PGECons
PostgreSQL Enterprise Consortium

PostgreSQLロールによる権限管理入門

2024年度成果報告

PostgreSQLエンタープライズ・コンソーシアム

WG2

アジェンダ

- WG2の活動紹介
  - 移行WG（WG2）活動内容
- 2024年度活動紹介
  - PostgreSQLのロール・権限の調査
- おわりに

WG2の活動紹介

ワーキンググループ活動について

- 現在3つのワーキンググループにて活動中

□ **新技術検証WG（WG1）**
- 新バージョンの性能や新技術の検証を通じて有用性を明確化
- スケールアップ検証、新機能における性能特性調査等

□ **移行WG（WG2）**
- 異種DBMSからの移行をテーマに活動
- 商用DBMSからの移行プロセスに伴う技術調査や検証を実施

□ **課題検討WG（WG3）**
- データベース管理者やアプリケーション開発者が抱える
  現場の課題や困り事に対するテーマを設定
- 可用性・運用性・保守性・セキュリティ・接続性が主な課題領域

移行WG(WG2)活動内容

活動テーマ:異種DBMSからPostgreSQLへの移行

課題認識

- 異種DBMSシステムをPostgreSQLへ移行するプロセスが確立していないことが、普及を妨げる大きな障壁と認識
  - 移行作業をどのように進めればよいかがわからない。
  - 初期段階で移行に必要なトータルコストを算出できない。
  - 過去の経験則や点在するノウハウに依存しているのが現状

活動目標

- 異種DBMSからPostgreSQLへの移行を検討する際のガイドラインを提示する。
  (難易度判断、留意すべき事項、移行手順)

成果物

- 「異種DBMSからPostgreSQLへの移行ガイド」を作成

成果物の公開

- **成果物は無償でダウンロード可能**
  - https://pgecons-sec-tech.github.io/tech-report/

移行WG (WG2)

「異種DBMSからPostgreSQLへの移行」をテーマとして調査・検証を行い、収集した技術ノウハウを成果として取り纏めた資料を公開しています。

成果物一覧

文書名	概要	リンク(HTML・PDF版)	リンク(圧縮版)	最終更新日
PostgreSQL自習書	Oracleデータベース経験者がPostgreSQLの概要を理解することを目的とした技術文書です。	• 本文(PDF)		2021/05/25 NEW!
移行ガイドブック	これからPostgreSQLへの移行を検討される方の一助となる、DBMS移行の概要をつかめるガイドブックです。	• 本文(HTML) • 本文(PDF)		2020/09/02
異種DBMSからPostgreSQLへの移行ガイド	各成果物の活用場面をイメージして頂くために、一般的なシステム移行手順を提示した上で、各タスクとWG2成果物の関係を表現しています。	• 本文(HTML) • 本文(PDF)		2017/06/22
DB移行フレームワーク編	異種DBMSからの移行とは具体的に何を行うのかを紹介します。 DBMSの移行作業において一般的に発生すると考えられる作業工程を定義し、各工程における検討結果をベースとして移行可否判断の手がかりとなる情報を提供します。	• 本文(PDF)	• 2013年度WG2成果物(zip) • 2013年度WG2成果物(tar.bz2)	2014/04/16
システム構成調査編	DBMSの代表的なシステム構成とその特徴を挙げ、PostgreSQL移行時に採用可能な構成を紹介します。	• 本文(PDF)	• 2013年度WG2成果物(zip) • 2013年度WG2成果物(tar.bz2)	2014/04/16

2024年度活動報告

活動メンバー

- 2024年度は下記3社にて活動
  - NECソリューションイノベータ株式会社（主査）
  - 富士通株式会社
  - 三菱電機株式会社

2024年度 WG2の活動

新規テーマ

- PostgreSQLのロール・権限の調査
  - これまであまり触れられていなかった、権限・ロールについて整理
  - PostgreSQLの情報について整理しつつ、Oracleとの比較を検討

ロールの概要

ロールとは

これらの両方の概念が含まれている。

- PostgreSQLのデータベースユーザー
- PostgreSQLのデータベースユーザーのグループ

- 昔のPostgreSQL
  - PostgreSQLバージョン8.1より前まで「ユーザー」と「グループ」は別の物だった

データベースロール概要

PostgreSQLはロールを使って以下の管理をする

- PostgreSQLの接続承認の管理
  - クライアントからデータベースへ接続・接続の認証をするためロール名を使う
  - このロールにはLOGIN権限が必要
- データベースオブジェクトに対する操作の管理
- データベースオブジェクトの所有者
- データベースオブジェクトのアクセス権限の管理

ロールの基礎

- データベースロールとOSユーザーは別のもの
  - データベースインスタンス作成時は、デフォルトではinitdbを実行した OSユーザー名と同じ名前のロールが初期ユーザーとして作成される
  - このロールはスーパーユーザー権限とログイン権限を持つ
- ロール名はデータベースクラスタ全体で共通
  - pg_rolesシステムカタログで管理する
    - このシステムカタログはデータベースクラスタで共通で利用する

ロールの属性

- CREATE ROLEコマンドで新しいロールを作成する
  - またはCREATE USERコマンドやシェルコマンドラインのcreateruserを利用する（CREATE USERやcreateuserではデフォルトでLOGIN権限付与）
- ロールの作成時にそのロールの属性を定義できる
  - SUPERUSER/NOSUPERUSER：スーパーユーザーかどうか
  - CREATEDB/NOCREATEDB：データベース作成権限を持つかどうか
  - CREATEROLE/NOCREATEROLE：ロール作成権限を持つかどうか
  - LOGIN/NOLOGIN：データベースサーバーへログイン可能かどうか
  - REPLICATION/NOREPLICATION：レプリケーションロールかどうか
  - さらに詳しくはCREATE ROLEやALTER ROLEのドキュメントを参照
- ALTER ROLEでロールの属性を変更する
- DROP ROLEでロールを削除する

ロールの権限

- データベースオブジェクトの所有者としてのロール
  - CREATE文を実行する実行ロールが所有者となる

例：ロールidaがテーブルaccountを作成
=# CREATE TABLE account (id integer);

  - ALTERで所有者変更が可能
    - スーパーユーザーで実行可能
    - 現在の所有者（または所有者の権限を継承）で新しい所有者へSET ROLEできる場合

例：accountテーブルの所有者をidaからjonに変更する
=# ALTER TABLE account OWNER TO jon;

ロール権限

- 所有者とスーパーユーザー以外はオブジェクトのアクセス権限が必要
  - SELECT：列の参照を許可
  - INSERT：新しい行のINSERTを許可
  - UPDATE：列のUPDATEを許可。SELECT権限も必要
  - DELETE：列のDELETEを許可。SELECT権限も必要。
  - TRUNCATE：テーブルにTRUNCATEを許可
  - さらに詳しくは「権限」を参照
- GRANT文で定義
- REVOKE文で権限を取り消し

グループとしてのロール

- データベースユーザーもグループも「ロール」と呼ぶ
- 権限管理を簡単にするためにユーザーをグループ用のロールにまとめる
  - GRANTコマンドでユーザーを追加する
  - REVOKEコマンドでユーザーを削除する

例：adminロール（グループ）へbrianロール（ユーザー）を追加する

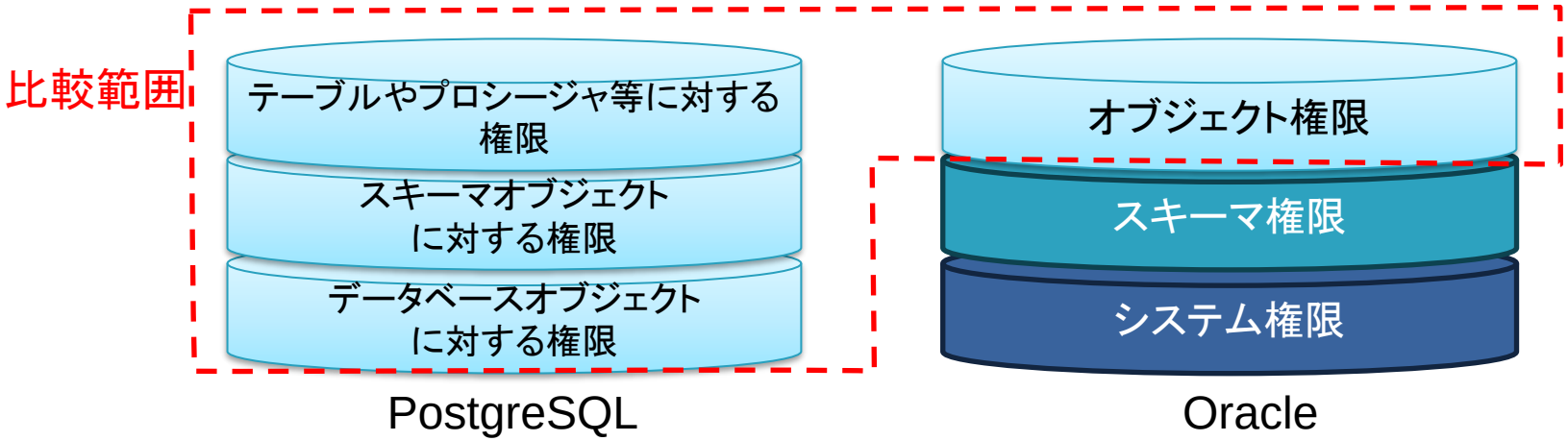
```
=# CREATE ROLE admin;
=# GRANT admin TO brian;
=# REVOKE admin FROM brian ;
```

- グループ用ロールの削除
  - DROP ROLEを利用する
  - ユーザーはそのグループロールによる資格がはく奪される
  - ユーザーが削除されることはない

オブジェクト権限

主なオブジェクト権限の比較

- PostgreSQL 17.4とOracle DB 23aiのマニュアルをもとに、オブジェクトに対する主な権限の比較を行った。
  - テーブルに対するSELECTなどの主な権限は、PostgreSQL、Oracleともに使用可能。ただし、仕様に差異がある部分もあるので注意が必要。
  - また、PostgreSQLではオブジェクトとしてデータベースやスキーマを対象にすることで、データベース全体やスキーマ全体に対する権限の設定が可能。それに対して、Oracleではシステム権限やスキーマ権限によって同じような設定が可能だが、GRANTの構文の違いなどに注意が必要。



テーブルに対する権限

- Oracleの方が権限の種類が多いが、基本的な権限（SELECT、DELETEなど）は両方に存在。
- SELECT ・・・ FOR UPDATEなど、それぞれ権限に差異がある。

権限	PostgreSQL	Oracle	備考
SELECT	○	○	PostgreSQLでは、SELECT … FOR UPDATEにUPDATE権限も必要。
READ	－	○	SELECTと同様。SELECT … FOR UPDATEを許可しない。
INSERT	○	○	
UPDATE	○	○	
DELETE	○	○	
TRUNCATE	○	－	Oracleではシステム権限やスキーマ権限のDROP ANY TABLE権限が必要。
REFERENCES	○	○	
MAINTAIN	○	－	
ALTER	－	○	
DEBUG	－	○	
INDEX	－	○	
FLASHBACK	－	○	
SIGN	－	○	ブロックチェーン表内の行への署名に必要。

テーブルの列に対する権限

- PostgreSQLではSELECT権限があるのに対して、Oracleでは列単位のSELECT権限がない。Oracleで特定の列のSELECTを制限するためにはビューを介する必要があると思われる。

権限	PostgreSQL	Oracle	備考
SELECT	○	－	
INSERT	○	○	
UPDATE	○	○	
REFERENCES	○	○	

ファンクション、プロシージャに対する権限

- Oracleはデバッグ実行に関する権限がある。

権限	PostgreSQL	Oracle	備考
EXECUTE	○	○	
DEBUG	―	○	

シーケンスに対する権限

- PostgreSQLではSELECT権限でcurrvalを管理し、UPDATE権限などでnextvalを管理するのに対して、OracleではSELECT権限でnextvalの管理も含む。

権限	PostgreSQL	Oracle	備考
SELECT	○	○	Oracleではnextvalの権限を含む。
UPDATE	○	ー	nextval、setvalの使用。
USAGE	○	ー	currval、nextvalの使用。
ALTER	ー	○	シーケンスの変更。
KEEP SEQUENCE	ー	○	nextval使用時に元の値を保持可能になる。エラー発生時の動作に影響。

データベースに対する権限

- PostgreSQLでは、GRANT文の対象オブジェクトにデータベースを指定可能。データベース全体のスキーマやテーブル等の作成（CREATE）、一時表の作成（TEMPORARY）、接続（CONNECT）の権限を管理する。
- 一方、Oracleではデータベースオブジェクトに対する設定ではなく、GRANT文を使用したシステム権限として管理する。

権限	PostgreSQL	Oracle	備考
CREATE	○	―	Oracleではシステム権限によりスキーマ作成やテーブル作成を管理する。
TEMPORARY	○	―	
CONNECT	○	―	OracleではシステムCREATE SESSION権限を使用する。

スキーマに対する権限

- PostgreSQLでは、GRANT文の対象オブジェクトにスキーマを指定可能。スキーマ内のオブジェクト作成(CREATE)、オブジェクトへのアクセス(USAGE)の権限を管理する。
- 一方、Oracleではスキーマオブジェクトに対する設定ではなく、GRANT文を使用したスキーマ権限として管理する。

権限	PostgreSQL	Oracle	備考
CREATE	○	―	Oracleではスキーマ権限によりテーブルなどオブジェクト作成を管理する。
USAGE	○	―	

定義済みロール

定義済みロール

- 定義済みロールとは？
  - データベースの運用に必要な特定の機能の利用、および情報の参照のための権限をあらかじめ割り当てられたロール
    - システムビューの参照
    - シグナルの送信
    - ファイル入出力
    - メンテナンスコマンドの実行
    - etc...
  - 通常はスーパーユーザやデータベース所有者などの権限が必要になる操作が、定義済みロールを付与することで他のユーザでも実行可能となる
    ⇒ 非スーパーユーザーの管理用ユーザーの作成
  - PostgreSQL 9.6で「デフォルトロール」として登場
    ⇒ PostgreSQL 14で現在の呼称に変更

定義済みロールの変遷

■ 定義済みロール一覧

追加バージョン	ロール名	概要
9.6	pg_signal_backend	他のバックエンドに問い合わせのキャンセルやセッションの終了のシグナルを送信する
10	pg_read_all_settings	通常スーパーユーザのみが読み取れる、全ての設定変数を読み取る
	pg_read_all_stats	通常スーパーユーザのみが読み取れる、すべてのpg_stat_*ビューを読み取り、各種の統計関連のエクステンションを使用する
	pg_stat_scan_tables	潜在的に長時間、テーブルのACCESS SHAREロックを取得する可能性がある監視機能を実行する
	pg_monitor	各種の監視ビューや機能を読み取り/実行する
11	pg_read_server_files	COPYやその他のファイルアクセス関数で、サーバ上でアクセスできる任意の場所からファイルを読み取ることを許可する
	pg_write_server_files	COPYやその他のファイルアクセス関数で、サーバ上でアクセスできる任意の場所にファイルを書き込むことを許可する
	pg_execute_server_program	COPYやサーバ側のプログラムを実行できるその他の関数で、データベースを実行しているユーザとしてデータベースサーバ上でのプログラムの実行を許可する
14	pg_read_all_data	それらのオブジェクトに対するSELECT権限を持っていて、明示的に持っていなかったとしてもすべてのスキーマに対してUSAGE権限を持っているかのように、すべてのデータ(テーブル、ビュー、シーケンス)を読み取る
	pg_write_all_data	それらのオブジェクトに対するINSERT、UPDATEおよびDELETE権限を持っていて、明示的に持っていなかったとしてもすべてのスキーマに対してUSAGE権限を持っているかのように、すべてのデータ(テーブル、ビュー、シーケンス)に書き込む
	pg_database_owner	データベース所有者の権限。メンバ資格は暗黙に現在のデータベースの所有者から構成される
15	pg_checkpoint	CHECKPOINTコマンドの実行を許可する
16	pg_use_reserved_connections	reserved_connectionsによって予約済みの接続スロットの使用を許可する
	pg_create_subscription	CREATE SUBSCRIPTIONを実行するために、ユーザに対してデータベースへのCREATE権限を許可する
17	pg_maintain	すべてのリレーションに対してVACUUM、ANALYZE、CLUSTER、REFRESH MATERIALIZED VIEW、REINDEXおよびLOCK TABLEを実行することを許可する

システムビューの参照

- pg_read_all_settings ロール
  - PostgreSQLの設定情報 (pg_settings) を参照可能とするロール
  - 通常は一部のパラメータはスーパーユーザ以外は参照不可
- pg_read_all_stats ロール
  - pg_stat_XXX ビューおよび関連関数
  - pg_backend_memory_contexts、 pg_shmem_allocations ビューおよび関連関数
  - pg_database_size、pg_tablespace_size 関数
  - 拡張機能 (pg_stat_statements)
- pg_stat_scan_tables ロール
  - 拡張機能 (pgstattuple、pg_freespacemap、pg_visibility、pgrowlocks)
- pg_monitor ロール
  - 上記3ロールを包含する権限を持つロール

システムビューの参照

[実行ユーザ]
=> testuserユーザ
=# postgresユーザ（SUPERUSER）

■ 実行例

□ セッション情報の参照（ユーザ自身の情報のみ出力）

```
testdb=> select pid, datname, usename, state, xact_start from pg_stat_activity where backend_type = 'client backend';
  pid   | datname | usename  | state  |          xact_start
--------+---------+----------+--------+-------------------------------
 319373 | testdb  | testuser | active | 2025-05-01 16:41:54.518154+09
(1 row)
```

□ pg_read_all_stats権限を付与

```
testdb=# grant pg_read_all_stats to testuser;
GRANT ROLE
```

□ セッション情報の参照（他のユーザの情報も含めて出力）

```
testdb=> select pid, datname, usename, state, xact_start from pg_stat_activity where backend_type = 'client backend';
  pid   | datname |  usename  |        state        |          xact_start
--------+---------+-----------+---------------------+-------------------------------
 319373 | testdb  | testuser  | active              | 2025-05-01 16:42:39.465203+09
 319660 | testdb  | testuser2 | idle in transaction | 2025-05-01 16:39:05.439286+09
(2 rows)
```

シグナルの送信

- pg_signal_backend ロール
  - 一部のサーバシグナル送信関数の処理権限を付与

pg_cancel_backend	指定したプロセスIDを持つバックエンドプロセスの現在のセッションの問い合わせを取り消す
pg_terminate_backend	バックエンドが指定したプロセスIDを持つセッションを終了させる

  - 通常は以下のケースでのみ使用可能
    - スーパーユーザ権限を持つユーザで実行
    - 自身が所属するロールのセッションに対する実行
  - 本ロールの権限を付与しても、スーパーユーザのセッションに対しては処理不可

シグナルの送信

[実行ユーザ]
=> testuserユーザ
=# postgresユーザ（SUPERUSER）

■ 実行例

□ 他のユーザのセッションの終了（権限不足でエラー）

```
testdb=> select pg_terminate_backend(319660);
ERROR:  permission denied to terminate process
DETAIL:  Only roles with privileges of the role whose process is being terminated or with privileges of the
"pg_signal_backend" role may terminate this process.
```

□ pg_signal_backend権限を付与

```
testdb=# grant pg_signal_backend to testuser;
GRANT ROLE
```

□ 他のユーザのセッションの終了（成功）

```
testdb=> select pg_terminate_backend(319660);
 pg_terminate_backend
----------------------
 t
(1 row)
```

ファイル入出力

- pg_read_server_files、pg_write_server_files ロール
  - COPY TO/FROMによるサーバ上へのファイル入出力を可能とする
    - 通常はスーパーユーザ以外では実行不可のため、下記方法で回避
      - 標準入出力(STDIN/STDOUT)を使用してファイルへのリダイレクト
      - psqlの¥copyメタコマンドでクライアント側のファイル操作
  - その他の対象となる操作
    - read
      - 関数:pg_read_file、pg_read_binary_file、pg_stat_file、pg_ls_dir
      - 拡張:file_fdw、pg_walinspect
    - write
      - コマンド:pg_basebackup

ファイル入出力

[実行ユーザ]
=> testuserユーザ
=# postgresユーザ（SUPERUSER）

■ 実行例

□ COPY TOによるサーバ上へのファイル出力（権限不足でエラー）

```
testdb=> copy pgbench_accounts to '/var/lib/pgsql/pgbench_accounts.csv' csv;
ERROR:  permission denied to COPY to a file
DETAIL:  Only roles with privileges of the "pg_write_server_files" role may COPY to a file.
HINT:  Anyone can COPY to stdout or from stdin. psql's ¥copy command also works for anyone.
```

□ pg_write_server_files権限を付与

```
testdb=# grant pg_write_server_files to testuser;
GRANT ROLE
```

□ COPY TOによるサーバ上へのファイル出力（成功）

```
testdb=> copy pgbench_accounts to '/var/lib/pgsql/pgbench_accounts.csv' csv;
COPY 100000
```

テーブルの参照権は別途必要

メンテナンスコマンドの実行

- pg_maintain ロール（PostgreSQL 17で追加）
  - テーブル系のオブジェクトに対する下記のメンテナンス系操作を許可する
    - VACUUM
    - ANALYZE
    - CLUSTER
    - REFRESH MATERIALIZED VIEW
    - REINDEX
    - LOCK TABLE
  - システムテーブルを含むすべてのテーブルに対する権限が付与される
    - 権限を持たないテーブルに対しての処理はエラー、もしくはスキップされる

メンテナンスコマンドの実行

[実行ユーザ]
=> testuserユーザ
=# postgresユーザ（SUPERUSER）

■ 実行例

□ MAINTAIN権限を持たずVACUUM ANALYZE処理をスキップ、REINDEXはエラー

```
testdb=> vacuum analyze pgbench_accounts;
WARNING:  permission denied to vacuum "pgbench_accounts", skipping it
VACUUM
testdb=> reindex table pgbench_accounts;
ERROR:  permission denied for table pgbench_accounts
```

□ pg_maintain権限を付与

```
testdb=# grant pg_maintain to testuser;
GRANT ROLE
```

□ VACUUM ANALYZEおよびREINDEX成功

```
testdb=> vacuum analyze pgbench_accounts;
VACUUM
testdb=> reindex table pgbench_accounts;
REINDEX
testdb=> select count(*) from pgbench_accounts;
ERROR:  permission denied for table pgbench_accounts
```

MAINTAIN権限のみ付与されるため、
テーブルの参照などは不可のまま

おわりに

2024年度の活動を振り返って

- 成果
  - これまで触れられていなかったロールや権限について調査実施
- 反省
  - 成果物の作成まで至らずプレビュー状態となってしまった
  - 過去のドキュメントの最新化も検討していたが未着手

2025年度に取り組みたいこと

- 今年度取り組めなかったテーマに取り組む
- 継続的に過去の成果物の更新
  - 少なくとも、調査対象がEOLが過ぎているもの・来年迎えるものを更新したい

移行で検討してほしいテーマがあれば
アンケートで教えてください。

ライセンス

文書の内容、表記に関する誤り、ご要望、感想等につきましては、PGEConsのサイトを通じてお寄せいただきますようお願いいたします。

- Linux **は、**Linus Torvalds **氏の日本およびその他の国における登録商標または商標です。**
- Oracle**は、**Oracle Corporation **及びその子会社、関連会社の米国及びその他の国における登録商標です。 文中の社名、商品名等は各社の商標または登録商標である場合があります。**
- PostgreSQL**は、**PostgreSQL Community Association of Canada**のカナダにおける登録商標およびその他の国における商標です。**
- Windows **は米国** Microsoft Corporation **の米国およびその他の国における登録商標です。**
- **その他、本資料に記載されている社名及び商品名はそれぞれ各社が 商標または登録商標として使用している場合があります 。**

PGECons

PostgreSQL Enterprise Consortium