SRA OSS, INC.

OSS-DB
LPI-Japan
Approved
Training Partner

【全2回】OSS-DB Exam Silver 技術解説セミナー

（前半）2020-05-19

（後半）2020-05-26

SRA OSS, Inc. 日本支社
OSS事業本部 技術部 データベース技術グループ
千田 貴大

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

目次

- **OSS-DB技術者認定試験**
- **PostgreSQL とは**
- **環境作り**
  - インストール
  - 環境変数
  - データベースクラスタ
- **基本操作**
  - 起動/停止
  - ユーザ作成
  - データベース作成
  - psql
- **設定ファイル**
- **VACUUM/ANALYZE**
- **バックアップ・リストア**
  - PITR

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

OSS-DB技術者認定資格

オープンソースデータベース（OSS-DB）に関する技術と知識を認定するIT技術者認定

OSS-DB /Silver

データベースシステムの設計・開発・導入・運用ができる技術者

OSS-DB /Gold

大規模データベースシステムの
改善・運用管理・コンサルティングができる技術者

OSS-DB技術者認定資格の必要性

商用/OSSを問わず様々なRDBMSの知識を持ち、データベースの構築、運用ができる、
または顧客に最適なデータベースを提案できる技術者が求められている

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

OSS-DB Exam Silver出題範囲

- **一般知識（16%）**
  - OSS-DBの一般的特徴
  - リレーショナルデータベースに関する一般知識
- **運用管理（52%）**
  - インストール方法
  - 標準ツールの使い方
  - 設定ファイル
  - バックアップ方法
  - 基本的な運用管理作業
- **開発/SQL（32%）**
  - SQLコマンド
  - 組み込み関数
  - トランザクションの概念

OSS-DB Exam Ver.2.0はPostgreSQL 10以上を基準のRDBMSとして採用
※2019年4月現在、11まで対応

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

～ PostgreSQL とは ～

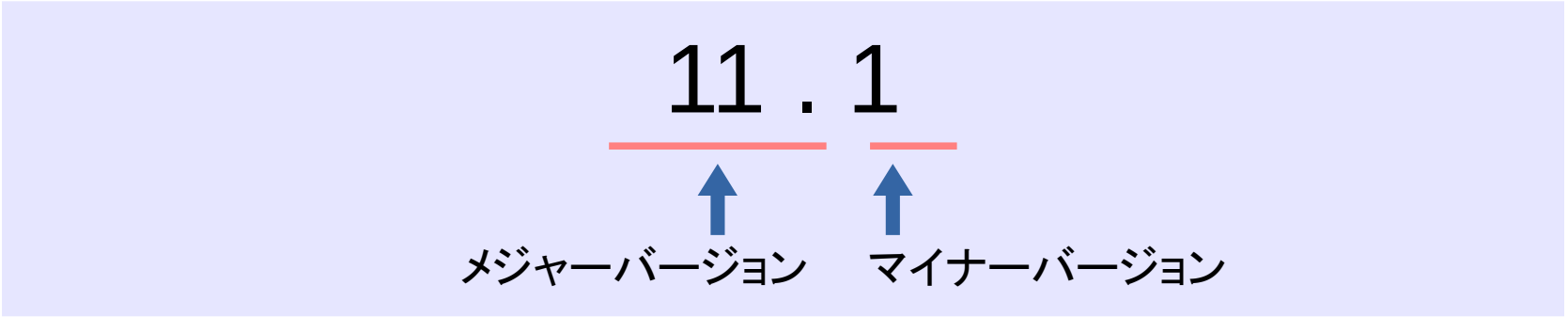
Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

PostgreSQL とは (1)
特徴

- **標準SQLの大部分とその他先進的な機能を サポートする本格的なオープンソースRDBMS**
- **20年以上の歴史を持ち、現在も活発な開発体制**
  - 1年に1度のメジャーバージョンアップ
  - 1年に数回のマイナーバージョンアップ
- **豊富なプラットフォームに対応**
  - Unix系OS全般、Windows OS、Mac OS
- **豊富なサポート言語**
  - C, ECPG, C++, Java, Tcl/TK, Python, Perl, PHP, Ruby, ODBC, .NET Data Providerなど

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

PostgreSQL とは (2)
バージョン番号規則



- **メジャーバージョンアップでは仕様の追加・変更**
  - 移行には付属コマンドでのバックアップ・リストアが必要
- **マイナーバージョンアップでは主にバグ修正**
- **最初のメジャーバージョンリリースから5年でEOLを迎える**

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

PostgreSQL とは (3)
コミュニティの役割

- **PostgreSQL公式開発コミュニティ**
  - 仕様検討、開発、リリース、不具合報告
- **日本 PostgreSQL ユーザ会（JPUG）**
  - 迅速な最新ドキュメント翻訳
- **PostgreSQL エンタープライズ・コンソーシアム（PGECons）**
  - 毎年テーマ別に検証および結果公表
  - 公式開発コミュニティへの改善リクエスト

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

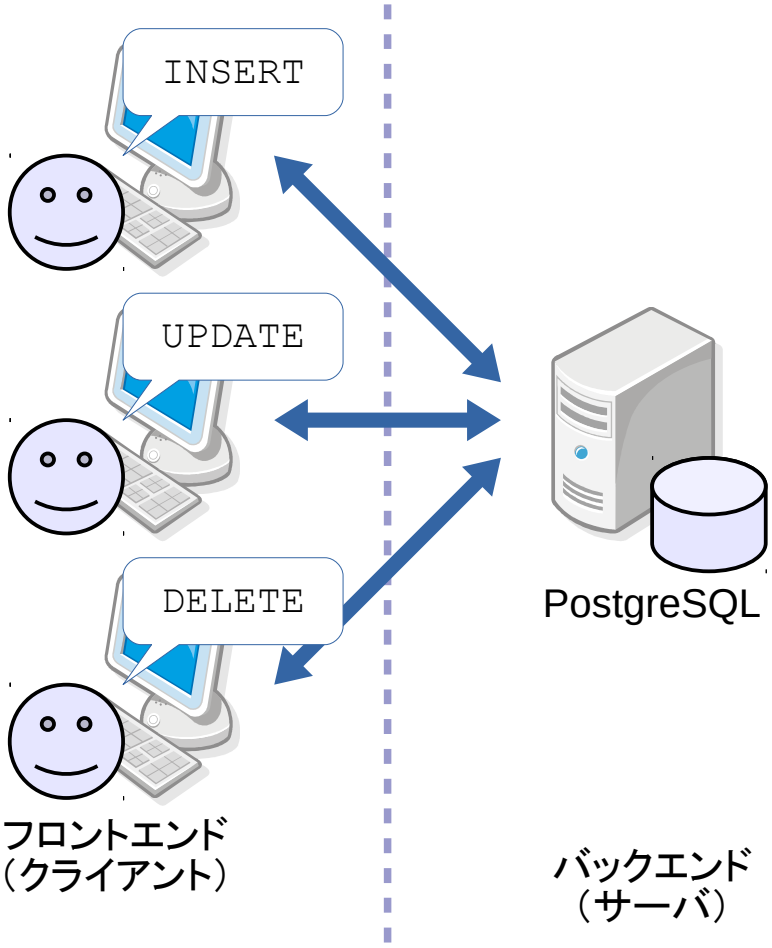
PostgreSQL とは (4)
ライセンス

- **PostgreSQLライセンス**
  - BSDライセンスに類似
    - 広告条項はなし（修正BSDライセンス）
  - 使用、複製、改変、配布の自由
    - 複製においては以下を含めることが条件
      - 著作権表示
      - ライセンス条文
      - 免責条項
    - 具体的には、PostgreSQLソースコードに添付されているCOPYRIGHTファイルを配布物に含めるか、その内容をマニュアルに印刷すればよい

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

PostgreSQL とは (5)
クライアント・サーバ構成

- **libpqプロトコル**
  - クライアント・サーバのOSの違いを吸収
- **軽量クライアント**
  - サーバの変更に影響されにくい



Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

PostgreSQL とは (6)
マルチバイト文字対応

- **文字エンコーディング**
  - データベースごとに指定
  - フロントエンド（バックエンドとの通信）ごとに変更できる

- **日本語を扱う一般的な組み合わせ**

バックエンド （データベース）	フロントエンド （クライアント）
UTF8	UTF8, EUC_JP, SJIS
EUC_JP	UTF8, EUC_JP, SJIS

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

～ 環境作り ～

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

環境作り (1)
インストール

- **Linux**
  - ソースコードからのインストール
  - パッケージによるインストール
    - ディストリビュータ配布のパッケージ
    - PostgreSQL開発元配布のパッケージ
- **Windows**
  - one click installerによるインストール

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

環境作り (2)
インストール

- ソースコードからのインストール
  - 前準備
    - ソースコードの入手
      - https://www.postgresql.org/ftp/source
    - postgres ユーザの作成
    - ソースコードの展開
  - configure
  - make
  - (make check)
  - make install

```
# useradd postgres
```

```
# tar xf postgresql-11.6.tar.bz2 -C /usr/local/src
```

```
# cd /usr/local/src/postgresql-11.6
# ./configure
```

```
# make check
```

```
# make world
```

```
# make install-world
```

- ※インストール先は、/usr/local/pgsql（デフォルト）

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

環境作り (3)
インストール

- パッケージからのインストール

https://www.postgresql.org/download/

- postgres ユーザが作成される
  - ホームディレクトリは /var/lib/pgsql
  - 環境変数の設定 /var/lib/pgsql/.bash_profile
    - 既存の.bash_profile は上書きされる
- インストール先ディレクトリ
  - /usr/pgsql-11
- セットアップスクリプト
  - /usr/pgsql-11/bin/postgresql11-setup

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

環境作り (4)
インストール

- **Windowsでのインストール**

  https://www.postgresql.org/download/windows

  - **EnterpriseDB 社が提供している**
  - **導入が簡単な one click installer**
    - **Windows版 PostgreSQL サーバ/クライアント**
    - **PgAdmin4**
    - **Stack Builder**
  - **データベースクラスタの作成**
  - **Windowsサービスとして動作**
  - **データベースのスーパユーザ "postgres" の作成**
  - **ディレクトリ構成**
    - **インストール時に指定したディレクトリ**
      - C:¥Program Files¥PostgreSQL¥11¥
    - **データベースクラスタ**
      - C:¥Program Files¥PostgreSQL¥11¥data¥

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

環境作り (5)
環境変数

- PostgreSQL コマンドディレクトリをコマンドパスに追加
  - ~/.bash_profileに以下を追加
    - /usr/local/pgsql （ソースコードからインストールした場合、デフォルトのインストール先）にインストールした場合の例

```
$ vi ~/.bash_profile
PGHOME=/usr/local/pgsql
export PATH=$PGHOME/bin:$PATH
export LD_LIBRARY_PATH=$PGHOME/lib:$LD_LIBRARY_PATH
export MANPATH=$PGHOME/share/man:$MANPATH
export PGDATA=/usr/local/pgsql/data
```

  - 設定の反映
    - 反映後 psql などのコマンドにパスが通っていることを確認

```
$ . ~/.bash_profile
```

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

環境作り (6)
データベースクラスタ

- $PGDATAで指定したディレクトリを作成し、
  データベースクラスタの初期化を行う

```
[postgres]$ initdb --encoding=UTF8 --no-locale
```

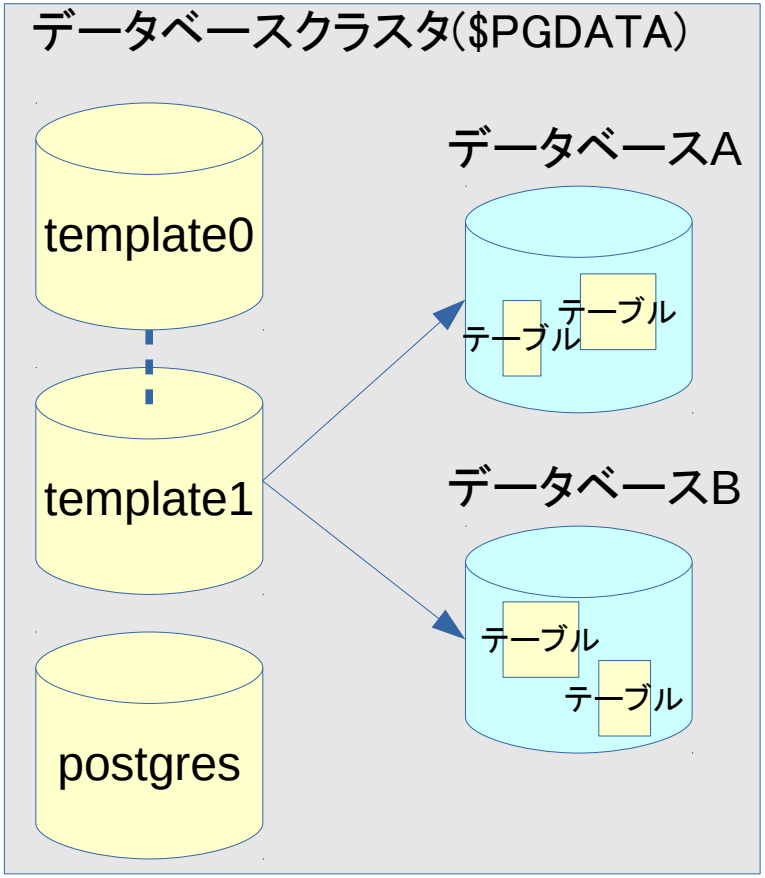
- --encoding=UTF8
  - デフォルトの文字エンコーディングをUTF8に設定
- --no-locale　または --locale=C
  - デフォルトのロケールを「利用しない」に設定

    主に日本語と英語を格納するならロケールは不要

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

環境作り (7)
データベースクラスタ

- initdbを実行したユーザがPostgreSQLのスーパーユーザになる
- テンプレートデータベース（template0, template1)とpostgres データベースが作成される
  - 実際に使用すべきデータベースではない
    - template0は書き込み不可
    - template1は書き込み可



Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

～ 基本操作 ～

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

基本操作 (1)
起動/停止

- pg_ctlコマンド
  - 起動

```
$ pg_ctl start [-D データベースクラスタパス]
```

  - 停止

```
$ pg_ctl stop
```

    - 停止モード
      - fast mode　→　直ちに終了（デフォルト）
      - smart mode　→　全クライアントの接続終了を待つ
      - immediate mode　→　適切な終了処理をせず直ちに終了
  - その他
    - 再起動、リロード ...etc.
    - --help で確認可能

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

基本操作 (2)
ユーザ作成

- PostgreSQLコマンド
  - createuser
  - -d オプションでデータベース作成権限を与えます

```
$ createuser --help
$ createuser -d ロール名
```

- SQL文
  - CREATE ROLE
  - CREATEDB でデータベース作成権限を与えます
  - LOGIN でログイン権限を与えます
    - (createuser の場合デフォルトで付与されています)

```
=# CREATE ROLE ロール名 CREATEDB LOGIN
```

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

基本操作 (3)
データベース作成

- PostgreSQLコマンド
  - createdb
    - データベース名を省略すると、コマンドを実行している OSのユーザと同じ名前のデータベースを作成

```
[postgres]$ createdb   データベース名
```

- SQL文
  - CREATE DATABASE

```
=# CREATE DATABASE データベース名
```

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

基本操作 (4)

psql

- **psql　＝　SQL発行ツール**
  - psqlの起動方法

```
psql [ オプション ] . . [ DB 名 [ ユーザ名 ] ]
```

  - 例

```
$ psql postgres <一般ユーザ名>
psql (11.6)
Type "help" for help.
postgres=>
```

接続しているデータベース名

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

基本操作 (5)

psql

- プロンプト
  - 「=#」… 接続しているユーザはスーパーユーザ
  - 「=>」… 接続しているユーザは一般ユーザ
    - 2行目以降は「-#」「→」

```
userdb=> SELECT * FROM --ここにコメントが書けます
userdb-> pg_user;      --ここにコメントが書けます
```

- SQLの発行
  - 「;」でSQLの終わりを意味し、コメントは「--」
  - SQL構文中の空白やタブ、改行は一つの空白と扱う
  - SQL構文の大／小文字は区別されない　日本語も可
  - Linuxでは「Ctrl-C」で入力をキャンセル
  - psqlの終了方法 … 「¥q」または「Ctrl-D」

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

基本操作 (6)

psql

- **psqlコマンドオプション**

-h ホスト名	データベースホスト名を指定
-p ポート番号	TCP/IPポートの指定
-d データベース	接続するデータベース名を指定
-U ユーザ名	接続ユーザ名を指定
-c “SQL文”	引数で与えたSQL文（１つ）を実行させる
-f ファイル名	ファイルに記述してあるSQLコマンドを受け付け
-o ファイル	出力結果をファイルに格納する
-l	データベースの一覧を表示
-t	カラム名と行数フッターなどの表示を無効にする
-V	psqlのバージョンを表示（--version）
-?	psqlのヘルプを表示（--help）

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

基本操作 (7)

psql

- バックスラッシュコマンド

\?	コマンド一覧
\encoding	エンコーディングの表示・設定
\h [SQL 文]	SQL 文のヘルプ
\l	データベース一覧
\d[t|i|s|v|S]	テーブル、インデックス、シーケンス、ビュー、システムテーブルの一覧
\d [NAME]	指定したテーブル、インデックス、シーケンス、viewを表示
\d+ [NAME]	指定したオブジェクトの詳細を表示、view定義を表示
\dp (\z)	テーブル、ビュー、シーケンスのアクセス権限一覧
\du	ロール(データベースユーザ)一覧
\df	関数一覧
\ef [NAME]	関数編集、または作成
\x	テーブルの表示モードを変更

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

～ 設定ファイル ～

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

設定ファイル (1)

データベースクラスタ（$PGDATAディレクトリ）

```
PG_VERSION
base
global
pg_xacts
pg_dynshmem
pg_hba.conf
pg_ident.conf
pg_log
pg_logical
(省略)
pg_stat_tmp
pg_subtrans
pg_tblspc
pg_twophase
pg_wal
postgresql.auto.conf
postgresql.conf
postmaster.opts
postmaster.pid
```

```
$ ls $PGDATA
PG_VERSION      pg_logical    pg_subtrans
base            pg_multixact  pg_tblspc
global          pg_notify     pg_twophase
pg_xact         pg_replslot   pg_wal
pg_dynshmem     pg_serial     postgresql.auto.conf
pg_hba.conf     pg_snapshots  postgresql.conf
pg_ident.conf   pg_stat       postmaster.opts
pg_log          pg_stat_tmp   postmaster.pid
```

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

設定ファイル (2)
postgresql.conf

- postgresql.conf
  - 書式

```
変数　＝　値
```

```
$ vi $PGDATA/postgresql.conf
```

  - 設定の反映

```
$ pg_ctl reload
```

```
$ pg_ctl restart
```

(change requires restart)
の記述があるパラメータを変更した場合

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

設定ファイル (3)
postgresql.conf

- **postgresql.conf 設定項目抜粋**

listen_addresses	接続を受け付けるIPアドレスを記述。 * なら全てのIPインターフェイスで受付。空ならUNIXドメイン接続のみ。
port	接続ポート番号。デフォルトは 5432 。
client_encoding	クライアント側文字エンコーディングのデフォルトを指定。
max_connections	データベースサーバへの同時接続の最大数。
shared_buffers	共有メモリバッファをメモリ容量またはページ数で指定。デフォルトは 128MB 。
log_destination	ログの出力先を指定。(stderr, syslog, eventlog, csvlog)
logging_collector	stderr に出力したエラーを $PGDATA/log 以下のローテーションするログファイルにリダイレクトする。
log_line_prefix	ログの各行の先頭に出力する書式文字列を指定。例えば、'%m [%p] ' と指定するとログの時刻とプロセスIDが出力。
log_connections	クライアントからサーバへの接続試行をログに出力。
log_disconnections	クライアントのサーバ接続終了をログに出力。

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

設定ファイル (4)
postgresql.conf

- **SQLで閲覧・設定もできる**
  - SHOW/SET
  - 設定されたパラメータは接続セッション中のみ有効

- **実行時に変更できる（GUC変数）**
  - **一部のパラメータに限る**
  - client_encodingなど
- **reloadが必要**
  - log_connectionsなど
- **再起動が必要**
  - shared_buffersなど

```
SET name TO value;
```

```
SHOW name;
```

```
$ psql
postgres=# SHOW listen_addresses;
 listen_addresses
------------------
 *
(1 row)

postgres=# SHOW ALL;
(省略)
```

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

設定ファイル (5)
pg_hba.conf

- pg_hba.conf
  - クライアント認証
    - データベースサーバを起動時に読み込まれ、
      複数行ある場合は、上から評価される
    - 1行に1つの設定を記述
      - 接続タイプ、データベース名、ユーザ名、IPアドレス範囲
      - 認証方法とそのオプション
    - 接続しようとしているクライアントに関する設定がない場合、
      アクセスは拒否される
  - pg_ctl reloadで修正を反映

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

設定ファイル (6)

pg_hba.conf

- **接続方式**

local	ローカル接続（UNIXドメイン接続）の場合に対応
host	ホスト接続（TCP/IP）の場合に対応。ホストを指定した場合が該当する。SSL接続でも通常の接続でもどちらでもよい
hostssl	ホスト接続でSSL接続である場合に対応
hostnossl	ホスト接続でSSL接続でない場合に対応

- **データベース名/ユーザ名**
  - カンマ区切りで複数指定可能

dbname 接続するデータベース	
all	すべてのデータベース
sameuser	接続ユーザと同じ名前のデータベース
samerole （samegroup）	接続ユーザが属しているグループと同じ名前のデータベース
@filename	ファイルに書いてあるデータベース

user 接続時のPostgreSQLのユーザ	
all	すべてのユーザ
+groupname	指定したグループに属しているメンバ
@filename	ファイルに書いてあるユーザ

c. Japan All rights reserved.

設定ファイル (7)
pg_hba.conf

- IPアドレス範囲
  - CIDR-address(CIDR記法によるアドレス)
    - ホスト接続のクライアントIPアドレス範囲
    - local指定のときは空欄とする
    - ホスト名での指定も可能
  - 例：ローカルホストからのTCP/IP接続のみに限定する

```
127.0.0.1/32
```

    - 代わりにIP-addressとIP-maskによる記述も可能

```
192.168.128.0  255.255.255.0
```

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

設定ファイル (8)
pg_hba.conf

- **認証方式**

auth-type	
trust	無条件で許可
reject	無条件で拒否
scram-sha-256	SCRAM-SHA-256認証
md5	MD5暗号化パスワード認証
password	平文パスワード認証
gss	GSSAPI認証
sspi	SSPI認証
peer	Peer認証
ident	Ident認証
ldap	LDAP認証
radius	RADIUS認証
cert	SSLクライアント証明書認証
pam	PAM認証

※ SCRAM-SHA-256認証とは、チャレンジ/レスポンス認証の一種

※ gss は、接続形式がlocalの場合は使用できません

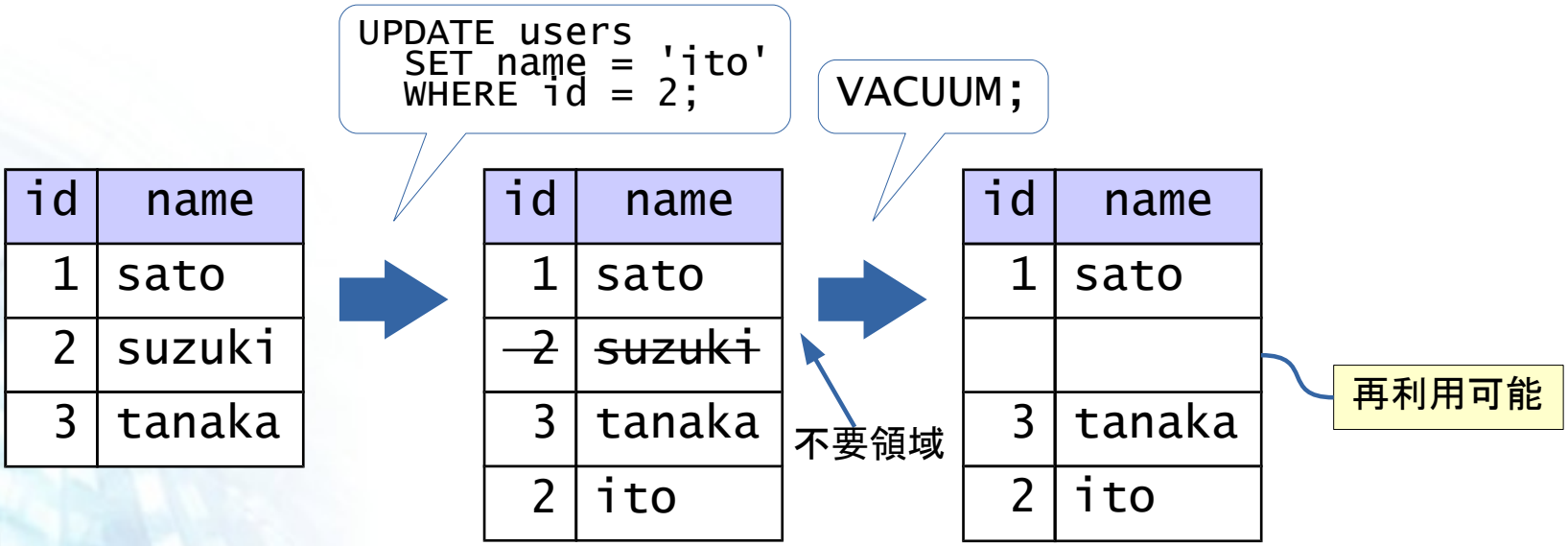
Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

～ VACUUM/ANALYZE ～

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

VACUUM/ANALYZE (1)
VACUUM

- VACUUMとは
  - 不要領域の削除（ガベージコレクション）
    - 更新しても古い行は残す仕組み
    - どのトランザクションからも参照されない行は、無駄データなのでVACUUMコマンドで削除する
      - 削除しないと、ファイルの肥大化し、パフォーマンスが低下



Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

VACUUM/ANALYZE (2)
VACUUM

- VACUUM実行方法
  - PostgreSQLコマンド

```
postgres$ vacuumdb [-t テーブル名] [データベース名]
```

  - SQL文

```
userdb=# VACUUM [テーブル名];
```

    - 不要となった行を探し、再利用できるようにする
    - ファイルサイズは減らない

- VACUUM FULL
  - ファイルサイズを縮める
  - 実行中はSELECTもできなくなるので、運用中には使えない

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

VACUUM/ANALYZE (3)
ANALYZE

- **ANALYZEとは**
  - **統計情報の更新が必要**

```
userdb=# ANALYZE [テーブル名];
```

  - **インデックスを使うかどうかは、統計情報による**
  - **ANALYZEで統計情報を更新する**
    - **ANALYZEしないと最適な検索方法が使用されない**
    - **テーブルの内容が大幅に変わった後には、ANALYZE実行を推奨**

```
postgres$ vacuumdb -az
```

すべてのデータベースで
VACUUMとANALYZEを実行する

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

VACUUM/ANALYZE (4)

自動バキューム

- 自動バキューム（autovacuum）
  - postgresql.confで設定
    - デフォルトでon

```
#autovacuum = on
```

  - データベースの更新量に合わせて、適宜VACUUMやANALYZEを自動で実行
    - データベース更新量を把握するには、統計情報コレクタの機能を使用
    - その分の負荷はわずかながら余分にかかる

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

～ バックアップ・リストア ～

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (1)

- バックアップ戦略
  - pg_dumpコマンドによる論理バックアップ
  - データベースクラスタ全体の物理バックアップ
  - PITR（Point-In-Time Recovery）

- その他
  - レプリケーション機能の利用
    - PostgreSQLのストリーミングレプリケーション
    - レプリケーションソフトウェアの利用
  - 外部ツールを使う

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

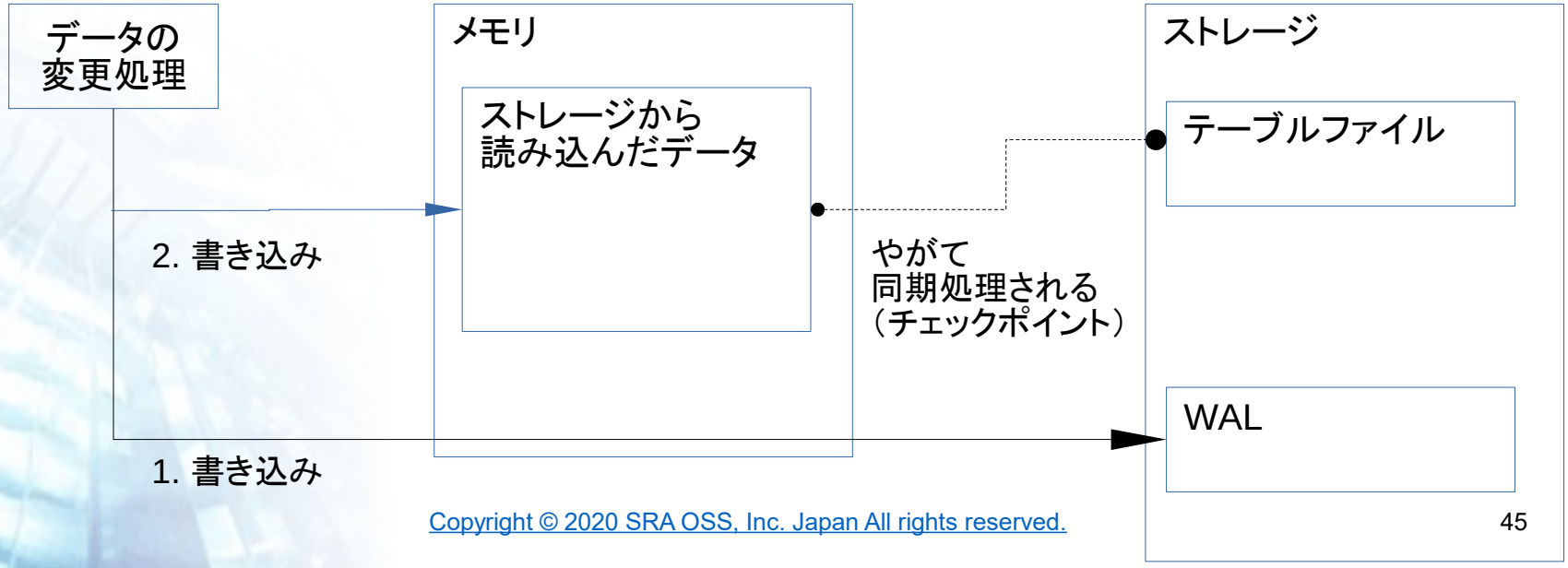
バックアップ・リストア (2)
PITR

- PITRとは
  - Point In Time Recovery
    - ベースバックアップとアーカイブログを使って
      データベースを最新の状態までリストアする手法
  - ベースバックアップ
    - データベースクラスタの物理コピー
    - pg_basebackup コマンドを使用
  - アーカイブログ
    - WALの物理コピー

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (3)
PITR

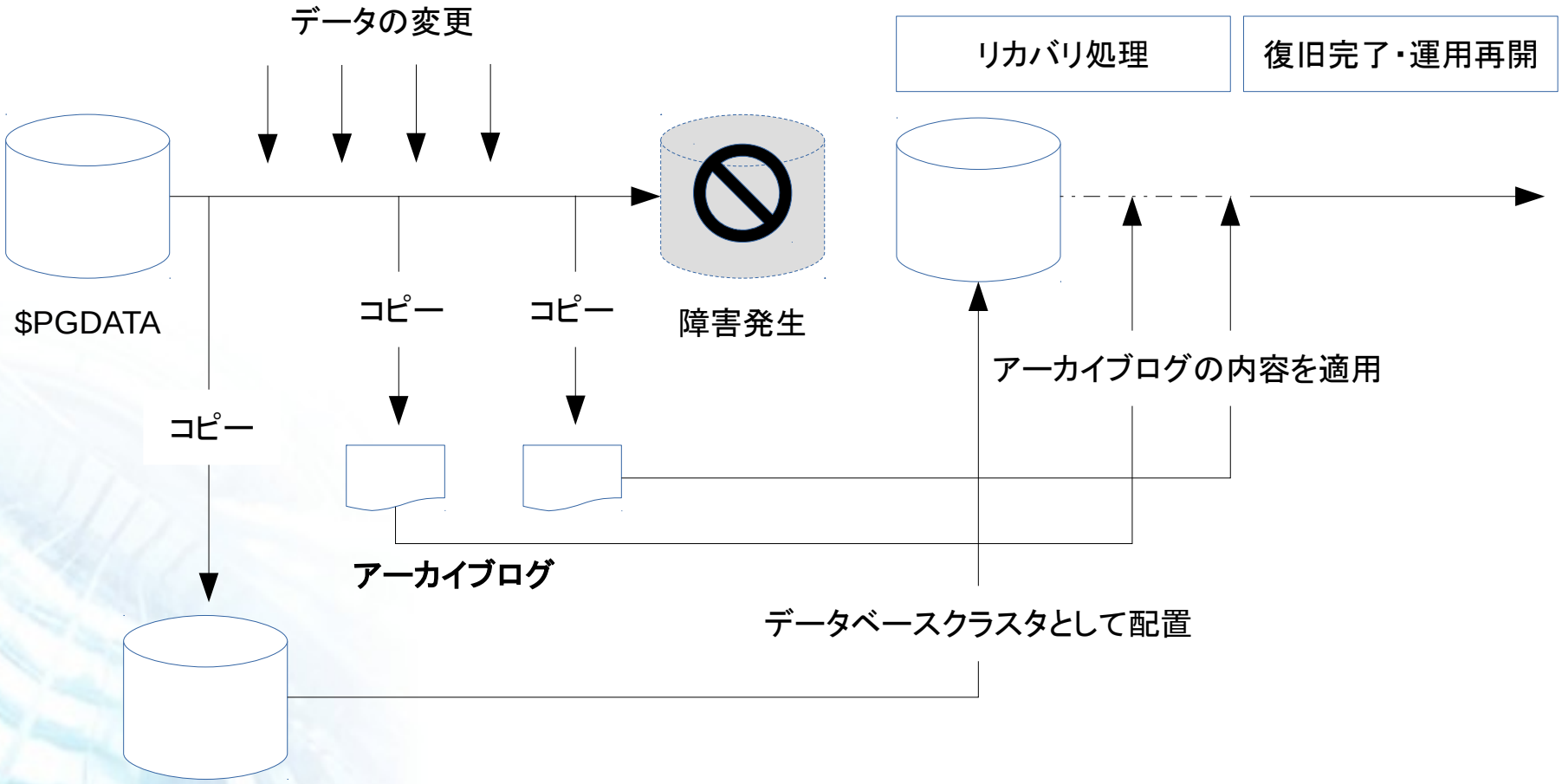
- PostgreSQLのデータ書き込み
  - データのやり取りはメモリを介す
  - メモリの内容はいずれストレージに同期される
  - WAL = データの保全性と書き込み速度を担保する仕組み
    - データの変更内容をまず WAL に書き込んでからメモリに書き込む
    - 同期処理前に PostgreSQL がクラッシュしたら WAL から復旧できる
  - 同期処理によって不要となった WAL は通常削除される



Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (4)
PITR

- **PITR全体の流れ**



Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (5)
PITR

- **準備**
  - PITRで使用する
    ディレクトリを作成

```
$ su -
# mkdir /mnt/pg_arc
# chown postgres /mnt/pg_arc
$ su - postgres
$ mkdir /mnt/pg_arc/base
$ mkdir /mnt/pg_arc/log
```

- **WALアーカイブ設定**
  - postgresql.conf を編集

```
# wal_level = replica
# max_wal_senders = 10
archive_mode = on
archive_command = 'cp "%p" "/mnt/pg_arc/log/%f"'
```

%p WALの相対パスに置き換わる
%f WALのファイル名に置き換わる

```
$ pg_ctl restart
```

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (6)
PITR

- ベースバックアップ
  - pg_basebackup（PostgreSQLコマンド）
    - データベースサーバを停止せずに
      物理バックアップを取得するコマンド
    - バックアップモードへの移行を自動的に行い
    - データベースクラスタ全体をコピー
      - スーパーユーザ(postgres)で実行
      - PostgreSQLのレプリケーション接続を経由してコピーを取得
      - あらかじめサーバ側へ
        ストリーミングレプリケーションと同等の設定が必要

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (7)
PITR

- **pg_basebackup に必要な設定**
  - postgresql.conf
    - wal_level
      - WALをどのくらい詳細に出力するか指定
      - replica … ストリーミングレプリケーション機能や pg_basebackup を利用する場合
    - max_wal_senders
      - WAL送信プロセスの最大数

```
$ cd $PGDATA
$ vi postgresql.conf
$ vi pg_hba.conf
$ pg_ctl restart
```

```
# wal_level = replica
# max_wal_senders = 10
```

  - pg_hba.conf（クライアント認証設定）
    - local接続でreplication(仮想データベース)へpostgresロールでの接続について無条件で接続を許可

```
local   replication   postgres                    trust
```

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (8)
PITR

- ベースバックアップの作成

```
$ pg_basebackup -D /mnt/pg_arc/base/001 -P -Xn
46282/46282 kB (100%), 1/1 tablespace
$ ls /mnt/pg_arc/base/001
```

- -D　　バックアップ先ディレクトリを指定
- -P　　進行状況を表示
- -h, -p　　接続先ホスト、ポートを指定
  （デフォルトではlocalの5432ポートへ接続）
- -Xn　　WAL ファイルを含めない

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (9)
PITR

- リカバリ
  - ベースバックアップの展開

```
$ pwd
/usr/local/pgsql
$ mv data data_crash
$ cp -r /mnt/pg_arc/base/001 data
```

  - リカバリ設定ファイル作成

```
$ vi recovery.conf
```

```
restore_command = 'cp "/mnt/pg_arc/log/%f" "%p"'
```

%p WALの相対パスに置き換わる
%f WALのファイル名に置き換わる

    - リカバリーモード
      - リカバリ設定ファイルrecovery.confが
        データベースクラスタ内に存在している状態で起動
      - restore_command
        - アーカイブされたWALファイルを取得するためのコマンド
        - アーカイブ領域に存在する最新のWALファイルまでリカバリ

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (10)
PITR

- リカバリ開始前に
  - 最後に稼働していたデータベースクラスタ内のアーカイブされていないWALファイルをリカバリ先データベースクラスタ内のWAL領域へ手動コピー

```
$ cd /usr/local/pgsql/data_crash
$ ls pg_wal
$ ls /mnt/pg_arc/log
$ cp pg_wal/* $PGDATA/pg_wal/
```

- リカバリ開始
  - リカバリモードで起動

```
$ pg_ctl start
```

  - リカバリ中はデータベースに接続できない
    - 別端末で実行（サーバがリカバリ中は、以下の出力になる）

```
$ psql
psql: FATAL: the database system is starting up
```

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

バックアップ・リストア (11)
PITR

- リカバリ終了後の確認
  - recovery.conf の名称が recovery.done に変わる
  - ログに “archive recovery complete” が出力
  - 確認用に作成したテーブルが復旧 (あれば)

Copyright © 2020 SRA OSS, Inc. Japan All rights reserved.

ご清聴ありがとうございました。

■お問い合わせ■
SRA OSS, Inc. 日本支社
OSS事業本部 マーケティング部
pub@sraoss.co.jp