 文が出力されるようになりました。

例 62 ALTER DEPENDS ON EXTENSION

```
$ pg_dump -d demodb | grep EXTENSION
-- Name: cube; Type: EXTENSION; Schema: -; Owner: -
CREATE EXTENSION IF NOT EXISTS cube WITH SCHEMA public;
-- Name: EXTENSION cube; Type: COMMENT; Schema: -; Owner:
COMMENT ON EXTENSION cube IS 'data type for multidimensional cubes';
ALTER FUNCTION public.func1() DEPENDS ON EXTENSION cube;
$
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.4.4. pg_rewind

pg_rewind コマンドには以下の 3 つのオプションが追加されました。

□ --no-ensure-shutdown オプション

PostgreSQL 13 の pg_rewind は正常にシャットダウンしていないデータベース・クラスタを検知した場合にはシングルユーザー・モードで起動を行います。--no-ensure-shutdown コマンドを指定すること、起動処理を行いません。

□ --write-recovery-conf オプション

--write-recovery-conf オプション（または-R オプション）は、レプリケーション関連設定を行います。このオプションは--source-server オプションと一緒に指定する必要があります。

□ --restore-target-wal オプション

--restore-target-wal オプション（または-c オプション）は、ターゲット・クラスターから restore_command パラメーター値を取得し、必要な WAL セグメントをアーカイブから復元します。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 63 pg_rewind コマンド

```
$ pg_rewind --help
pg_rewind resynchronizes a PostgreSQL cluster with another copy of the cluster.

Usage:
  pg_rewind [OPTION]...

Options:
  -c, --restore-target-wal       use restore_command in target configuration to
                                 retrieve WAL files from archives
  -D, --target-pgdata=DIRECTORY  existing data directory to modify
      --source-pgdata=DIRECTORY  source data directory to synchronize with
      --source-server=CONNSTR    source server to synchronize with
  -n, --dry-run                  stop before modifying anything
  -N, --no-sync                  do not wait for changes to be written
                                 safely to disk
  -P, --progress                 write progress messages
  -R, --write-recovery-conf      write configuration for replication
                                 (requires --source-server)
      --debug                    write a lot of debug messages
      --no-ensure-shutdown       do not automatically fix unclean shutdown
  -V, --version                  output version information, then exit
  -?, --help                     show this help, then exit

Report bugs to <pgsql-bugs@lists.postgresql.org>.
PostgreSQL home page: <https://www.postgresql.org/>
```

3.4.5. pg_verifybackup

pg_verifybackup コマンドはバックアップとマニフェストの内容を比較することで整合性をチェックします。このコマンドは以下のチェックを行います。

- マニフェスト・ファイルのバージョン
- マニフェスト・ファイル自体のチェックサム
- ファイルのサイズ
- ファイルのチェックサム
- WAL ファイルの整合性

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

構文

```
pg_verifybackup [オプション] ディレクトリ
```

表 26 pg_verifybackup コマンド・オプション

オプション	短縮形	説明
--exit-on-error	-e	エラー発生時にすぐに終了
--ignore=PATH	-i	指定パスのチェックを行わない
--manifest-path=PATH	-m	マニフェスト・ファイルのパスを指定
--no-parse-wal	-n	WAL をパースしない
--quiet	-q	エラー以外の画面出力を抑制
--skip-checksums	-s	チェックサムのチェックをスキップ
--wal-directory=PATH	-w	WAL ディレクトリを指定
--version	-V	バージョン情報の出力後に終了
--help	-?	ヘルプメッセージの出力

例 64 pg_verifybackup の実行

```
$ pg_basebackup -D back
$ pg_verifybackup back
backup successfully verified
$ echo $?
0
```

□ 検証エラー

ファイルのサイズや内容が変更されている場合、pg_verifybackup コマンドはエラー・メッセージを出力します。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 65 pg_verifybackup の実行

```
$ pg_basebackup -D back
$ vi back/postgresql.conf
$ vi back/pg_hba.conf
$ pg_verifybackup back
pg_verifybackup: error: "pg_hba.conf" has size 4761 on disk but size 4760 in the manifest
pg_verifybackup: error: checksum mismatch for file "postgresql.conf"
$
```

□ 実行エラー

マニフェストが作成されていないバックアップや、tar フォーマットで作成されたバックアップに対して実行するとエラーになります。

例 66 pg_verifybackup コマンドの実行エラー

```
$ pg_basebackup -D back --no-manifest
$ pg_verifybackup back
pg_verifybackup: fatal: could not open file "back/backup_manifest": No such file or directory
$ rm -r back
$ pg_basebackup -D back -Ft
$ pg_verifybackup back
pg_verifybackup: error: "base.tar" is present on disk but not in the manifest
pg_verifybackup: error: "pg_wal.tar" is present on disk but not in the manifest
pg_verifybackup: error: "base/13577/3433" is present in the manifest but not on disk
<<途中省略>>
Try "pg_waldump --help" for more information.
pg_verifybackup: error: WAL parsing failed for timeline 1
$
```

3.4.6. pg_waldump

pg_waldump コマンドには--quiet オプション(または-q オプション)が追加されました。このオプションはエラー以外の画面出力を抑止します。

3.4.7. psql

psql コマンドには以下の機能が追加されました。

□ ¥dAc, ¥dAf, ¥dAo, ¥dAp コマンド

オペレーター情報を出力するコマンドが追加されました。

表 27 追加コマンド

コマンド	出力内容	備考
¥dAc	AM, Input type, Storage type, Operator class, Default?	
¥dAf	AM, Operator family, Applicable types	
¥dAo	AM, Operator family, Operator, Strategy, Purpose	
¥dAp	AM, Operator family, Registered left type, Registered right type, Number, Function	

例 67 ¥dAc コマンド

```
postgres=> ¥dAc
                              List of operator classes
   AM   |         Input type          | Storage type |     Operator class      | Default?
--------+-----------------------------+--------------+-------------------------+----------
 brin   | "char"                      |              | char_minmax_ops         | yes
 brin   | anyrange                    |              | range_inclusion_ops     | yes
 brin   | bigint                      |              | int8_minmax_ops         | yes
 brin   | bit                         |              | bit_minmax_ops          | yes
 brin   | bit varying                 |              | varbit_minmax_ops       | yes
 brin   | box                         |              | box_inclusion_ops       | yes
 brin   | bytea                       |              | bytea_minmax_ops        | yes
 brin   | character                   |              | bpchar_minmax_ops       | yes
 brin   | date                        |              | date_minmax_ops         | yes
 brin   | double precision            |              | float8_minmax_ops       | yes
<<以下省略>>
```

□ ¥g, ¥gx コマンド

¥g および¥gx コマンドに¥pset コマンドで指定する変数を一時的に変更する機能が追加されました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 68 ¥g コマンド

```
postgres=> SELECT * FROM data1 WHERE c1=10 ;
 c1 |  c2
----+-------
 10 | data1
 10 | data1
 10 | data1
(3 rows)

postgres=> ¥g (format=csv)
c1,c2
10,data1
10,data1
10,data1
```

□ ¥warn コマンド

標準エラーにメッセージを出力する¥warn コマンドが追加されました。

例 69 ¥warn コマンド

```
postgres=> ¥warn `date`
Thu Sep 24 21:10:09 JST 2020
```

□ ¥e コマンド

エディタが終了した後、編集された SQL 文が表示されるようになりました。

例 70 ¥e コマンド

```
postgres=> ¥e
postgres=> SELECT * FROM data1
postgres->
```

□ ¥d+コマンド

テーブルの種類が Persistence 列として出力されるようになりました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 71 ¥d+コマンド

```
postgres=> ¥d+
                                    List of relations
  Schema   |        Name        | Type  |  Owner   | Persistence |    Size    | Description
-----------+--------------------+-------+----------+-------------+------------+------------
 pg_temp_3 | temp1              | table | demo     | temporary   | 8192 bytes |
 public    | data1              | table | demo     | permanent   | 42 MB      |
 public    | unlogged1          | table | demo     | unlogged    | 16 kB      |
(3 rows)
```

□ PROMPT2 の指定値

変数 PROMPT2 に%w を使用できるようになりました。この値を指定すると PROMPT1 と同じ長さのスペースに変換されます。

例 72 PROMPT2 変数の指定

```
postgres=> ¥set PROMPT2 '>%w'
postgres=> CREATE TABLE data1(c1 INT, c2 VARCHAR(10))
>           TABLESPACE pg_default ;
CREATE TABLE
```

□ I/O エラーの検知

ファイル出力時のエラーが検知されるようになりました。

□ PROMPT1/PROMPT2 のデフォルト値

プロンプトのデフォルト値にトランザクション状態を示す%x が含まれるようになりました[6]。

3.4.8. reindexdb

reindexdb コマンドに並列処理を実行するための--jobs オプションが追加されました。デフォルト値は 1 で並列処理を行いません。

例 73 reindexdb --jobs オプション

```
$ reindexdb --concurrently --jobs 2 postgres
reindexdb: warning: cannot reindex system catalogs concurrently, skipping all
```

3.4.9. vacuumdb

--parallel オプション（または-P オプション）が追加されました。オプションには並列度（0～1024）を指定します。このオプションは--full や--analyze-only オプションとは同時に指定できません。

例 74 vacuumdb --parallel オプション

```
$ vacuumdb --parallel=4 postgres
vacuumdb: vacuuming database "postgres"
```

3.4.10. その他

多くのコマンドの--help オプションの出力に www.postgresql.org への URL が追加されました。

例 75 --help オプション

```
$ pg_controldata --help
pg_controldata displays control information of a PostgreSQL database cluster.

Usage:
  pg_controldata [OPTION] [DATADIR]

Options:
 [-D, --pgdata=]DATADIR  data directory
  -V, --version          output version information, then exit
  -?, --help             show this help, then exit

If no data directory (DATADIR) is specified, the environment variable PGDATA
is used.

Report bugs to <pgsql-bugs@lists.postgresql.org>.
PostgreSQL home page: <https://www.postgresql.org/>
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

3.5. Contrib モジュール

Contrib モジュールに関する新機能を説明しています。

3.5.1. adminpack

エクステンション adminpack には pg_file_sync 関数が追加されました。この関数は指定されたファイルまたはディレクトリに対して確実な書き込みを保証します。

例 76 pg_file_sync 関数

```
postgres=# CREATE EXTENSION adminpack ;
CREATE EXTENSION
postgres=# SELECT pg_file_sync('log/postgresql-2020-09-24_120622.log') ;
 pg_file_sync
--------------


(1 row)
```

3.5.2. auto_explain

エクステンション auto_explain には WAL の情報を出力する log_wal パラメーターが追加されました。デフォルト値は off で、WAL の情報は出力されません。このパラメーターを on に指定する場合には auto_explain.log_analyze も on に指定する必要があります。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 77 log_wal パラメーター

```
postgres=# LOAD 'auto_explain' ;
LOAD
postgres=# SET auto_explain.log_wal = on ;
SET
postgres=# SET auto_explain.log_analyze = on ;
SET
postgres=# SET auto_explain.log_min_duration = 0 ;
SET
postgres=# DELETE FROM data1 WHERE c1=100 ;
DELETE 1
postgres=# ¥! tail -7 log/postgres.log
2020-09-24 21:16:28.707 JST [4731] LOG:  duration: 0.064 ms  plan:
        Query Text: DELETE FROM data1 WHERE c1=100 ;
        Delete  on  data1     (cost=0.00..4.50  rows=2  width=6)  (actual
time=0.063..0.063 rows=0 loops=1)
          WAL: records=2 fpi=2 bytes=8966
          ->  Seq Scan on data1  (cost=0.00..4.50 rows=2 width=6) (actual
time=0.018..0.026 rows=2 loops=1)
                Filter: (c1 = '100'::numeric)
                Rows Removed by Filter: 198
```

3.5.3. dict_int

dict_int エクステンションには ABSVAL 属性が追加されました。この属性を指定すると整数値に指定された符号を無視します。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 78 ABSVAL 属性

```
postgres=# CREATE EXTENSION dict_int ;
CREATE EXTENSION
postgres=# ALTER TEXT SEARCH DICTIONARY intdict (ABSVAL = true) ;
ALTER TEXT SEARCH DICTIONARY
postgres=# ¥dFd+ intdict
                            List of text search dictionaries
 Schema |  Name   |        Template         |  Init options   |       Description
--------+---------+-------------------------+-----------------+-------------------------
 public | intdict | public.intdict_template | absval = 'true' | dictionary for integers
(1 row)
```

3.5.4. ltree

ltree 型、lquery 型、ltxtquery 型がバイナリ入出力をサポートするようになりました。

例 79 ltree 型の情報

```
postgres=> SELECT typreceive, typsend FROM pg_type WHERE typname='ltree' ;
-[ RECORD 1 ]----------
typreceive | ltree_recv
typsend    | ltree_send
```

3.5.5. pageinspect

pageinspect エクステンションには以下の拡張が実装されました。

□ heap_tuple_infomask_flags 関数

この関数は t_infomask, t_infomask2 の値を情報を人が理解できる形式に変換します。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 80 heap_tuple_infomask_flags 関数

```
postgres=# SELECT t_infomask, t_infomask2, raw_flags, combined_flags
               FROM heap_page_items(get_raw_page('data1', 0)),
               LATERAL heap_tuple_infomask_flags(t_infomask, t_infomask2)
               m(raw_flags, combined_flags) ;
-[ RECORD 1 ]--+-----------------------------------------------------
t_infomask     | 2306
t_infomask2    | 2
raw_flags      | {HEAP_HASVARWIDTH,HEAP_XMIN_COMMITTED,HEAP_XMAX_INVALID}
combined_flags | {}
```

□ bt_metap 関数

bt_metap 関数の出力にはインデックスの重複データの状況を示す allequalimage 列が追加されました。

例 81 bt_metap 関数の出力

```
postgres=# SELECT * FROM bt_metap('idx1_data1') ;
-[ RECORD 1 ]-----------+-------
magic                   | 340322
version                 | 4
root                    | 3
level                   | 1
fastroot                | 3
fastlevel               | 1
oldest_xact             | 0
last_cleanup_num_tuples | -1
allequalimage           | t
```

□ bt_page_items 関数

bt_page_items 関数の出力には dead, htid, tids 列が追加されました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

例 82 bt_page_items 関数の出力

```
postgres=# SELECT * FROM bt_page_items('idx1_data1', 1) ;
-[ RECORD 1 ]----------------------
itemoffset | 1
ctid       | (0,1)
itemlen    | 16
nulls      | f
vars       | t
data       | 0b 00 80 01 00 00 00 00
dead       | f
htid       | (0,101)
tids       |
```

表 28 追加列

列名	説明	備考
dead	LP_DEAD ビットの値を示す boolean 値	
htid	各タプルの単一のヒープ TID 値を示す	
tids	TID の配列	

3.5.6. pg_stat_statements

pg_stat_statements エクステンションには実行計画と WAL の情報を出力する拡張が追加されました。

□ track_planning パラメーター

実行計画のトラッキングを行うかを指定するパラメーターです。デフォルト値は off です。このパラメーターが指定された場合、pg_stat_statements ビューには実行計画に関する情報が追加されます。

□ pg_stat_statements ビュー

pg_stat_statements ビューには以下の列が追加されました。

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

表 29 追加列

列名	データ型	説明
plans	bigint	実行計画作成回数
total_plan_time	double precision	実行計画作成の合計時間
min_plan_time	double precision	実行計画作成の最小時間
max_plan_time	double precision	実行計画作成の最大時間
mean_plan_time	double precision	実行計画作成の平均時間
stddev_plan_time	double precision	実行計画作成の標準偏差
wal_records	bigint	生成された WAL の合計レコード数
wal_fpi	bigint	生成された WAL の全ページ書き込みの回数
wal_bytes	numeric	生成された WAL の合計バイト数

既存の列名が以下のように変更されています。

表 30 変更列

PostgreSQL 12	PostgreSQL 13	説明
total_time	total_exec_time	SQL 文の合計実行時間
min_time	min_exec_time	SQL 文の最小実行時間
max_time	max_exec_time	SQL 文の最大実行時間
mean_time	mean_exec_time	SQL 文の平均実行時間
stddev_time	stddev_exec_time	SQL 文実行時間の標準偏差

例 83 pg_stat_statements ビューの検索

```
postgres=# SELECT queryid, plans, total_plan_time, min_plan_time
        FROM pg_stat_statements WHERE plans != 0;
       queryid        | plans |   total_plan_time    | min_plan_time
----------------------+-------+----------------------+---------------
  1809546128015582813 |     1 |               0.9533 |        0.9533
  1580010417825738145 |   900 |   29.697499999999998 |        0.0194
   726988806503882440 |     1 |               0.0889 |        0.0889
  6547867665616429211 |     1 |               0.3364 |        0.3364
  1518237015596383425 |     1 |               0.0522 |        0.0522
  6090531631178874043 |   900 |    51.09360000000001 |        0.0221
 -8944905077122340531 |     6 |  0.22340000000000002 |        0.0333
<<以下省略>>
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

□ SQL 文の正規化

FOR UPDATE 句が指定された SELECT 文と、指定されていない SELECT 文を分離して正規化するようになりました。従来は FOR UPDATE 句の有無以外同一の SELECT 文は同一とみなしていました。

3.5.7. postgres_fdw

postgres_fdw エクステンションには以下のオプションが追加されました。

□ password_required オプション

デフォルトでは一般ユーザーはパスワードを使用しない接続は拒否されます。ユーザー・マッピングの作成／変更時に password_required 属性を指定することでこの動作を変更することができるようになりました。このオプションを変更できるユーザーはスーパーユーザーだけです。

例 84 パスワード無しのユーザー・マッピング作成

```
postgres=# CREATE USER MAPPING FOR demo SERVER remhost1 OPTIONS (user 'demo') ;
CREATE USER MAPPING
```

例 85 一般ユーザーによる外部テーブルの作成とエラー

```
postgres=> CREATE FOREIGN TABLE data1(c1 INT, c2 VARCHAR(10)) SERVER remhost1 ;
CREATE FOREIGN TABLE
postgres=> SELECT * FROM data1 ;
ERROR:  password is required
DETAIL:  Non-superusers must provide a password in the user mapping.
postgres=>
```

オプション password_required を false に設定することでユーザー・マッピングに対するパスワード設定を回避します。

例 86 パスワード省略を許可する設定

```
postgres=# ALTER USER MAPPING FOR demo SERVER remhost1 OPTIONS
                (password_required 'false');
ALTER USER MAPPING
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

□ 認証オプション追加

USER MAPPING 作成時のオプションに sslkey と sslcert オプションを指定することができます。これらのオプションはスーパーユーザーが USER MAPPING を作成／変更している場合のみ指定できます。

3.5.8. bool_plperl

エクステンション bool_plperl が追加されました。このエクステンションは PL/Perl と一緒に使います。Bool 値の値を正しく Perl プログラムに転送するために使われます。

例 87 bool_plperl エクステンション

```
postgres=> CREATE EXTENSION plperl ;
CREATE EXTENSION
postgres=> CREATE EXTENSION bool_plperl ;
CREATE EXTENSION
postgres=> CREATE FUNCTION perl_and(bool, bool) RETURNS bool
           TRANSFORM FOR TYPE bool
           AS $$
               my ($a, $b) = @_;
               return $a && $b;
           $$ LANGUAGE plperl ;
CREATE FUNCTION
postgres=>
```

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

Hewlett Packard
Enterprise

参考にした URL

本資料の作成には、以下の URL を参考にしました。

- Release Notes
  https://www.postgresql.org/docs/13/release.html
- Commitfests
  https://commitfest.postgresql.org/
- PostgreSQL 13 Manual
  https://www.postgresql.org/docs/13/index.html
- Git
  git://git.postgresql.org/git/postgresql.git
- GitHub
  https://github.com/postgres/postgres
- Open source developer based in Japan（Michael Paquier さん）
  http://paquier.xyz/
- PostgreSQL 13 Open Items
  https://wiki.postgresql.org/wiki/PostgreSQL_13_Open_Items
- Qiita（ぬこ@横浜さん）
  http://qiita.com/nuko_yokohama
- PostgreSQL Deep Dive
  http://pgsqldeepdive.blogspot.jp/ （Satoshi Nagayasu さん）
- pgsql-hackers Mailing list
  https://www.postgresql.org/list/pgsql-hackers/
- PostgreSQL 13 のアナウンス
  https://www.postgresql.org/about/news/2077/
- PostgreSQL Developer Information
  https://wiki.postgresql.org/wiki/Development_information
- Slack - postgresql-jp
  https://postgresql-jp.slack.com/

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

Hewlett Packard
Enterprise

変更履歴

変更履歴

版	日付	作成者	説明
0.1	2020/04/09	篠田典良	内部レビュー版作成 レビュー担当（敬称略）: 高橋智雄 竹島彰子 （日本ヒューレット・パッカード株式会社）
1.0	2020/05/27	篠田典良	PostgreSQL 13 Beta 1 公開版に合わせて修正完了 レビュー担当（敬称略）: 永安悟史（アップタイム・テクノロジーズ合同会社）
1.1	2020/9/27	篠田典良	PostgreSQL 13 GA 公開にあわせて修正完了 変更点 - ANALYZE (BUFFERS) 文の動作変更 - pg_replication_slot ビューの列名変更 - pg_stat_replication ビューの列削除 - GUC enable_hashagg_disk 削除 - GUC enable_incrementalsort の名前変更 - GUC hash_mem_multiplier 追加 - GUC max_slot_wal_keep_size 追加 - GUC server_version の値変更 - GUC wal_keep_segments の名前変更 - Contrib earthdistance 設定変更 - pg_stat_statements.track_planning デフォルト値変更 - libpq 接続 SSL 最小バージョンのデフォルトを TLSv1.2 に変更 - 記述ミスの修正

以上

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.

© 2019-2020 Hewlett-Packard Enterprise Japan Co, Ltd.