Hewlett Packard Enterprise

2022 年 5 月 24 日

PostgreSQL 15 新機能検証結果（Beta 1）

日本ヒューレット・パッカード合同会社
篠田典良

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

Hewlett Packard
Enterprise

目次

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

© 2021-2022 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

本文書はオープンソース RDBMS である PostgreSQL 15 (15.0) Beta 1 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 14 (14.3) と PostgreSQL 15 (15.0) Beta 1 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。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

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 14.3 (比較対象) PostgreSQL 15 (15.0) Beta 1 (2022/05/16 21:15:02)
オペレーティング・システム	Red Hat Enterprise Linux 7 Update 8 (x86-64)
Configure オプション	--with-ssl=openssl --with-python --with-lz4 --with-zstd --with-llvm --with-icu

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・パッカード合同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は後日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。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

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

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

Hewlett Packard
Enterprise

2. PostgreSQL 15 における変更点概要

PostgreSQL 15 には 200 以上の新機能が追加されました。本章では代表的な新機能と利点の概要について説明します。新機能の詳細は「3. 新機能解説」で説明します。

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・クエリーの拡張

SELECT DISTINCT 文についてパラレル・クエリーが動作するようになりました。

□ 圧縮アルゴリズムの拡張

WAL の圧縮、ベースバックアップの圧縮に LZ4 や Zstandard が利用できるようになりました。pg_receivewal コマンドの圧縮メソッドも追加されました。

□ 稼働統計の拡張

従来 stats collector プロセスは UDP を使って統計情報を受け取っていましたが、共有メモリー領域を使う方法に変更されました。

2.2. 信頼性向上に関する新機能

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。

□ アーカイブ・ライブラリ

アーカイブログの取得を共有ライブラリで実施できるようになりました。リファレンス実装として Contrib モジュール basic_archive が追加されました。

□ チェックポイントのログ

log_checkpoints パラメーターのデフォルト値が on に変更されました。チェックポイント実行のログが出力されるようになります。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

2.3. 運用性向上に関する新機能

運用性を向上できる以下の機能が追加されました。

□ ロジカル・レプリケーションの拡張

ロジカル・レプリケーションは PostgreSQL 15 で大きく拡張されました。ロジカル・レプリケーションで特定の条件に合致するタプルのみレプリケーションできるようになりました。レプリケーションできるタプルの条件は CREATE PUBLICATION 文や ALTER PUBLICATION 文に WHERE 句を指定します。レプリケーション対象列を選択できるようになりました。CREATE PUBLICATION 文にテーブルに加えて列のリストを指定します。また指定した LSN の更新をスキップできるようになりました。

□ モニタリング機能の拡張

ロジカル・レプリケーション環境のサブスクリプション・ワーカー上で発生したエラーの状況を確認する pg_stat_subscription_stats ビューが追加されました。pg_stat_statements モジュールでは一時ファイルの I/O 時間や JIT に関する情報がモニタリングできるようになりました。

□ ログファイル形式

ログファイルを JSON 形式で出力できるようになりました。

2.4. プログラミングに関する新機能

SQL 文に以下の機能が追加されました。

□ JSON 関連

多くの RDBMS と同様の JSON 構文がサポートされます。コンストラクタ（JSON, JSON_OBJECT, JSON_ARRAY, JSON_ARRAYAGG, JSON_SCALAR 等）、検索関数 ()、関数（JSON_TABLE, JSON_QUERY, JSON_EXISTS, JSON_VALUE, JSON_SERIALIZE, ）、チェック構文（IS JSON, IS JSON SCALAR, IS JSON ARRAY, IS JSON WITH UNIQUE KEYS 等）が追加されました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ MERGE 文

テーブルの結合条件を元に、合致する場合と合致しない場合それぞれで INSERT / UPDATE / DELETE 処理を一括で実行できる MERGE 文がサポートされました。

2.5. 将来の新機能に対する準備

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。

□ テーブル・アクセス・メソッドの変更

ALTER TABLE 文と ALTER MATERIALIZED VIEW 文にテーブル・アクセス・メソッドを変更する構文が追加されました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

2.6. 非互換

PostgreSQL 15 は PostgreSQL 14 から以下の仕様が変更されました。

2.6.1. PUBLIC スキーマに対するアクセス権

従来は public スキーマには全ユーザー（PUBLIC）に対して CREATE 権限、USAGE 権限が付与されていました。PostgreSQL 15 では public スキーマに対するアクセス権限はデータベース所有者（pg_database_owner ロール）に限られます。public スキーマのオーナーも pg_database_owner に変更されました。この修正により CVE-2018-1058 以降に推奨された状態になります。

例 1 PUBLIC スキーマに対する権限

```
postgres=# CREATE USER demo PASSWORD 'demo' ;
CREATE ROLE
postgres=# ¥connect postgres demo
You are now connected to database "postgres" as user "demo".
postgres=> CREATE TABLE public.data1(c1 INT, c2 VARCHAR(10)) ;
ERROR:  permission denied for schema public
LINE 1: CREATE TABLE public.dataA(c1 INT, c2 VARCHAR(10)) ;
                    ^
postgres=> CREATE VIEW public.view1 AS SELECT 1 ;
ERROR:  permission denied for schema public
postgres=> CREATE SEQUENCE public.seq1 ;
ERROR:  permission denied for schema public
```

2.6.2. 排他的バックアップ・モード

非推奨だった排他的バックアップ・モードが削除されました。排他的バックアップはバックアップの複数同時実行を制限する方法です。排他的バックアップの開始は pg_start_backup 関数の exclusive パラメーターに true を設定（デフォルト）して実行します。排他的バックアップ・モードの削除に伴いオンライン・バックアップ開始／終了時に実行する関数名が変更されました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

表 4 関数名の変更

操作	PostgreSQL 14	PostgreSQL 15	備考
バックアップ開始	pg_start_backup	pg_backup_start	
バックアップ終了	pg_stop_backup	pg_backup_stop	

例 2 オンライン・バックアップの実行

```
postgres=# SELECT pg_backup_start('start online backup#1', false) ;
 pg_backup_start
-----------------
 0/4000028
(1 row)

postgres=# ¥! cp -r data backup
postgres=# SELECT pg_backup_stop(true) ;
NOTICE:  all required WAL segments have been archived
                              pg_backup_stop
-----------------------------------------------------------------------------
 (0/4000138,"START WAL LOCATION: 0/4000028 (file 000000010000000000000004)+
 CHECKPOINT LOCATION: 0/4000060                                            +
 BACKUP METHOD: streamed                                                   +
 BACKUP FROM: primary                                                      +
 START TIME: 2022-05-20 23:26:40 JST                                       +
 LABEL: start online backup#1                                              +
 START TIMELINE: 1                                                         +
 ","")
(1 row)
```

2.6.3. psql

psql コマンドは互換性に関する以下の変更がありました。

□ 接続先インスタンスのサポート・バージョン

PostgreSQL 9.2 より前のバージョンのサーバーに対する接続がサポート対象外になりました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ パスワード変更

¥password コマンドの挙動が変更されました。従来はログイン時に指定されたユーザーのパスワードを変更していましたが、SELECT CURRENT_USER 文の実行結果を元にユーザーを決定するようになりましたて。これにより SET SESSION AUTHORIZATION 文を実行した場合の挙動が変更されます。またプロンプトに変更対象のユーザー名が表示されるようになりました。

例 3 PostgreSQL 14 の動作

```
postgres=# SET SESSION AUTHORIZATION demo ;
SET
postgres=> ¥password
Enter new password: <<PASSWORD>> ← postgres ユーザーのパスワード
Enter it again: <<PASSWORD>> ← postgres ユーザーのパスワード
ERROR:  must be superuser to alter superuser roles or change superuser attribute
postgres=>
```

例 4 PostgreSQL 15 の動作

```
postgres=# SET SESSION AUTHORIZATION demo ;
SET
postgres=> ¥password
Enter new password for user "demo": <<PASSWORD>> ← demo ユーザーのパスワード
Enter it again: <<PASSWORD>> ← demo ユーザーのパスワード
postgres=>
```

□ 複数結果の表示

従来のバージョンでは psql コマンドは複数の結果セットが返された場合最後の結果のみを表示していました。PostgreSQL 15 ではすべての結果を出力します。従来のバージョンと同じ動作に変更する際は¥set SHOW_ALL_RESULTS off コマンドを実行します。

2.6.4. リテラル

数字から始まるリテラル文字列は従来は数字部分と文字列部分に分割されていましたが、PostgreSQL 15 ではエラーになります。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 5 PostgreSQL 14 の動作

```
postgres=> SELECT 123abc ;
abc
-----
 123
(1 row)
```

例 6 PostgreSQL 15 の動作

```
postgres=> SELECT 123abc ;
ERROR:  trailing junk after numeric literal at or near "123a"
LINE 1: SELECT 123abc;
               ^
```

2.6.5. ANALYZE

ANALYZE 文の実行には effective_io_concurrency ではなく、maintenance_io_concurrency を使うように変更されました。この仕様は PostgreSQL 14 にもバックポートされます。

2.6.6. Python 2

Python 2.7 のサポート切れに伴い、PL/Python における Python 2 のサポートが削除されました。configure コマンド実行時に Python 2 が拒否されます。

例 7 Python 2 のみの環境

```
$ ./configure --with-python
checking build system type... x86_64-pc-linux-gnu
<<<途中省略>>>
configure: using perl 5.16.3
checking for python3... no
checking for python... /bin/python
configure: using python 2.7.5 (default, Sep 26 2019, 13:23:47)
configure: error: Python version 2.7 is too old (version 3 or later is required)
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

また hstore_plpython2u、jsonb_plpython2u、ltree_plpython2u 等のエクステンションも削除されています。

2.6.7. date_bin

date_bin 関数のインターバルに負の値を指定できなくなりました。

例 8 date_bin 関数の実行

```
postgres=> SELECT date_bin('-2'::interval, timestamp '1970-01-01 01:00:00',
timestamp '1970-01-01 00:00:00') ;
ERROR:  stride must be greater than zero
```

2.6.8. pg_amcheck

pg_amcheck コマンドの--quiet オプション（短縮形 -q）は削除されました。この修正は PostgreSQL 14 にもバックポートされます。

2.6.9. pgcrypto

独自の組み込み暗号化アルゴリズムは削除され、OpenSSL が必須になりました。

2.6.10. pg_dump / pg_dumpall

PostgreSQL 9.2 より前のバージョンからのアップグレード機能を削除しました。また--no-synchronized-snapshots オプションが削除されました。

2.6.11. pg_upgrade

PostgreSQL 9.2 より前のバージョンからのアップグレード機能を削除しました。

2.6.12. postmaster

postmaster 起動時のオプション--forkboot は--forkaux に変更されました。

2.6.13. array_to_tsvector

array_to_tsvector 関数による空の語彙素を作成することを禁止されました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 9 array_to_tsvector 関数の実行

```
postgres=> SELECT array_to_tsvector(ARRAY['base','hidden','rebel', '']) ;
ERROR:  lexeme array may not contain empty strings
```

2.6.14. xml2

Contrib モジュール xml2 から xml_is_well_formed 関数が削除されました。

© 2021-2022 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やビューが変更されました。

表 5 追加されたシステムカタログ／ビュー

カタログ／ビュー名	説明
pg_ident_file_mappings	pg_ident.conf ファイルの内容を提供します。
pg_parameter_acl	ALTER SYSTEM SET 文の実行を許可する ACL 情報を提供します。
pg_publication_namespace	スキーマと PUBLICATION の対応をマッピングを格納します。
pg_stat_recovery_prefetch	リカバリー時の WAL プリフェッチ状況を提供します。
pg_stat_subscription_stats	サブスクリプション作成時に実行される初期データ同期の失敗情報を提供します。

表 6 列が追加されたシステムカタログ／ビュー

カタログ／ビュー名	追加列名	データ型	説明
pg_collation	colliculocale	text	ICU Collation 名
pg_constraint	confdelsetcols	smallint[]	外部キーの ON DELETE 句が指定された列
pg_database	daticucollate	text	ICU Collation 名
	datlocprovider	char	ICU Collation プロバイダ
	datcollversion	boolean	Collation バージョン
pg_index	indnullsnotdistinct	boolean	一意インデックスの NULL 値許容設定を示す
pg_publication_rel	prqual	pg_node_tree	レプリケーションを行うタプル条件
	prattrs	int2vector	レプリケーションを行う列番号

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

カタログ／ビュー名	追加列名	データ型	説明
pg_statistic_ext_data	stxdinherit	boolean	継承した子列の統計情報を含むかどうか
pg_stats_ext	inherited	boolean	継承した子列の統計情報を含むかどうか
pg_stats_ext_exprs	inherited	boolean	継承した子列の統計情報を含むかどうか
pg_subscription	subskiplsn	pg_lsn	スキップされた LSN
	subtwophasestate	char	Two Phase モードの状態を示す
	subdisableonerr	boolean	エラー発生時にレプリケーション無効化
information_schema.table_constraints	nulls_distinct	information_schema.yes_or_no	一意インデックスの NULL 値許容設定を示す

表 7 列が削除されたシステムカタログ／ビュー

カタログ名	削除列名	説明
pg_database	datlastsysoid	使用されていないため削除

表 8 出力内容が変更されたシステムカタログ／ビュー

カタログ／ビュー名	説明
pg_type	typcategory 列に内部用を示す'Z'が追加されました。
pg_collation	collcollate 列と collctype 列のデータ型が name から text に変更
pg_database	datcollate 列と datctype 列のデータ型が name から text に変更
pg_backend_memory_contexts	統計データ用メモリー領域（PgStat*）の情報が出力されるようになりました。

追加されたシステムカタログやビューから、主要なものについて詳細を以下に記載します。

□ pg_ident_file_mappings

pg_ident.conf ファイルの内容を SQL 文から検索するためのビューです。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

表 9 pg_ident_file_mappings カタログ

列名	データ型	説明
line_number	integer	pg_ident.conf ファイルの行番号
map_name	text	マップ名
sys_name	text	システム名
pg_username	text	PostgreSQL ユーザー名
error	text	エラー・メッセージ

例 10 pg_ident_file_mappings ビューの検索

```
postgres=# SELECT * FROM pg_ident_file_mapping ORDER BY 1 ;
line_number | map_name | sys_name | pg_username | error
-------------+----------+----------+-------------+-------
          43 | map1     | sysname1 | pgname1     |
          44 | map2     | sysname2 | pgname2     |
(2 rows)
```

□ pg_parameter_acl

ALTER SYSTEM 文によるパラメーター変更を許可するロールの情報が格納されます。

表 10 pg_parameter_acl カタログ

列名	データ型	説明
oid	oid	オブジェクト ID
parname	text	変更を許可するパラメーター名
paracl	aclitem[]	変更を許可するロール情報

□ pg_publication_namespace

CREATE PUBLICATION 文に FOR ALL TABLE IN SCHEMA 句を指定した場合にデータが格納されます。PUBLICATION と SCHEMA のマッピングを示します。

表 11 pg_publication_namespace カタログ

列名	データ型	説明
oid	oid	レコード ID
pnpubid	oid	PUBLICATION の oid
pnnspid	oid	pg_namespace カタログの oid への参照

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 11 pg_publication_namespace カタログの検索

```
postgres=> SELECT * FROM pg_publication_namespace ;
  oid  | pnpubid | pnnspid |
-------+---------+---------+
 16411 |   16410 |    2200 |
(1 row)
```

□ pg_stat_recovery_prefetch

リカバリ時にプリフェッチされたブロック統計を確認できます。

表 12 pg_stat_recovery_prefetch ビュー

列名	データ型	説明
stats_reset	timestamp with time zone	ビューがリセットされた日時
prefetch	bigint	バッファプールに存在しないためプリフェッチされたブロック数
hit	bigint	バッファプールにヒットしたブロック数
skip_init	bigint	初期化のためにプリフェッチされなかったブロック数
skip_new	bigint	存在しないためにプリフェッチされなかったブロック数
skip_fpw	bigint	フルページが含まれていたためにプリフェッチされなかったブロック数
skip_rep	bigint	既にプリフェッチされていたためスキップされたブロック数
wal_distance	integer	プリフェッチャが探しているバイト先
block_distance	integer	プリフェッチャが探しているブロック数
io_depth	integer	未完了のプリフェッチ数

□ pg_stat_subscription_stats

pg_stat_subscription_stats ビューにはロジカル・レプリケーションのサブスクリプション・ワーカーで発生したエラー情報が含まれます。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

表 13 pg_stat_subscription_stats ビュー

列名	データ型	説明
subid	oid	SUBSCRIPTION の oid
subname	name	SUBSCRIPTION 名
apply_error_count	bigint	適用エラー発生回数
sync_error_count	bigint	初期同期エラー発生回数
stats_reset	timestamp with time zone	統計情報のリセット日時

例 12 pg_stats_subscription_stats ビューの検索

```
postgres=# SELECT * FROM pg_stat_subscription_stats ;
-[ RECORD 1 ]-----+------
subid             | 16395
subname           | sub1
apply_error_count | 0
sync_error_count  | 23
stats_reset       |
```

□ メモリー関連ビューの参照

pg_backend_memory_contexts ビュー、pg_shmem_allocations ビューは SUPERUSER 属性を持つユーザーだけでなく pg_read_all_stats ロールに属するユーザーも参照できるようになりました。

例 13 pg_read_all_stats ロールの付与

```
postgres=# GRANT pg_read_all_stats TO demo ;
GRANT ROLE
postgres=# ¥connect postgres demo
You are now connected to database "postgres" as user "demo".
postgres=> SELECT COUNT(*) FROM pg_backend_memory_contexts ;
 count
-------
   118
(1 row)
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.1.2. ロジカル・レプリケーションの拡張

ロジカル・レプリケーションには以下の機能が追加されました。

□ 列指定レプリケーション

テーブル内の特定列のみレプリケーションを行うことができるようになりました。CREATE PUBLICATION 文または ALTER PUBLICATION 文のテーブル名に列リストを指定します。対象列の番号は pg_publication_rel カタログの prattrs 列に格納されます。

例 14 列指定レプリケーション

```
postgres=> CREATE TABLE repl1(c1 INT PRIMARY KEY, c2 INT, c3 VARCHAR(10)) ;
CREATE TABLE
postgres=> CREATE PUBLICATION pub1 FOR TABLE repl1(c1, c2) ;
CREATE PUBLICATION
postgres=> SELECT oid, prpubid, prrelid, prattrs FROM pg_publication_rel ;
-[ RECORD 1 ]--
oid     | 16394
prpubid | 16393
prrelid | 16388
prattrs | 1 2
```

UPDATE 文、DELETE 文を実行する場合は、列の指定にレプリカ・アイデンティティ（主キー等）をすべて含む必要があります。下記の例ではパブリケーションに主キーを含まない列を指定して UPDATE 文がエラーになっています。

例 15 UPDATE 文の失敗

```
postgres=> CREATE TABLE repl2(c1 INT PRIMARY KEY, c2 VARCHAR(10)) ;
CREATE TABLE
postgres=> CREATE PUBLICATION pub2 FOR TABLE repl2(c2) ;
CREATE PUBLICATION
postgres=> UPDATE repl2 SET c2='update' WHERE c1=100 ;
ERROR:  cannot update table "repl2"
DETAIL:  Column list used by the publication does not cover the replica
identity.
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ 行フィルター指定レプリケーション

CREATE PUBLICATION 文または ALTER PUBLICATION 文に WHERE 句を指定することでテーブル内の特定のタプルのみレプリケーションすることができます。レプリケーション条件はテーブル単位で指定します。WHERE 句の記述は括弧で囲む必要があります。

例 16 WHERE 句の指定

```
postgres=> ALTER PUBLICATION pub1 SET TABLE repl1 WHERE (c1 < 1000) ;
ALTER PUBLICATION
```

レプリケーション対象のテーブルに対して UPDATE 文や DELETE 文を実行するためには WHERE 句にレプリカ・アイデンティティ（主キー等）の一部を含んでいる必要があります。

例 17 レプリカアイデンティティを含まない WHERE 句

```
postgres=> CREATE TABLE repl3(c1 INT PRIMARY KEY, c2 VARCHAR(10)) ;
CREATE TABLE
postgres=> CREATE PUBLICATION pub3 FOR TABLE repl3 WHERE (c2='update') ;
CREATE PUBLICATION
postgres=> UPDATE repl3 SET c2='modify' WHERE c1=1000 ;
ERROR:  cannot update table "repl3"
DETAIL:  Column used in the publication WHERE expression is not part of the replica identity.
postgres=> DELETE FROM repl3 WHERE c1=1000 ;
ERROR:  cannot delete from table "repl3"
DETAIL:  Column used in the publication WHERE expression is not part of the replica identity.
```

WHERE 句に指定できる条件文は静的に決定される必要があります。このためユーザー定義関数や、MUTABLE 指定の関数等は指定できません。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 18 MUTABLE 関数の指定

```
postgres=> CREATE PUBLICATION pub4 FOR TABLE repl1 WHERE (c1 = random()) ;
ERROR:  invalid publication WHERE expression
LINE 1: ...EATE PUBLICATION pub4 FOR TABLE repl1 WHERE (c1 = random());
                                                             ^

DETAIL:  User-defined or built-in mutable functions are not allowed.
```

WHERE 句で指定された条件は pg_publication_rel カタログの prqual 列に変換されて格納されます。psql コマンドからは¥d コマンドや¥dRp+コマンドで確認できます。

例 19 ¥d コマンド

```
postgres=> ¥d repl1
                    Table "public.repl1"
 Column |         Type          | Collation | Nullable | Default
--------+-----------------------+-----------+----------+---------
 c1     | integer               |           | not null |
 c2     | character varying(10) |           |          |
Indexes:
    "repl1_pkey" PRIMARY KEY, btree (c1)
Publications:
    "pub1" WHERE (c1 < 1000)
```

□ スキーマ内の全テーブル指定

特定のスキーマ内に作成されたすべてのテーブルを PUBLICATION オブジェクトに登録する構文が利用できるようになりました。CREATE PUBLICATION 文（または ALTER PUBLICATION 文）に FOR ALL TABLES IN SCHEMA 句とスキーマ名を指定します。この構文は SUPERUSER 属性を持ったユーザーのみが実行できます。従来から利用できた FOR ALL TABLE 句の場合は全スキーマのテーブルが登録されていました。

構文

```
CREATE PUBLICATION publication_name FOR ALL TABLES IN SCHEMA schema_name
ALTER PUBLICATION publication_name ADD ALL TABLES IN SCHEMA schema_name
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 20 IN SCHEMA 句の指定

```
postgres=# CREATE PUBLICATION pub5 FOR ALL TABLES IN SCHEMA schema1 ;
CREATE PUBLICATION
postgres=# ¥dRp
                                List of publications
 Name |  Owner   | All tables | Inserts | Updates | Deletes | Truncates | …
------+----------+------------+---------+---------+---------+-----------+---
 pub5 | postgres | f          | t       | t       | t       | t         | f
(1 row)
```

この実装に伴い pg_publication_namespace カタログが追加されました。

□ LSN のスキップ

ALTER SUBSCRIPTION 文に SKIP 句を指定できるようになりました。SKIP 句には実行をキャンセルする LSN を指定します。ロジカル・レプリケーションの SUBSCRIPTION 側で同期エラーを発生させたトランザクションをスキップすることができます。

構文

```
ALTER SUBSCRIPTION subscription_name SKIP (LSN='lsn_value')
```

スキップした LSN の情報は pg_subscription カタログの subskiplsn 列で確認できます。また psql コマンドからは¥dRs+コマンドの出力に Skip LSN 項目が追加されました。

例 21 SKIP 句の指定

```
postgres=# ALTER SUBSCRIPTION sub1 SKIP (LSN = '0/30B51E0') ;
ALTER SUBSCRIPTION
postgres=# SELECT subskiplsn FROM pg_subscription ;
 subskiplsn
------------
 0/30B51E0
(1 row)
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ エラー発生時の SUBSCRIPTION 無効化

レプリケーション実行エラーが発生した場合に SUBSCRIPTION を無効化するオプション disable_on_error を指定できるようになりました。デフォルトは false で無効化は行われません。このオプション値を示すために pg_subscription カタログに subdisableonerr 列が追加されました。

例 22 CREATE SUBSCRIPTION 文の実行

```
postgres=# CREATE SUBSCRIPTION sub1 CONNECTION 'host=remsvr1, dbname=postgres'
PUBLICATION pub1 WITH (disable_on_error=true, two_phase=true) ;
NOTICE: created replication slot "sub1" on publisher
CREATE SUBSCRIPTION
```

□ TWO_PHASE オプションの追加

レプリケーション・プロトコルのコマンド CREATE_REPILICATION_SLOT が拡張され、TWO_PHASE オプションが追加されました。これにより PREPARE TRANSACTION、COMMIT PREPARED、ROLLBACK PREPARED 等がデコードできるようになりました。

3.1.3. パラレル・クエリーの拡張

SELECT DISTINCT 文がパラレル・クエリーで動作できるようになりました。

例 23 SELECT DISTINCT 文のパラレル化

```
postgres=> EXPLAIN SELECT DISTINCT c2 FROM data1 ;
                                QUERY PLAN
-------------------------------------------------------------------------------
 Unique  (cost=11614.55..11614.56 rows=1 width=6)
   ->  Sort  (cost=11614.55..11614.56 rows=2 width=6)
         Sort Key: c2
         ->  Gather  (cost=11614.33..11614.54 rows=2 width=6)
               Workers Planned: 2
               ->  HashAggregate  (cost=10614.33..10614.34 rows=1 width=6)
                     Group Key: c2
                     ->  Parallel Seq Scan on data1  (cost=0.00..9572.67 ro…
(8 rows)
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.1.4. WAL の圧縮

WAL に出力されるフルページ・イメージの圧縮アルゴリズムに LZ4 と Zstandard を指定できるようになりました。WAL 圧縮を行うためにはパラメーターwal_compression を変更します。従来の圧縮アルゴリズムである設定値 on は設定値 pglz のエイリアスになります。デフォルト値の変更は無く off のままです。

例 24 WAL 圧縮設定

```
postgres=# SELECT name, setting, vartype, enumvals FROM pg_settings
                WHERE name='wal_compression' ;
-[ RECORD 1 ]---------------
name     | wal_compression
setting  | off
vartype  | enum
enumvals | {pglz,lz4,zstd,on,off}
```

LZ4 または Zstandard を利用するためには configure コマンド実行時に--with-lz4 および--with-zstd を指定してコンパイルされている必要があります。

3.1.5. アーカイブ・ライブラリ

WAL ファイルのアーカイブを作成する方法に従来のコマンド（archive_command パラメーター）だけでなく、共有ライブラリを使用できるようになりました。アーカイブ用ライブラリ名を指定するパラメーターarchive_library が追加されています。archive_library を設定しない場合、従来の archive_command も引き続き利用できます。

標準のアーカイブ・ライブラリとして Contrib モジュール basic_archive が追加されています。この Contrib モジュールは CREATE EXTENSION 文の実行は不要です。アーカイブ先のディレクトリとして basic_archive.archive_directory を指定することができます。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 25 アーカイブ・ライブラリの指定

```
postgres=# SELECT * FROM pg_settings WHERE name LIKE 'archive_library' ;
-[ RECORD 1 ]---+-----------------------------------------------------------
name            | archive_library
setting         | basic_archive
unit            |
category        | Write-Ahead Log / Archiving
short_desc      | Sets the library that will be called to archive a WAL file.
extra_desc      | An empty string indicates that "archive_command" should be
used.
context         | sighup
vartype         | string
source          | configuration file
min_val         |
max_val         |
enumvals        |
boot_val        |
reset_val       | basic_archive
sourcefile      | /home/postgres/data/postgresql.conf
sourceline      | 249
pending_restart | f

postgres=# SHOW basic_archive.archive_directory ;
-[ RECORD 1 ]------------------+--------------------
basic_archive.archive_directory | /home/postgres/arch
```

archive_library と archive_command を両方指定した場合には archive_library が使用されます。

3.1.6. グローバル・ロケール・プロバイダー

ICU をグローバル・ロケール・プロバイダーとして使用できるようになりました。データベース・クラスター全体またはデータベースのデフォルトのロケール・プロバイダーに ICU を使用できるようになります。この機能を利用する場合は--with-icu オプションを指定してコンパイルされたバイナリを使用する必要があります。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

initdb コマンド、createdb コマンドにはロケール・プロバイダ名を指定する--locale-provider オプションが指定できます。このオプションには libc または icu を指定します。また--icu-locale オプションには ICU ロケール名を指定できます。

例 26 initdb コマンドの指定

```
$ initdb --locale-provider=icu --icu-locale=ja --encoding=utf8 -D data
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with this locale configuration:
  provider:    icu
  ICU locale:  ja
  LC_COLLATE:  ja_JP.utf8
  LC_CTYPE:    ja_JP.utf8
  LC_MESSAGES: ja_JP.utf8
  LC_MONETARY: ja_JP.utf8
  LC_NUMERIC:  ja_JP.utf8
  LC_TIME:     ja_JP.utf8
initdb: could not find suitable text search configuration for locale
"ja_JP.utf8"
<<以下省略>>
```

CREATE DATABASE 文のオプションにはロケール・プロバイダーを指定する LOCALE_PROVIDER とロケール名を指定する ICU_LOCALE が追加されました。

例 27 CREATE DATABASE 文の実行

```
postgres=# CREATE DATABASE icudb1 LOCALE_PROVIDER=icu ICU_LOCALE=en
                TEMPLATE=template0 ;
CREATE DATABASE
postgres=# SELECT datname, daticulocale FROM pg_database
                WHERE datname='icudb1' ;
 datname | daticulocale
---------+--------------
 icudb1  | en
(1 row)
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

psql コマンドでは¥l コマンドの出力に ICU Locale と Locale Provider が出力されます。

例 28 データベース一覧の表示

```
postgres=> ¥l
                                                List of databases
   Name    |  Owner   | Encoding |  Collate   |   Ctype    | ICU Locale | Locale Provider…
-----------+----------+----------+------------+------------+------------+---------------- …
postgres   | postgres | UTF8     | en_US.utf8 | en_US.utf8 | ja         | icu             …
template0 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | ja         | icu             …
template1 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | ja         | icu             …
(3 rows)
```

3.1.7. 統計情報

stats collector プロセスは従来 UDP プロトコルにより統計情報を受け取り、一時ファイルに定期的に書き出していました。PostgreSQL 15 では統計情報は共有メモリーに保存されるようになります。pg_backend_memory_contexts ビューを検索すると、統計情報用の共有メモリー領域の情報を取得できます。

例 29 統計情報用の共有メモリー

```
postgres=# SELECT name, total_bytes, free_bytes, used_bytes FROM
               pg_backend_memory_contexts WHERE name LIKE 'PgStat%' ;
          name          | total_bytes | free_bytes | used_bytes
------------------------+-------------+------------+------------
 PgStat Shared Ref Hash |        7224 |        680 |       6544
 PgStat Shared Ref      |        8192 |       3408 |       4784
 PgStat Pending         |        8192 |       7808 |        384
(3 rows)
```

3.1.8. GiST インデックス

GiST インデックスには以下の拡張機能が実装されました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ インデックス統計

SP-GiST インデックスに対するアクセスが pg_stat_*ビューにカウントされるようになりました。

□ BOOL 型

BOOL 型に対してインデックスを作成することができるようになりました。

例 30 BOOL 型に対する GiST インデックス作成

```
postgres=> CREATE EXTENSION btree_gist ;
CREATE EXTENSION
postgres=> CREATE TABLE gist1(c1 INT, c2 BOOL, c3 VARCHAR(10)) ;
CREATE TABLE
postgres=> CREATE INDEX idx1_gist1 ON gist1 USING gist (c2) ;
CREATE INDEX
```

3.1.9. 待機イベント

以下の待機イベントが追加されました。

表 14 追加された待機イベント

イベント名	タイプ	説明
ArchiveCleanupCommand	IPC	archive_cleanup_command コマンド完了待ち
ArchiveCommand	IPC	archive_command コマンド完了待ち
BaseBackupSync	IO	ベースバックアップのストレージ同期待ち
BaseBackupWrite	IO	ベースバックアップ書き込み待ち
RecoveryEndCommand	IPC	recovery_end_command コマンド完了待ち
RestoreCommand	IPC	restore_command コマンド完了待ち
VersionFileWrite	IO	データベース作成時のバージョンファイル書き込み待ち
VacuumTruncate	Timeout	テーブル終端の空ページ削除のための排他ロック待ち

待機イベント PgStatMain、LogicalChangesRead、LogicalChangesWrite、LogicalSubxactRead、LogicalSubxactWrite は削除されました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.1.10. ロール

事前定義ロール pg_checkpointer が追加されました。このロール保持者は CHECKPOINT 文を実行することができます。

例 31 pg_checkpointer ロールの付与

```
postgres=# CREATE USER check1 PASSWORD 'check1' ;
CREATE ROLE
postgres=# ¥connect postgres check1
You are now connected to database "postgres" as user "check1".
postgres=> CHECKPOINT ;
ERROR:  must be superuser or have privileges of pg_checkpointer to do CHECKPOINT
postgres=> ¥connect postgres postgres
You are now connected to database "postgres" as user "postgres".
postgres=# GRANT pg_checkpointer TO check1 ;
GRANT ROLE
postgres=# ¥connect postgres check1
You are now connected to database "postgres" as user "check1".
postgres=> CHECKPOINT ;
CHECKPOINT
```

3.1.11. libpq

Libpq API には以下の拡張が実装されました。

□ 環境変数 PG_COLORS

ターミナル出力の色指定項目に note を指定できるようになりました。

□ PQsslAttribute

PQsslAttribute(NULL, "library") 関数の実行により、使用中のライブラリ名を取得できるようになりました。

□ PQcancel

PQcancel API の実行に接続文字列の tcp_user_timeout と keepalives が利用されるようになりました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ root 所有の秘密鍵

root 所有の SSL 秘密鍵の使用が許可されるようになりました。秘密鍵は現在のユーザーまたは root のいずれかが所有できます。

3.1.12. カスタム WAL リソース・マネージャー

パラメーターshared_preload_libraries に独自の WAL リソース・マネージャーを登録できるようになりました。現状の WAL リソース・マネージャーを確認するための pg_get_wal_resource_managers 関数が追加されました。

例 32 WAL リソース・マネージャーの確認

```
postgres=# SELECT * FROM pg_get_wal_resource_managers() ;
 rm_id |      rm_name      | rm_builtin
-------+-------------------+------------
     0 | XLOG              | t
     1 | Transaction       | t
     2 | Storage           | t
     3 | CLOG              | t
<<以下省略>>
```

3.1.13. フック

以下のフックが追加されました。

□ 文字列オブジェクトのアクセス・フック

文字列オブジェクトのアクセス・フックが追加されました。GUC の文字列アクセス時に実行されます。

例 33 フックの定義（src/include/catalog/objectaccess.h）

```
typedef void (*object_access_hook_type_str) (ObjectAccessType access,
                                             Oid classId,
                                             const char *objectStr,
                                             int subId,
                                             void *arg);
extern PGDLLIMPORT object_access_hook_type_str object_access_hook_str;
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ 共有メモリー要求・フック

拡張モジュールが共有メモリーを取得する際に_PG_init 関数内から指定する必要があります。

例 34 フックの定義 (src/include/miscadmin.h)

```
typedef void (*shmem_request_hook_type) (void);
extern PGDLLIMPORT shmem_request_hook_type shmem_request_hook;
extern PGDLLIMPORT bool process_shmem_requests_in_progress;
```

□ オブジェクト・アクセス・フック

ALTER TABLE SET {LOGGED, UNLOGGED, ACCESS METHOD} 文の実行により、オブジェクト・アクセス・フックが実行されるようになりました。

3.1.14. カスタム・スキャン

カスタム・スキャン・プロバイダーがプロジェクションのサポートを選択できるようになりました。以前のバージョンではすべてのカスタム・スキャン・プロバイダーはプロジェクションのサポートが必要でした。CUSTOMPATH_SUPPORT_PROJECTION マクロで制御を行います。

3.1.15 ビルド

ソースコードからビルドを行う際に指定する Makefile のターゲット名が追加されました。以下のターゲットはドキュメントのビルドやインストールを行いません。

- world-bin
- install-world-bin

この仕様は旧バージョンにもバックポートされます。

3.1.16. Dynamic Shared Memory

DSM をシングル・ユーザー・モードでも利用できるようになりました。

© 2021-2022 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. JSON 構文の拡張

JSON 関連機能が大きく拡張されました。

□ 形式のチェック

文字列や bytea 型のデータが JSON 形式であるかをチェックできる構文がサポートされました。

表 15 IS JSON 構文

構文	説明
data IS JSON [VALUE]	JSON 形式である
data IS JSON OBJECT	JSON オブジェクト形式である
data IS JSON ARRAY	JSON 配列形式である
data IS JSON SCALAR	JSON スカラー形式である
data IS JSON WITH | WITHOUT UNIQUE KEYS	一意なキーを持つ（持たない）JSON 形式である

例 35 IS JSON 構文

```
postgres=> SELECT '{ "Product": "PostgreSQL"}' IS JSON ;
 ?column?
----------
 t
(1 row)
postgres=> SELECT '["Array#1", "Array#2"]' IS JSON ARRAY ;
 ?column?
----------
 t
(1 row)
```

□ JSON データ生成関数

JSON データを生成するためのコンストラクタとして一般的な構文が追加されました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

表 16 JSON データ生成関数

関数	説明
JSON	テキストから JSON を生成
JSON_SCALAR	SQL データから JSON スカラーを生成
JSON_OBJECT	SQL データや JSON データから JSON オブジェクトを生成
JSON_OBJECTAGG	提供されたデータを JSON オブジェクトに集約
JSON_ARRAY	JSON 配列を生成
JSON_ARRAYAGG	JSON 配列を集約

例 36 JSON データ生成関数

```
postgres=> SELECT JSON('{ "Val" : 100 }'::bytea FORMAT JSON ENCODING UTF8) ;
       json
------------------
 { "Val" : 100 }
(1 row)

postgres=> SELECT JSON_OBJECT('a': 2 + 3) ;
 json_object
-------------
 {"a" : 5}
(1 row)
postgres=> SELECT JSON_ARRAY('Array1', 'Array2') ;
      json_array
----------------------
 ["Array1", "Array2"]
(1 row)
postgres=> SELECT JSON_ARRAY('a',  NULL, 'b' ABSENT ON NULL) ;
 json_array
------------
 ["a", "b"]
(1 row)
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 36 JSON データ生成関数<続>

```
postgres=> SELECT JSON_ARRAYAGG(val ORDER BY val DESC) FROM
                generate_series(1, 5) val ;
  json_arrayagg
------------------
 [5, 4, 3, 2, 1]
(1 row)

postgres=> SELECT JSON_SCALAR('ABC') ;
 json_scalar
-------------
 "ABC"
(1 row)
```

□ JSON 関数

JSON データの検索に関する以下の関数が追加されました。

表 17 IS JSON 関数

関数名	説明
JSON_TABLE	JSON データを検索し、結果をリレーショナル・ビューに変換
JSON_EXISTS	提供された JSON パスが SQL/JSON アイテムを返すかチェック
JSON_QUERY	JSON データから SQL/JSON 配列またはオブジェクトを抽出
JSON_VALUE	提供された JSON データからスカラーに変換
JSON_SERIALIZE	SQL/JSON データを文字列またはバイナリに変換

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 37 JSON 関数

```
Postgres=> SELECT JSON_EXISTS(jsonb '{"key1": [1,2,3]}',
                'strict $.key1[*] ? (@ > 2)') ;
 json_exists
-------------
 t
(1 row)

postgres=> SELECT JSON_SERIALIZE('{"foo": "bar", "baz": [1, 2]}' RETURNING
bytea) ;
                        json_serialize
------------------------------------------------------------
 ¥x7b22666f6f223a2022626172222c202262617a223a205b312c20325d7d
(1 row)

postgres=> SELECT js, JSON_QUERY(js, 'lax $[*]') AS "without",
        JSON_QUERY(js, 'lax $[*]' WITH WRAPPER)  AS "with uncond",
        JSON_QUERY(js, 'lax $[*]' WITH CONDITIONAL WRAPPER) AS "with cond"
  FROM (VALUES (jsonb '[]'), ('[1]'), ('[[1,2,3]]'),   ('[{"a": 1}]'), ('[1,
null, "2"]')) foo(js) ;
       js       |  without  |  with uncond   |   with cond
----------------+-----------+----------------+---------------
 []             |           |                |
 [1]            | 1         | [1]            | [1]
 [[1, 2, 3]]    | [1, 2, 3] | [[1, 2, 3]]    | [1, 2, 3]
 [{"a": 1}]     | {"a": 1}  | [{"a": 1}]     | {"a": 1}
 [1, null, "2"] |           | [1, null, "2"] | [1, null, "2"]
(5 rows)
```

上記関数は jsonb 型に対してのみ利用できます。json 型に対して実行すると下記のエラーが発生します。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 38 json 型に対する実行

```
postgres=> SELECT JSON_VALUE(NULL FORMAT JSON, '$') ;
ERROR:  JSON_VALUE() is not yet implemented for json type
LINE 1: SELECT JSON_VALUE(NULL FORMAT JSON, '$') ;
               ^
HINT:  Try casting the argument to jsonb
```

3.2.2. NUMERIC データ型

NUMERIC 型のスケールに負の数を指定できるようになりました。負数を指定された場合には指定された桁で四捨五入されます。またスケールに指定できる値が-1,000 から 1,000 まで拡張されました。

例 39 NUMERIC 型の桁数

```
postgres=> CREATE TABLE type1( c1 NUMERIC(5, -2) ) ;
CREATE TABLE
postgres=> INSERT INTO type1 VALUES (12345.678) ;
INSERT 0 1
postgres=> INSERT INTO type1 VALUES (1234567.89) ;
INSERT 0 1
postgres=> INSERT INTO type1 VALUES (12345678.901) ;
ERROR:  numeric field overflow
DETAIL:  A field with precision 5, scale -2 must round to an absolute value
less than 10^7.
postgres=> INSERT INTO type1 VALUES (-1234567.89) ;
INSERT 0 1
postgres=> SELECT * FROM type1 ;
    c1
----------
    12300
  1234600
 -1234600
(3 rows)
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.2.3. ALTER DATABASE

コレーションのバージョンを追跡する構文が ALTER DATABASE 文に追加されました。

構文

```
ALTER DATABASE database_name REFRESH COLLATION VERSION
```

コレーションのバージョンを取得する新しい関数 pg_database_collation_actual_version と、pg_database ビューに datcollversion 列が追加されました。

3.2.4. ALTER TABLE

テーブルのアクセス・メソッドを変更する ALTER TABLE 文に SET ACCESS METHOD 句を指定できるようになりました。SET ACCESS METHOD 句は ALTER MATERIALIZED VIEW 文にも指定できます。

構文

- ALTER TABLE *table_name* SET ACCESS METHOD *method_name*
- ALTER MATERIALIZED VIEW *view_name* SET ACCESS METHOD *method_name*

例 40 ALTER TABLE 文の実行

```
postgres=> ALTER TABLE data1 SET ACCESS METHOD heap ;
ALTER TABLE
```

3.2.5. COPY

COPY 文には以下の拡張が追加されました。

□ COPY TO 文

COPY TO 文のオプションで format 'text'と header 'true'を同時に使用できるようになりました。同時に file_fdw エクステンションでもテキスト形式のフォーマットでヘッダを有効にできるようになりました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 41 テキスト・フォーマットとヘッダ

```
postgres=# CREATE EXTENSION file_fdw ;
CREATE EXTENSION
postgres=# CREATE SERVER textsvr FOREIGN DATA WRAPPER file_fdw ;
CREATE SERVER
postgres=# CREATE FOREIGN TABLE text1(c1 INT, c2 VARCHAR(10)) SERVER textsvr
OPTIONS (filename '/tmp/data.txt', format 'text', header 'true') ;
CREATE FOREIGN TABLE
postgres=# COPY data1 TO '/home/postgres/data1.txt' (FORMAT text, HEADER true) ;
COPY 100
postgres=# ¥! cat data1.txt
c1      c2
1       data1
<<以下省略>>
```

□ COPY FROM 文

オプション HEADER に match を指定できるようになりました。このオプションを指定すると COPY FROM 文実行時にヘッダ行と列名が一致していることをチェックします。下記の例では CSV ファイルの先頭行とテーブルの列名が異なるためエラーになっています。

例 42 ヘッダのチェック

```
postgres=# ¥! cat /tmp/csvlog1.csv
c1,c3
1,data1
2,data2
3,data3
postgres=# COPY csvlog1 FROM '/tmp/csvlog1.csv' WITH
                (format 'csv', header 'match') ;
ERROR:  column name mismatch in header line field 2: got "c3", expected "c2"
CONTEXT:  COPY csvlog1, line 1: "c1,c3"
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.2.6. CLUSTER

パーティション・テーブルに対して CLUSTER 文を実行できるようになりました。

例 43 パーティション・テーブルのクラスター化

```
postgres=> CREATE TABLE part1 (c1 INT PRIMARY KEY, c2 VARCHAR(10))
                PARTITION BY LIST(c1) ;
CREATE TABLE
postgres=> CLUSTER part1 USING part1_pkey ;
CLUSTER
```

3.2.7. CREATE DATABASE

CREATE DATABASE 文には以下の拡張が実装されました。

□ ICU ロケール

CREATE DATABASE 文にはロケール・プロバイダーを指定する LOCALE_PROVIDER 句、ICU ロケールを指定する ICU_LOCALE 句、コレーションのバージョンを指定する COLLATION_VERSION 句が追加されました。

例 44 COLLATION_VERSION の指定

```
postgres=# CREATE DATABASE icudb1 LOCALE_PROVIDER=icu ICU_LOCALE=en
                COLLATION_VERSION='58.0.6.50' TEMPLATE=template0 ;
CREATE DATABASE
```

□ STRATEGY 句

データベース作成方法を指定する STRATEGY 句が追加されました。WAL_LOG を指定すると新しいデータベースはテンプレートからブロック単位にコピーされ、各ブロックの情報は WAL に出力されます。テンプレートが小さい場合には効果的な方法であり、デフォルト値です。旧バージョンの動作を実行するには FILE_COPY を指定します。

例 45 STRATEGY 句のの指定

```
postgres=# CREATE DATABASE strategydb1 STRATEGY WAL_LOG ;
CREATE DATABASE
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

createdb コマンドにも STRATEGY 句に対応するオプション--strategy が追加されました。

□ OID 句

作成するデータベースの OID を指定できるようになりました。OID は 16384 以上の空き番号を指定できます。

例 46 OID 句のの指定

```
postgres=# CREATE DATABASE oiddb1 OID=17000 ;
CREATE DATABASE
```

3.2.8. CREATE TABLE

CREATE TABLE 文および ALTER TABLE 文における外部キーの定義の ON DELETE SET NULL 句と SET DEFAULT 句に列リストを指定できるようになりました。旧バージョンでは列指定はできませんでした。これに伴い pg_constraint カタログに confdelsetcols 列が追加されました。

例 47 ON DELETE SET NULL 句の指定

```
postgres=> CREATE TABLE pktable1 (
   tid INT,
   id INT,
   name1 VARCHAR(10), PRIMARY KEY (tid, id)) ;
CREATE TABLE
postgres=> CREATE TABLE fktable1 (
   tid INT,
   id INT,
   fk_id_del_set_null INT,
   fk_id_del_set_default INT DEFAULT 0,
   FOREIGN KEY (tid, fk_id_del_set_null) REFERENCES pktable1
        ON DELETE SET NULL (fk_id_del_set_null),
   FOREIGN KEY (tid, fk_id_del_set_default) REFERENCES pktable1
        ON DELETE SET DEFAULT (fk_id_del_set_default)
) ;
CREATE TABLE
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.2.9. CREATE UNIQUE INDEX

一意インデックスの属性に NULLS DISTINCT と NULLS NOT DISTINCT を指定できるようになりました。NULLS DISTINCT 句を指定した一意インデックスには NULL 値が複数格納できます。何も指定しない場合は NULLS DISTINCT 指定がデフォルトです。NULLS NOT DISTINCT 句が指定されたインデックスには NULL 値は複数格納できません。この属性を示すために pg_index カタログに indnullsnotdistinct 列が追加されました。

構文

- CREATE UNIQUE INDEX *index_name* ON … NULLS [NOT] DISTINCT
- CONSTRAINT *constraint_name* NULLS [NOT] DISTINCT

例 48 UNIQUE INDEX の指定

```
postgres=> CREATE TABLE data1(c1 INT, c2 INT, c3 VARCHAR(10)) ;
CREATE TABLE
postgres=> CREATE UNIQUE INDEX idx1_data1 ON data1(c1) NULLS DISTINCT ;
CREATE INDEX
postgres=> CREATE UNIQUE INDEX idx2_data1 ON data1(c2) NULLS NOT DISTINCT ;
CREATE INDEX
postgres=> INSERT INTO data1 VALUES (1, 1, 'data1') ;
INSERT 0 1
postgres=> INSERT INTO data1 VALUES (2, NULL, 'data2') ;
INSERT 0 1
postgres=> INSERT INTO data1 VALUES (NULL, 3, 'data3') ;
INSERT 0 1
postgres=> INSERT INTO data1 VALUES (NULL, NULL, 'data4') ;
ERROR:  duplicate key value violates unique constraint "idx2_data1"
DETAIL:  Key (c2)=(null) already exists.
postgres=> INSERT INTO data1 VALUES (NULL, 5, 'data5') ;
INSERT 0 1
```

CREATE TABLE 文と ALTER TABLE 文の一意制約にも同様の指定を行うことができます。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 49 UNIQUE 制約の指定

```
postgres=> CREATE TABLE const1(c1 INT UNIQUE NULLS NOT DISTINCT, c2 INT) ;
CREATE TABLE
postgres=> ALTER TABLE const1 ADD CONSTRAINT uniq1 UNIQUE NULLS DISTINCT (c2) ;
ALTER TABLE
```

3.2.10. CREATE SEQUENCE

WAL の出力を抑制するために CREATE SEQUENCE 文に UNLOGGED 句を指定できるようになりました。CREATE UNLOGGED SEQUENCE 文で作成されたシーケンスは pg_class カタログの relpersistence 列が'u'になります。また ALTER SEQUENCE SET LOGGED 文または ALTER SEQUENCE SET UNLOGGED 文でシーケンスの属性を変更することができます。

例 50 UNLOGGED シーケンスの作成

```
postgres=> CREATE UNLOGGED SEQUENCE seq1 ;
CREATE SEQUENCE
postgres=> ¥d seq1
                      Unlogged sequence "public.seq1"
  Type  | Start | Minimum |       Maximum       | Increment | Cycles? | Cache
--------+-------+---------+---------------------+-----------+---------+-------
 bigint |     1 |       1 | 9223372036854775807 |         1 | no      |     1

postgres=> SELECT relname, relpersistence FROM pg_class WHERE relname = 'seq1' ;
 relname | relpersistence
---------+----------------
 seq1    | u
(1 row)

postgres=> ALTER SEQUENCE seq1 SET LOGGED ;
ALTER SEQUENCE
```

インスタンスがクラッシュすると UNLOGGED SEQUENCE は値がリセットされ、初期値に戻ります。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.2.11. CREATE VIEW

CREATE VIEW 文のオプションに security_invoker が追加されました。デフォルトではビューの元となるテーブルへのアクセスはビュー所有者の権限で実施されます。ビューの元となるテーブルに Row Level Security のポリシーが存在する場合、security_invoker オプションを設定すると、ビューの所有者ではなくビューのユーザーのポリシーが適用されます。このオプションは ALTER VIEW 文にも指定できます。

構文

- CREATE VIEW *view_name* WITH (security_invoker = true|false) WITH *sql_clause*
- ALTER VIEW *view_name* SET (security_invoker = true|false)
- ALTER VIEW *view_name* RESET (security_invoker)

例 51 security_invoker 指定

```
postgres=> CREATE VIEW view1 WITH(security_invoker) AS SELECT * FROM data1 ;
CREATE VIEW
```

3.2.12. EXPLAIN

JSON 形式で BUFFERS, VERBOSE を指定した場合に一時ファイルに対する I/O 時間が出力されるようになりました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 52 一時ファイル I/O 時間の出力

```
postgres=# SET track_io_timing = on ;
SET
postgres=# EXPLAIN (ANALYZE, BUFFERS, VERBOSE, FORMAT JSON)
                SELECT * FROM data1 ;
                     QUERY PLAN
-----------------------------------------------------
 [                                                  +
<<途中省略>>
       "I/O Write Time": 0.000,      +
       "Temp I/O Read Time": 0.000, +
       "Temp I/O Write Time": 0.000 +
    }
<<以下省略>>
```

3.2.13. GRANT

ALTER SYSTEM 文による特定のパラメーターの変更許可をロールに付与できるようになりました。

構文

```
GRANT {ALTER SYSTEM | SET} ON PARAMETER parameter_name TO role_spec
REVOKE {ALTER SYSTEM | SET} ON PARAMETER parameter_name FROM role_spec
```

例 53 GRANT ALTER SYSTEM 文の実行

```
postgres=# GRANT ALTER SYSTEM ON PARAMETER log_statement TO demo ;
GRANT
postgres=# SELECT * FROM pg_parameter_acl ;
-[ RECORD 1 ]-------------------------------------
oid     | 16385
parname | log_statement
paracl  | {postgres=sA/postgres,demo=A/postgres}

postgres=# ¥connect postgres demo
You are now connected to database "postgres" as user "demo".
postgres=> ALTER SYSTEM SET log_statement = 'all' ;
ALTER SYSTEM
```

3.2.14. MERGE

結合条件からテーブルに対する INSERT / DELETE / UPDATE 文を同時に実行する MERGE 文がサポートされました。結合条件に一致したタプルは WHEN MATCHED 句に更新（UPDATE）または削除（DELETE）または何もしない（DO NOTHING）を指定します。追加条件を指定することもできます。条件に一致しないタプルの動作は WHEN NOT MATCHED 句に指定します。

構文

```
MERGE INTO target_table [AS alias]
USING source_table [AS alias]
ON join_clause
WHEN MATCHED [AND condition] THEN
   UPDATE SET … | DELETE | DO NOTHING
WHEN NOT MATCHED
   INSERT VALUES … | DO NOTHING
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 54 MERGE 文の実行

```
postgres=> MERGE INTO dst1 AS d USING src1 AS s ON d.c1 = s.c1
                WHEN MATCHED THEN
                        UPDATE SET c2 = s.c2
                WHEN NOT MATCHED THEN
                        INSERT VALUES (s.c1, s.c2) ;
MERGE 2
postgres=> MERGE INTO dst1 AS d USING src1 AS s ON d.c1 = s.c1
                WHEN MATCHED AND s.c1 < 1000 THEN
                        DELETE
                WHEN NOT MATCHED THEN
                        DO NOTHING ;
MERGE 2
```

3.2.15. VACUUM

VACUUM VERBOSE 文の実行ログは大きく変更されました。平均読み込みレート、バッファ使用状況、WAL 出力情報等が追加されています。Pg_class.relfrozenxid と pg_class.relminmxid がどのように進んだかについての詳細も報告されるようになりました。

例 55 MERGE 文の実行

```
postgres=> VACUUM VERBOSE data1 ;
INFO:  vacuuming "postgres.public.data1"
INFO:  finished vacuuming "postgres.public.data1": index scans: 1
pages: 0 removed, 5406 remain, 5406 scanned (100.00% of total)
tuples: 500000 removed, 500000 remain, 0 are dead but not yet removable
removable cutoff: 757, which was 0 XIDs old when operation ended
<<途中省略>>>
avg read rate: 0.000 MB/s, avg write rate: 0.068 MB/s
buffer usage: 18975 hits, 0 misses, 1 dirtied
WAL usage: 18953 records, 0 full page images, 4007392 bytes
system usage: CPU: user: 0.11 s, system: 0.00 s, elapsed: 0.11 s
INFO:  vacuuming "postgres.pg_toast.pg_toast_32787"
<<以下省略>>
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.2.16. 関数

以下の関数が追加／拡張されました。

□ MAX / MIN

xid8 型に対する MAX 関数、MIN 関数が実装されました。

□ レプリケーション関連

レプリケーション情報を取得する以下の関数が追加されました。これらの関数は SUPERUSER 属性または pg_monitor ロールを持つユーザーが実行できます。

表 18 追加された関数

関数名	説明
pg_ls_logicalmapdir	pg_logical/mappings ディレクトリの情報を返す
pg_ls_logicalsnapdir	pg_logical/snapshots ディレクトリの情報を返す
pg_ls_replslotdir	pg_replslot/<<スロット名>> ディレクトリの情報を返す

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 56 ロジカル・レプリケーション情報

```
postgres=# SELECT pg_create_physical_replication_slot('slot1') ;
 pg_create_physical_replication_slot
-------------------------------------
 (slot1,)
(1 row)

postgres=# SELECT * FROM pg_ls_replslotdir('slot1') ;
 name   | size |      modification
--------+------+-------------------------
 state |   200 | 2022-05-20 10:11:43+09
(1 row)

postgres=# SELECT * FROM pg_ls_logicalsnapdir() ;
       name       | size |      modification
------------------+------+-------------------------
 0-3005558.snap |   128 | 2022-05-20 10:06:19+09
(1 row1)

postgres=# SELECT * FROM pg_ls_logicalmapdir() ;
 name | size | modification
------+------+--------------
(0 rows)
```

内部ではレプリケーション・スロットの情報を読む READ_REPLICATION_SLOT コマンドを利用しています。

□ pg_log_backend_memory_contexts

SUPERUSER 権限のチェックが削除され、GRANT 文による権限付与により一般ユーザーでも使用できるようになりました。

□ 正規表現関数

正規表現を扱う関数として regexp_count、regexp_instr、regexp_substr が追加されました。Regexp_replace 関数には指定できるパラメーターが追加されました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

表 19 追加／変更された関数

関数名	説明
regexp_count	パターンにマッチする回数を返します。
Regexp_instr	パターンにマッチする場所を返します。
Regexp_like	パターンにマッチするかを返します。
Regexp_replace	パターンにマッチする部分を置換する。Start, flags オプションが追加されました。
Regexp_substr	パターンにマッチする部分を返します。

構文

- integer REGEXP_COUNT (string text, pattern text [, start integer [, flags text]])
- integer REGEXP_INSTR (string text, pattern text [, start integer [, N integer [, endpos integer [, flags text [, subexpr integer]]]]])
- boolean REGEXP_LIKE (string text, pattern text [, flags text])
- text REGEXP_REPLACE (string text, pattern text, replace text start integer, N integer [, flags text])
- text REGEXP_SUBSTR (string text, pattern text [, start integer [, N intger [, flags text [, subexpr integer]]])

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 57 正規表現関数の実行

```
postgres=> SELECT regexp_count('ABCABC', 'Abc', 1, 'i') ;
-[ RECORD 1 ]+--
regexp_count | 2

postgres=> SELECT regexp_like('PostgreSQL', 'Postgre(s|S)QL') ;
-[ RECORD 1 ]--
regexp_like | t

postgres=> SELECT regexp_instr('ABCDEFGHT', 'D.F') ;
-[ RECORD 1 ]+--
regexp_instr | 4

postgres=> SELECT regexp_replace('PostgreSQL', 'a|e|i|o|u', 'X', 1, 2) ;
-[ RECORD 1 ]--+----------
regexp_replace | PostgrXSQL

postgres=> SELECT regexp_substr('ABCDEFGHT', 'D.F', 2) ;
-[ RECORD 1 ]-+----
regexp_substr | DEF
```

□ unnest

UNNEST 関数に MULTIRANGE 型を指定できるようになりました。

構文

```
setof anyrange UNNNEST(anymultirange)
```

例 58 unnest 関数の実行

```
postgres=> SELECT unnest(int4multirange(int4range('5', '6'), int4range('1',
'2'))) ;
 unnest
--------
 [1,2)
 [5,6)
(2 rows)
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ pg_size_bytes / pg_size_pretty

ペタバイトまでサポートするようになりました。

例 59 pg_size_pretty 関数の実行

```
postgres=> SELECT pg_size_pretty(pg_size_bytes('10.5 PB')) ;
 pg_size_pretty
----------------
 11 PB
(1 row)
```

□ starts_with

starts_with 関数が BTREE インデックスを利用できるようになりました。同時に^@オペレーターでも同様の動作になります。

例 60 starts_with 関数の実行

```
postgres=> EXPLAIN SELECT * FROM data1 WHERE c2 ^@ '999' ;
                                QUERY PLAN
---------------------------------------------------------------------------------
 Index Scan using idx2_data1 on data1  (cost=0.42..1337.98 rows=100 width=12)
   Index Cond: ((c2 >= '999'::text) AND (c2 < '99:'::text))
   Filter: (c2 ^@ '999'::text)
(3 rows)

postgres=> EXPLAIN SELECT * FROM data1 WHERE starts_with(c2, '999') ;
                                QUERY PLAN
---------------------------------------------------------------------------------
 Index Scan using idx2_data1 on data1  (cost=0.42..1337.98 rows=100 width=12)
   Index Cond: ((c2 >= '999'::text) AND (c2 < '99:'::text))
   Filter: starts_with(c2, '999'::text)
(3 rows)
```

□ pg_settings_get_flags

GUC の属性情報をテキストの配列で帰します。存在しない GUC 名を指定した場合には NULL が返ります。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 61 pg_settings_get_flags 関数の実行

```
postgres=> SELECT * FROM pg_settings_get_flags('session_authorization') ;
          pg_settings_get_flags
-----------------------------------------
 {NO_RESET_ALL,NO_SHOW_ALL,NOT_IN_SAMPLE}
(1 row)

postgres=> SELECT * FROM pg_settings_get_flags('bad_guc_name') ;
 pg_settings_get_flags
-----------------------
null
(1 row)
```

□ 時刻指定文字列

OF、TZH および TZM を小文字でも使用できるようになりました。

例 62 of, tzh, tzm の指定

```
postgres=> SELECT to_char(current_date, 'of / tzh / tzm') ;
    to_char
---------------
 +09 / +09 / 00
(1 row)
```

□ poly_distance

poly_distance 関数が実装されました。従来のバージョンでもこの関数は存在しましたが、実行すると ERRCODE_FEATURE_NOT_SUPPORTED エラーが返っていました。

例 63 poly_distance 関数の実行

```
postgres=> SELECT
poly_distance('((10, 10),(20,20))'::polygon, '((30,30),(40,40))'::polygon) ;
   poly_distance
--------------------
 14.142135623730951
(1 row)
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.3. パラメーターの変更

PostgreSQL 15 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 20 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
allow_in_place_tablespaces	テーブル空間を pg_tblspc ディレクトリ内に作成する開発者向けパラメーター（superuser）	off
archive_library	WAL アーカイブ用ライブラリ名（sighup）	-
enable_group_by_reordering	複数列に対する GROUP BY 処理の最適化を行う（user）	on
log_startup_progress_interval	スタートアップ・プロセスが長時間実行された処理が発生した場合にログを出力するまでの時間（sighup）	10s
recovery_prefetch	リカバリ時に WAL のプリフェッチを行うか（sighup）	try
recursive_worktable_factor	再帰クエリ用のワークテーブル・サイズを決める乗数（user）	10
shared_memory_size	メイン共有メモリーのサイズ計算値（internal）	-
shared_memory_size_in_huge_pages	共有メモリーに使用する Huge Pages のページ数（internal）	-
stats_fetch_consistency	累積統計に複数回アクセスする場合の動作を決定します（user）	cache
wal_decode_buffer_size	WAL デコード用バッファー・サイズ（postmaster）	512KB

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ allow_in_place_tablespaces

このパラメーターを on に設定し、CREATE TABLESPACE 文の LOCATION 句に空文字列（''）を指定することでテーブル空間をデータベース・クラスタ内に作成することができます。

例 64 インプレース・テーブル空間

```
postgres=# SET allow_in_place_tablespaces = on ;
SET
postgres=# CREATE TABLESPACE ts1 LOCATION '' ;
CREATE TABLESPACE
postgres=# ¥! ls -l data/pg_tblspc/
total 0
drwx------. 3 postgres postgres 29 May 20 16:49 16433
```

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。

表 21 変更されたパラメーター

パラメーター	変更内容
compute_query_id	設定値に regress を指定できるようになりました。リグレッションテストを容易にするために使用されます。この変更は PostgreSQL 14.3 以降にもバックポートされます。
Log_destination	設定値として jsonlog が指定できるようになりました。Json フォーマットのログを出力できます。
Wal_compression	圧縮アルゴリズムとして pglz、lz4、zstd を指定できるようになりました。設定値 on は pglz とみなされます。

□ JSON ログ

パラメーターlog_destination に jsonlog を指定すると、ログファイルが JSON 形式で出力されます。拡張子は.json です。一部の情報のみ出力される拡張子.log のファイルも作成されます。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 65 JSON 形式ログ

```
postgres=> SHOW log_destination ;
 log_destination
-----------------
 jsonlog
(1 row)
postgres=> ¥! cat postgresql-2022-05-20_113806.json
{"timestamp":"2022-05-20 11:38:06.974 JST","pid":20848,"session_id":"6249088e.5170","line_num":1,"session_start":"2022-05-20 11:38:06 JST","txid":0,"error_severity":"LOG","message":"ending log output to stderr","hint":"Future log output will go to log destination ¥"jsonlog¥".","backend_type":"postmaster","query_id":0}
{"timestamp":"2022-05-20 11:38:06.974 JST","pid":20848,"session_id":"6249088e.5170","line_num":2,"session_start":"2022-05-20 11:38:06 JST","txid":0,"error_severity":"LOG","message":"starting PostgreSQL 15devel on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit","backend_type":"postmaster","query_id":0}
```

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 22 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 14	PostgreSQL 15	備考
hash_mem_multiplier	1.0	2.0	
log_autovacuum_min_duration	-1	10min	
log_checkpoints	off	on	
server_version	14.3	15beta1	
server_version_num	140003	150000	

3.3.4. 削除されたパラメーター

以下のパラメーターは削除されました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

表 23 削除されたパラメーター

パラメーター	理由
stats_temp_directory	統計情報の共有メモリー化により削除されました。

3.3.5. パラメーター変更時のエラー

いくつかの拡張モジュール（auto_explain, pg_prewarm, pg_stat_statements, plperl, plpgsql）では、モジュールのロード後に存在しないパラメーターを設定しようとするとエラーが発生するようになりました。

例 66 エラーの出力

```
postgres=> CREATE FUNCTION getwork() RETURNS TEXT AS $$
BEGIN
  RETURN current_setting('work_mem') ;
END ;
$$ LANGUAGE plpgsql ;
CREATE FUNCTION
postgres=> SET plpgsql.bad_parameter_name TO false ;
ERROR:  invalid configuration parameter name "plpgsql.bad_parameter_name"
DETAIL:  "plpgsql" is a reserved prefix.
```

© 2021-2022 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拡張点を説明します。

3.4.1. configure

Zstandard 圧縮メソッド対応のため、--with-zstd オプションが追加されました。Zstandard 1.4.0 以上がサポートされます。

例 67 Zstandard の利用

```
$ ./configure --help | grep -i zstd
  --with-zstd             build with ZSTD support
  ZSTD_CFLAGS C compiler flags for ZSTD, overriding pkg-config
  ZSTD_LIBS   linker flags for ZSTD, overriding pkg-config
```

3.4.2. psql

psql コマンドには以下の拡張が実装されました。

□ 環境変数 PSQL_WATCH_PAGER

¥watch コマンド実行時に表示されるページャーを指定する環境変数 PSQL_WATCH_PAGER が利用できるようになりました。

□ ¥dconfig コマンド

¥dconfig コマンドはインスタンスのパラメーター設定を確認できます。オプションを指定しない場合、デフォルト値から変更されたパラメーター一覧が表示されます。パラメーター名を指定すると特定のパラメーター設定値が確認できます。パラメーター名にはワイルドカードを使用できます。

構文

```
¥dconfig [parameter_name]
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 68 ¥dconfig コマンド

```
postgres=> ¥dconfig work_mem
List of configuration parameters
 Parameter | Value
-----------+-------
 work_mem  | 4MB
(1 row)

postgres=> ¥dconfig log_*connect*
List of configuration parameters
     Parameter      | Value
--------------------+-------
 log_connections    | off
 log_disconnections | off
(2 rows)
```

□ ¥getenv コマンド

環境変数を取得する¥getenv コマンドが追加されました。変数値を格納する psql 変数名と環境変数名を指定します。存在しない環境変数名を指定してもエラーにはなりません。

構文

```
¥getenv 変数名 環境変数名
```

例 69 ¥getenv コマンド

```
postgres=> ¥getenv home HOME
postgres=> ¥echo :home
/home/postgres
postgres=> ¥getenv val BADENV
postgres=> ¥echo :val
:val
```

□ SQL 文内のコメント

SQL 文内のスラッシュ２つ（--）のコメントがバックエンドまで送信されるようになりました。従来は送信前に削除されていました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ Large Object の出力

¥dl+コマンド、¥lo_list+コマンドにアクセス権限が出力されるようになりました。

例 70 Large Object の権限表示

```
postgres=> ¥lo_import 'test.dat'
lo_import 16409
postgres=> GRANT ALL ON LARGE OBJECT 16409 TO PUBLIC ;
GRANT
postgres=> ¥dl+
                    Large objects
  ID   | Owner | Access privileges | Description
-------+-------+-------------------+-------------
 16409 | demo  | demo=rw/demo     +|
       |       | =rw/demo          |
(1 row)

postgres=> ¥lo_list+
                    Large objects
  ID   | Owner | Access privileges | Description
-------+-------+-------------------+-------------
 16409 | demo  | demo=rw/demo     +|
       |       | =rw/demo          |
(1 row)
```

□ データベース名の指定

¥d コマンド等によりテーブル名を指定する際に「データベース名.スキーマ名.テーブル名」の形式が許可されます。ただし「データベース名」の部分は接続中のデータベースのみ有効です。他のデータベース名を指定した場合にはエラーが表示されます。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 71 データベース名の修飾

```
postgres=> CREATE TABLE public.data1(c1 INT, c2 VARCHAR(10)) ;
CREATE TABLE
postgres=> ¥d postgres.public.data1
                    Table "public.data1"
 Column |         Type          | Collation | Nullable | Default
--------+-----------------------+-----------+----------+---------
 c1     | integer               |           |          |
 c2     | character varying(10) |           |          |

postgres=> ¥d demodb.public.data1
cross-database references are not implemented: demodb.public.data1
postgres=>
```

□ 複数 SQL の出力

PostgreSQL 15 ではサーバーから複数の結果セットが返った場合、すべての結果を出力します。この動作は¥set SHOW_ALL_RESULTS off コマンドで変更できます。

3.4.3. pg_amcheck

--install-missing オプションが追加されました。このオプションを指定すると、amcheck エクステンションがインストールされてない場合、自動的にインストールされます。

3.4.4. pg_basebackup

pg_basebackup コマンドには以下のオプションが追加／拡張されました。

□ —target オプション

出力先を指定する—target オプションが追加されました。本オプションにより新しいデータ転送プロトコル（BASE_BACKUP TARGET）を使います。以下のフォーマットが指定できます。出力先として server を指定することで、PostgreSQL インスタンスが稼働するサーバー上にベース・バックアップを出力することができます。サーバー側のバックアップは SUPERUSER または pg_write_server_files ロールを持つユーザーが実行できます。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

表 24 —target オプションの設定値

ターゲット	意味
blackhole	何も出力されない。テスト用オプション
server:/*path*	サーバの指定されたディレクトリに出力。現状では出力フォーマットは tar となる（左記の例では/path に出力）。
Shell*[:DETAIL]*	シェルコマンドを実行。

--target オプションを指定するためには—wal-method オプションを fetch または none に設定する必要があります。Shell を指定する場合には Contrib モジュール basebackup_to_shell の設定が必要です。

□ --compress オプション

従来は圧縮レベルを指定していましたが、圧縮アルゴリズムと圧縮レベルを指定できるようになりました。圧縮レベルはコロン (:) の後に「level=数字」を指定します。Zstandard 圧縮では並列処理を行う「workers=数字」オプションも設定可能です。旧バージョンの指定も互換性のために利用できます。プレイン・フォーマット (-Fp) でサーバー側圧縮 (server-gzip) を指定した場合は、転送データのみ圧縮され、クライアント側で展開されて保存されます。この指定はネットワーク帯域が小さい場合に有効な設定です。

表 25 —compress オプションの圧縮設定

設定	意味
none	圧縮しない
gzip	ターゲット指定されていない場合はクライアントで gzip 圧縮
lz4	ターゲット指定されていない場合はクライアントで LZ4 圧縮
zstd	ターゲット指定されていない場合はクライアントで Zstandard 圧縮
client-gzip	クライアント側で gzip 圧縮
server-gzip	サーバー側で gzip 圧縮
client-lz4	クライアント側で LZ4 圧縮
server-lz4	サーバー側で LZ4 圧縮
client-zstd	クライアント側で Zstandard 圧縮
server-zstd	サーバー側で Zstandard 圧縮

例 72 圧縮メソッドの指定

```
$ pg_basebackup --compress=none --format=tar -D back.1
$ pg_basebackup --compress=zstd:level=9,workers=2 --format=tar -D back.2
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

3.4.5. pg_dump

pg_dump コマンドには以下の拡張が実装されました。

□ —no-table-access-method

出力される DDL からテーブル・アクセス・メソッドを削除する—no-table-access-method オプションを指定できるようになりました。このオプションはダンプファイルを非テキスト・フォーマットで出力した場合には無視されます。また pg_dumpall コマンド、pg_restore コマンドにも同様のオプションが指定できます。

□ TOAST データ量の考慮

pg_dump コマンドは並列モード実行時にテーブルのサイズを考慮します。PostgreSQL 15 では TOAST データ量も考慮するようになりました。

3.4.6. pg_recvlogical

レプリケーション・プロトコルのコマンド CREATE_REPILICATION_SLOT の TWO_PHASE オプション追加に対応するため、pg_recvlogical コマンドに—two-phase オプションが追加されました。

例 73 —two-phase オプション

```
$ pg_recvlogical --create-slot --slot=slot1 --two-phase --dbname=postgres
$ psql
postgres=# SELECT two_phase FROM pg_replication_slots WHERE slot_name='slot1';
 two_phase
-----------
 t
```

3.4.7. pg_receivewal

ログの圧縮方法を指定する—compression オプションが変更されました。圧縮メソッド、レベル等が指定できます。

構文

```
pg_receivewal --compress=METHOD[:DETAIL]
METHOD = {'gzip', 'none', 'lz4'}
DETAIL = level=[1-9]
```

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

圧縮方法として有効な値は gzip, lz4 または none です。オプションで圧縮レベル (level) を指定できます。圧縮レベルの範囲は 1～9 です。

例 74 —two-phase オプション

```
$ pg_receivewal -D wal --compress=lz4:level=9
^Cpg_receivewal: not renaming "000000010000000000000028.lz4.partial", segment is not complete
$ ls wal
000000010000000000000022.lz4  000000010000000000000026.lz4
000000010000000000000023.lz4  000000010000000000000027.lz4
000000010000000000000024.lz4  000000010000000000000028.lz4.partial
000000010000000000000025.lz4
```

LZ4 圧縮を実行するには—with-lz4 オプション付きでコンパイルされている必要があります。

3.4.8. pg_resetwal

もっとも古いトランザクション ID を指定する—oldest-transaction-id オプション（短縮形-u）が追加されました。

3.4.9. pg_rewind

設定ファイル(postgresql.conf)のパスを指定する—config-file オプションが追加されました。このオプションは設定ファイルがデータディレクトリ外にある時に役立ちます。

3.4.10. pg_upgrade

pg_upgrade コマンドには以下の拡張が実装されました。

□ —no-sync オプション

--no-sync オプション（短縮形-N）が追加されました。このオプションを指定すると、ファイルの書き込み完了を待ちません。このオプションはテスト用であり、商用環境での使用は想定されていません。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

□ 一時ディレクトリ

アップグレード中に作成される一時ファイルは新規データベース・クラスターの pg_upgrade_output.d ディレクトリに作成されます。

3.4.11. pg_waldump

pg_waldump コマンドには以下の拡張が追加されました。

□ オプション

複数のオプションが追加されました。また--rmgr オプションを複数指定できるようになりました。

表 26 追加されたオプション

オプション	短縮形	説明
--block=N	-B	--relation で指定されたリレーションのブロックを表示
--fork=N	-F	マッチした番号のフォークのみ表示
--relation=N/N/N	-R	該当するリレーションの情報のみを表示
--fullpage	-w	Full page write の情報のみを表示

□ 出力情報の追加

コマンド出力にトランザクション・コミットの remote_apply 情報、PREPARE TRANSACTION のレプリケーション・オリジン情報が追加されます。

例 75 コマンド出力ログ

```
$ pg_waldump data/pg_wal/000000010000000000000008 | grep origin
rmgr: Transaction len (rec/tot):     117/    117, tx:          775, lsn: 0/080025F0, prev 0/080025C0, desc: COMMIT 2022-05-20 10:01:46.390500 JST; inval msgs: catcache 57 catcache 56 catcache 66; origin: node 2, lsn 0/0, at 2022-05-20 10:01:46.382305 JST
rmgr: Transaction len (rec/tot):      85/     85, tx:          776, lsn: 0/08002770, prev 0/08002740, desc: COMMIT 2022-05-20 10:01:46.391231 JST; inval msgs: catcache 66; origin: node 2, lsn 0/0, at 2022-05-20 10:01:46.382305 JST
<<<以下省略>>>
```

□　シグナル

SIGINT シグナルを受信して終了すると、サマリー情報が出力されるようになりました。

© 2021-2022 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. amcheck

シーケンスのチェックがサポートされました。

例 76 シーケンスのチェック

```
postgres=# CREATE SEQUENCE seq1 ;
CREATE SEQUENCE
postgres=# SELECT verify_heapam(relation=>'seq1') ;
 verify_heapam
---------------
(0 rows)
```

3.5.2. basebackup_to_shell

ベースバックアップのターゲットに shell を指定した場合に利用されるモジュールです。このモジュールを利用する場合にはパラメーターshared_preload_libraries または local_preload_libraries に basebackup_to_shell を指定します。以下のパラメーターを指定できます。

表 27 使用できるパラメーター

パラメーター名	説明
basebackup_to_shell.command	実行されるコマンド・パス
basebackup_to_shell.required_role	実行に必要なロール

3.5.3. file_fdw

CREATE FOREIGN TABLE 文の header オプションに match を指定できるようになりました。このオプションを指定すると外部テーブルの検索時にテキスト・ファイルのヘッダ行と列名が一致しているかチェックを行います。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 77 ヘッダの一致チェック

```
postgres=# ¥! cat /tmp/data1.csv
c1,c3
1,data1
2,data2
postgres=# CREATE EXTENSION file_fdw ;
CREATE EXTENSION
postgres=# CREATE SERVER filesvr1 FOREIGN DATA WRAPPER file_fdw ;
CREATE SERVER
postgres=# CREATE FOREIGN TABLE head1 (c1 int, c2 text) SERVER filesvr1
        OPTIONS (format 'csv', filename '/tmp/data1.csv', delimiter ',',
                header 'match') ;
CREATE FOREIGN TABLE
postgres=# SELECT * FROM head1 ;
ERROR:  column name mismatch in header line field 2: got "c3", expected "c2"
CONTEXT:  COPY head1, line 1: "c1,c3"
```

3.5.4. pg_stat_statements

pg_stat_statements ビューにはモニタリングできる項目が追加されています。

□ 一時ファイルに対する I/O 情報

一時ファイルに対する読み書き時間がモニタリングできるようになりました。pg_stat_statements ビューに以下の列が追加されています。

表 28 追加された列情報

列名	データ型	説明
temp_blk_read_time	double precision	一時ブロック読み込み時間
temp_blk_write_time	double precision	一時ブロック書き込み時間

□ JIT 関連情報の追加

JIT 関連の情報がモニタリングできるようになりました。pg_stat_statements ビューに以下の列が追加されています。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

表 29 追加された列情報

列名	データ型	説明
jit_functions	bigint	JIT コンパイルされた関数の実行回数
jit_generation_time	double precision	JIT コードの生成時間（ミリ秒）
jit_inlining_count	bigint	インライン化された関数の実行回数
jit_inlining_time	double precision	インライン化された関数の実行時間（ミリ秒）
jit_optimization_count	bigint	最適化された文の実行回数
jit_optimization_time	double precision	最適化された文の実行時間（ミリ秒）
jit_emission_count	bigint	コードが発行された回数
jit_emission_time	double precision	コードが発行に使用された時間（ミリ秒）

3.5.5. pg_walinspect

pg_walinspect モジュールは pg_waldump コマンドに近い処理を SQL 文で実行するためのモジュールです。以下の関数が提供されています。これらの関数は SUPERUSER 属性または pg_read_server_files ロールを持つユーザーのみ実行できます。

表 30 提供されている関数

関数名	説明
pg_get_wal_record_info	指定された LSN 間の WAL 情報を返す
pg_get_wal_records_info	指定された LSN 間の WAL 情報を返す
pg_get_wal_records_info_till_end_of_wal	指定された LSN から最後まで WAL 情報を返す
pg_get_wal_stats	指定された LSN 間の WAL 統計を返す
pg_get_wal_stats_till_end_of_wal	指定された LSN から最後まで WAL 統計を返す

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 78 pg_get_wal_stats 関数の実行

```
postgres=# CREATE EXTENSION pg_walinspect ;
CREATE EXTENSION
postgres=# SELECT pg_get_wal_stats('0/B4BB822'::pg_lsn ,
                  '0/B4BB828'::pg_lsn) ;
          pg_get_wal_stats
-------------------------------------
 (XLOG,0,0,0,0,0,0,0,0)
 (Transaction,0,0,0,0,0,0,0,0)
<<以下省略>>
```

3.5.6. postgres_fdw

postgres_fdw には以下の拡張が実装されました。

□ アプリケーション名の指定

リモート・コネクションに対して application_name を指定できるようになりました。postgres_fdw.application_name パラメーターを変更します。

例 79 GUC の設定

```
postgres=# SET postgres_fdw.application_name TO 'fdw_name1' ;
SET
postgres=# CREATE SERVER svr5432 FOREIGN DATA WRAPPER postgres_fdw OPTIONS
(host 'remhost1', port '5432', dbname 'postgres') ;
CREATE SERVER
```

アプリケーション名には以下のエスケープ・シーケンスを指定できます。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

表 31 使用できるエスケープ・シーケンス

エスケープ・シーケンス	説明	備考
%a	ローカル・アプリケーション名	
%c	セッション ID	
%C	クラスター名（cluster_name）	
%d	ローカル・データベース名	
%u	ローカル・ユーザー名	
%p	バックエンド・プロセス ID	
%%	文字%	

□ parallel_commit オプション

リモート・トランザクションのコミットの実行方法を制御する parallel_commit オプションが追加されました。このオプションを on に設定すると、リモート・トランザクションのコミットを並列に行うことができます。デフォルト値は off で、コミットをシリアルに実行されます。

例 80 parallel_commit オプションの設定

```
postgres=# CREATE SERVER svr5433 FOREIGN DATA WRAPPER postgres_fdw OPTIONS
( host 'remhost1', port '5433', dbname 'postgres', parallel_commit 'true') ;
CREATE SERVER
postgres=# SELECT srvoptions FROM pg_foreign_server WHERE srvname='svr5433' ;
                            srvoptions
------------------------------------------------------------------
 {host=localhost,port=5433,dbname=postgres,parallel_commit=true}
(1 row)
```

□ CASE 式

CASE 式がリモート・インスタンスにプッシュされるようになりました。

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

例 81 CASE 式のプッシュ

```
postgres=> EXPLAIN (ANALYZE, VERBOSE) SELECT COUNT(*) FROM remote1 WHERE
                CASE WHEN c1 > 100 THEN c1 END < 100 ;
                                QUERY PLAN
-------------------------------------------------------------------------------
 Foreign Scan  (cost=102.84..164.07 rows=1 width=8) (actual time=1.364..1.365
rows=1 loops=1)
   Output: (count(*))
   Relations: Aggregate on (public.remote1)
   Remote SQL: SELECT count(*) FROM public.remote1 WHERE (((CASE WHEN (c1 >
100::numeric) THEN c1 ELSE NULL::numeric END) < 100::numeric))
 Planning Time: 0.077 ms
 Execution Time: 1.620 ms
(6 rows)
```

3.5.7. sepgsql

ログ内に permissive/enforcing 状態が追加されました。permissive=0 または permissive=1 が行末に出力されます。

© 2021-2022 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/15/release-15.html
- Commitfests
  https://commitfest.postgresql.org/
- PostgreSQL 15 Manual
  https://www.postgresql.org/docs/15/index.html
- Git
  git://git.postgresql.org/git/postgresql.git
- GitHub
  https://github.com/postgres/postgres
- PostgreSQL 15 Beta 1 のアナウンス
  https://www.postgresql.org/about/news/postgresql-15-beta-1-released-2453/
- Postgres Professional
  https://habr.com/ru/company/postgrespro/blog/541252/
- PostgreSQL 15 Open Items
  https://wiki.postgresql.org/wiki/PostgreSQL_15_Open_Items
- Qiita（ぬこ@横浜さん）
  http://qiita.com/nuko_yokohama
- pgsql-hackers Mailing list
  https://www.postgresql.org/list/pgsql-hackers/
- PostgreSQL Developer Information
  https://wiki.postgresql.org/wiki/Development_information
- pgPedia
  https://pgpedia.info/postgresql-versions/postgresql-15.html
- SQL Notes
  https://sql-info.de/postgresql/postgresql-15/articles-about-new-features-in-postgresql-15.html
- Slack - postgresql-jp
  https://postgresql-jp.slack.com/

© 2021-2022 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	2022/05/10	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・パッカード合同会社）
1.0	2022/05/24	篠田典良	PostgreSQL 15 Beta 1 公開版に合わせて修正完了

以上

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.

© 2021-2022 Hewlett Packard Enterprise Japan Co, Ltd.