LPI-JAPAN
Linux Professional Institute Japan

OSS-DB Exam Silver 技術解説無料セミナー

2011/7/25

特定非営利活動法人エルピーアイジャパン
テクノロジー・マネージャー
松田　神一

Agenda

OSS-DB

- はじめに
- データベースとは何か
- PostgreSQLのインストール
- SQLによるデータ操作の基本
- データベース運用管理の基礎
- OSS-DB Exam Silverの例題

自己紹介

OSS-DB

- 松田　神一（まつだ　しんいち）
  LPI-JAPAN テクノロジー・マネージャー

- NEC、オラクル、トレンドマイクロなどで約20年間、ソフトウェア開発に従事（専門はアプリケーション開発）
  うち10年間はデータベース、およびデータベースアプリケーションの開発（Oracle、C言語、SQL言語）

- 2010年7月から現職

今日のゴール

OSS-DB

■データベース初心者の方

- RDBMSってどんなもの？
- どうやって使うの？
- どうすれば学習できる？
- 何を学習すべき？

■データベースについて知識のある方

- OSS-DB技術者認定試験の受験準備のために何をすべきか

データベースとは何か

データベースとは何か

OSS-DB

- データを格納し、検索、更新などの機能を提供するソフトウェア

- 厳密に言えば、データベース(DB)は、管理されているデータの集合を指し、データを管理するソフトウェアは、データベース管理システム(DBMS)と呼ばれるが、DBはDBMSを包含する言葉として使われることが多い

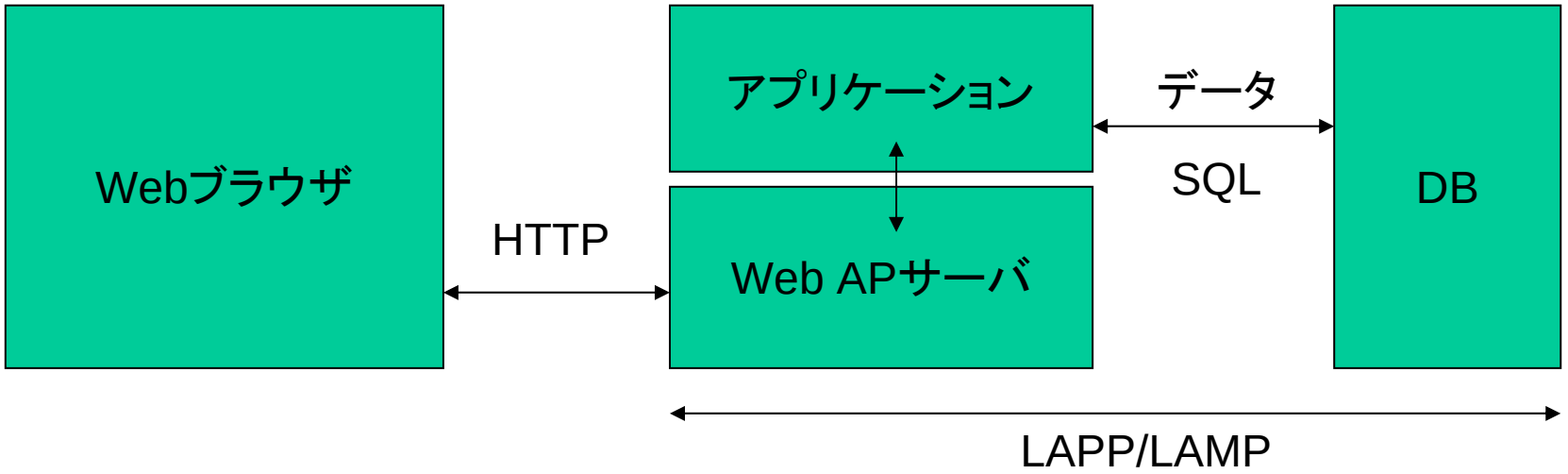
- データの管理をデータベースに任せることで、アプリケーションはデータの処理に専念できる

- データ管理機能
  - データ入出力、データ加工
  - マルチユーザサポート、データの一元管理
  - アクセス権限の管理、機密保護
  - データのバックアップ、リストア、リカバリ

データベースアプリケーションの例

OSS-DB

- Webアプリケーションが必要とするデータは、データベースが管理する
- アプリケーションは、HTTPで受け取ったデータを処理、必要に応じてDBとデータの入出力を実行
- アプリケーションはHTMLデータを生成し、Webブラウザを介してユーザ画面に表示される



データベースの種類

OSS-DB

- 格納するデータの種類、データの格納形式（データ構造）により分類される
  - リレーショナルデータベース（RDBMS）
  - カード型データベース
  - オブジェクト型データベース
  - 階層型データベース
  - KVS（Key Value Store）など新しいタイプのデータベース（NoSQL）
- 目的によって採用するデータベースの種類が異なる
- 最も汎用で、広く使われているのがRDBMS
  - 単にDB、DBMSと言ったとき、RDB、RDBMSを指すことが多い

RDBMSの種類

OSS-DB

主な商用RDBMS

- Oracle
- SQL-Server（Microsoft）
- DB2（IBM）

主なオープンソースRDBMS（OSS-DB）

- PostgreSQL
- MySQL
- Firebird

商用DBとOSS-DBの違い

OSS-DB

■最大の違いはソースコードが公開されているかどうか

- (その気になれば) 自分で調査ができる
- OSSでも製品によりライセンス形態は様々 (BSD/GPL/LGPL)
  - 特に、ソースを改変したときの再配布の制限に注意

■OSSは無償？　サポートが受けられない？

- 有償のサポートが受けられる製品も多い
- OSSでも有償の製品もある

■機能的には...

- 商用DBに一日の長があるが、その差は小さくなってきている
- 通常の用途であれば、OSS-DBでも全く遜色がない
- ライセンスのコストを削減するため、商用DB→OSS-DBへの移行が進んでいる

RDBMSの共通点と違い

OSS-DB

共通点

- DBMSとしての各種機能
  - データ管理／入出力
  - ユーザ管理
  - アクセス権限管理、セキュリティ
  - バックアップ、リカバリ
  - 分散DB（レプリケーション）
- SQL言語（ANSI/ISOで標準化）

違い

- 各種機能の使い方
  - コマンドとオプション
  - 設定ファイルとパラメータ
- SQLの方言
- 独自拡張機能

データベース学習のヒント

OSS-DB

- どの製品にも共通の機能もあれば、同じ機能でも製品によって
  実行方法の異なるもの、特定の製品にしかない機能もある

- まずはDBの種類による差分はあまり気にせずに、
  特定のDBについて学習し、マスターする

  次のステップは…

- 横展開
  他のDBについて、最初に学習したDBとの差分に注意しながら学習する

- 深掘り
  その製品のエキスパートとなるべく、更に深く学ぶ

OSS-DB技術者認定が必要な理由

OSS-DB

- 使う前に設定が必要
  - ユーザ
  - アクセス権
  - テーブル
- 重要な用途
  - 基幹業務での利用
  - バックアップ
  - セキュリティ
- 複雑な用途
  - 分散DB
  - パフォーマンスチューニング
  - トラブルシューティング
- 製品による違い
  - 一般論だけ学んでも、現場で活躍できない

OSS-DB技術者認定試験の概要

OSS-DB

■認定の種類

- Silver (ベーシックレベル)
- Gold (アドバンストレベル)

■Silver認定の基準

- データベースの導入、DBアプリケーションの開発、DBの運用管理ができること
- OSS-DBの各種機能やコマンドの目的、使い方を正しく理解していること

■Gold認定の基準

- トラブルシューティング、パフォーマンスチューニングなどOSS-DBに関する高度な技術を有すること
- コマンドの出力結果などから、必要な情報を読み取る知識やスキルがあること

OSS-DB Exam Silverの出題範囲

OSS-DB

- 一般知識 (20%)
  - OSS-DBの一般的特徴
  - ライセンス
  - コミュニティと情報収集
  - RDBMSに関する一般的知識
- 運用管理 (50%)
  - インストール方法
  - 標準付属ツールの使い方
  - 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 (30%)
  - SQLコマンド
  - 組み込み関数
  - トランザクションの概念

出題範囲に関する注意事項

OSS-DB

- 最新の出題範囲は
  http://www.oss-db.jp/outline/examarea.shtml
  で確認できる

- 前提とするRDBMSはPostgreSQL 9.0

- SilverではOSに依存する問題は出題しないが、記号や用語がOSによって異なるものについては、Linuxのものを採用している
  - OSのコマンドプロンプトには ＄ を使う
  - 「フォルダ」ではなく「ディレクトリ」と呼ぶ
  - ディレクトリの区切り文字には / を使う

- 出題範囲に関するFAQ
  http://www.oss-db.jp/faq/#n02

PostgreSQLのインストール

PostgreSQLのインストール

OSS-DB

インストールに必要な環境

- インターネットにつながっているマシン (Windows/Mac/Linux)

おススメの環境

- ある程度、Linuxの知識がある方にはLinuxを使うことを勧める。
  各種情報の入手の容易さを考えるとCentOSが良い。
- VMware Playerなどを使えば、Windows PC上に仮想Linux環境を構築し、
  そこにPostgreSQLをインストールして学習することができる。
- 仮想環境の良い点は、それを破壊しても、簡単に最初からやり直せるところ
- もちろん、WindowsやMacの環境に直接、PostgreSQLを
  インストールするのでもOK。

参考書などを読むだけでは、十分な学習をすることはできません。
自分専用の環境を作り、そこでいろいろ試すことで学習してください。

PostgreSQLのインストールと初期設定

OSS-DB

インストール方法

- ソースコードから自分でビルドしてインストール
- ビルド済みのパッケージをインストール（様々なビルド済みパッケージがある）

インストール後の初期設定

- データベースのスーパーユーザ（postgresユーザ）の作成
- 環境変数（PATH, PGDATAなど）の設定
- データベースの初期化（データベースクラスタの作成）
- データベース（サーバープロセス）の起動
- データベース（サーバープロセス）起動の自動化

インストール方法によっては、初期設定の一部が自動的に実行される

インストール方法によって、プログラムがインストールされる場所、データベースファイルが作られる場所が大きく異なるので注意

ワンクリックインストール

OSS-DB

- Windows/Mac/Linuxいずれでも利用可能
  - EnterpriseDB社のサイト（別紙1）から、ビルド済みのパッケージをダウンロードしてインストールする
    http://www.enterprisedb.com/products-services-training/pgdownload
  - GUIの管理ツール（pgAdmin III）も同時にインストールされる
  - ApacheやPHPなど、PostgreSQLと一緒に使われるソフトウェアも、同時にインストール可能
  - Windowsではワンクリックインストールの利用を推奨
- インストールガイド（英語）は
  http://www.enterprisedb.com/resources-community/pginst-guide
  （別紙2）
- 多くの項目はデフォルト値のままで良い
  - スーパーユーザ（postgres）のパスワードの設定を求められるので、適切に設定し、それを忘れないようにすること
  - ロケール（Locale）の設定を求められるが、"Default locale"となっているのを"C"に変更することを推奨する
  - インストール終了時にスタックビルダ（Stack Builder）を起動するかどうか尋ねられるが、ここはチェックボックスを外して終了してよい。
    必要なら後でスタックビルダを起動することができる

ワンクリックインストール後の初期設定

OSS-DB

- postgresユーザは自動的に作成される。
- データベースの初期化、起動はインストール時に実行されるので、
  インストール後、すぐにデータベースに接続できる。
- データベースの自動起動の設定がされるので、
  マシンを再起動したときもデータベースが自動的に起動する。
- 【Windowsでは】
  C:¥Program Files¥PostgreSQL¥9.0 の下にインストールされる。
  データベースは C:¥Program Files¥PostgreSQL¥9.0¥data の下に作られる。
  環境変数PATHに C:¥Program Files¥PostgreSQL¥9.0¥bin を追加するか、
  あるいは C:¥Program Files¥PostgreSQL¥9.0 の下の pg_env.bat を
  実行する。
- 【Linuxでは】
  /opt/PostgreSQL/9.0 の下にインストールされる。
  データベースは /opt/PostgreSQL/9.0/data の下に作られる。
  環境変数PATHに /opt/PostgreSQL/9.0/bin を追加するか、
  あるいは /opt/PostgreSQL/9.0 の下の pg_env.sh を読み込む。
  (". pg_env.sh" を実行する)

Linux (RedHat系) へのインストール

OSS-DB

- CentOSやFedoraでは、yumコマンドでインストールするのが基本だが、
  # yum install postgresql-server
  とすると、PostgreSQL 8.4がインストールされるので注意。
- PostgreSQL 9.0をyumコマンドでインストールする前に、
  パッケージのダウンロードが必要。
  http://yum.pgrpms.org/howtoyum.php （別紙3）
  にパッケージとインストールガイド (英語) がある。
- CentOSの場合、上記ページの“Please click here and download…”の
  “here”をクリック。
  http://yum.pgrpms.org/reporpms/repoview/letter_p.group.html
  “pgdg-centos”をクリック。（別紙4）
  http://yum.pgrpms.org/reporpms/repoview/pgdg-centos.html
  ”pgdg-centos-9.0-2.noarch.rpm”をクリック (別紙5) して、
  ダウンロードしたrpmを
  # rpm -ivh pgdg-centos-9.0-2.noarch.rpm
  としてインストールする。

Linux (RedHat系) へのインストール

OSS-DB

- http://yum.pgrpms.org/howtoyum.php （別紙3）
  の中ほどにあるImportant noteの指示に従い、/etc/yum.repos.dの下の
  CentOS-Base.repoを編集する。
  [base]と[updates]に
  exclude=postgresql*
  を追加する。

- 最後に
  # yum install postgresql-server
  とすればインストールされる。

- Fedoraの場合は、ダウンロードするファイル、編集する repo ファイルの
  ファイル名が違う他、yumコマンドでpostgresql-serverではなく、
  postgresql90-serverを指定する、つまり
  # yum install postgresql90-server
  としてインストールする。

Linux (RedHat系) へのインストール後の初期設定

OSS-DB

- postgres ユーザは自動的に作成される。

- プログラムは /usr/pgsql-9.0 の下にインストールされる。
  データベースは /var/lib/pgsql/9.0/data の下に作成される。

- 主なコマンドは /usr/bin の下にシンボリックリンクが作られるが、
  pg_ctl や initdb など一部のコマンドについてはリンクが作成されないので、
  PATHを設定するか、絶対パスで起動する必要がある。

- インストールしただけでは、データベースの初期化、起動、自動起動の
  設定などはされない。
  - # service postgresql-9.0 initdb （データベース初期化）
  - # service postgresql-9.0 start （データベース起動）
  - # chkconfig postgresql-9.0 on （データベース自動起動の設定）

Linux(Ubuntu)へのインストール

OSS-DB

- $ sudo apt-get install postgresql
  とすると、やはりPostgreSQL 8.4がインストールされてしまう。
- PPA (Personal Package Archives) を利用すれば、以下の手順で
  インストール可能。
  $ sudo add-apt-repository ppa:pitti/postgresql
  $ sudo apt-get update
  $ sudo apt-get install postgresql
- postgres ユーザが自動的に作られる。データベースも作成され、
  自動起動の設定もされる。
- プログラムは /usr/lib/postgresql/9.0 の下、
  データベースは /var/lib/postgresql/9.0/main の下に作られる。
- 主なコマンドは /usr/bin の下にシンボリックリンクが作られるが、
  pg_ctl や initdb など一部のコマンドについてはリンクが作成されないので、
  PATHを設定するか、絶対パスで起動する必要がある。
- インストール後の環境がちょっと特殊なので、学習環境としては推奨しない。

Linuxへのインストール（OpenSCGのパッケージ） OSS-DB

- http://www.openscg.org/se/postgresql/packages.jsp （別紙6）
  に、RedHat系、Debian系、それぞれのバイナリパッケージが用意されているので、ダウンロードして、rpmコマンド、dpkgコマンドを使ってインストールすることが可能。
  インストール方法、インストール後のセットアップなどの詳細は、
  上記のページの"Installing RPM's"および"Installing DEB's"のリンクに記述されている。（別紙7、別紙8）
- RedHat系は"rpm -ihv filenamem"を、Debian系は"dpkg -i filename"をrootで実行すると、/opt/postgres/9.0 の下にプログラムがインストールされる。
- # /etc/init.d/postgres-9.0-openscg start
  を実行すると、postgres ユーザの作成、データベースの初期化、
  自動起動の設定などが行われる。
- PATHに /opt/postgres/9.0/bin を追加するか、
  /opt/postgres/9.0 の下の pg90-openscg.env を読み込む。
  (". pg90-openscg.env"を実行)

ソースコードからのインストール

OSS-DB

- Linuxでは、コンパイラなどの開発環境が標準で用意されており（インストールされていなくても簡単にセットアップ可能）、ソースコードから自分でビルドしてインストールするのも難しくない。
- ソースコードはPostgreSQLの公式サイト
  http://www.postgresql.org/ftp/source/
  からダウンロードできる。
- ビルド、およびインストールの手順は、オンラインマニュアル
  http://www.postgresql.jp/document/9.0/html/
  の15章（Linux）、16章（Windows）に解説されている。
- 基本的には、
  ```
  $ ./configure
  $ make
  # make install
  ```
  を実行するだけ。
- 多くの環境では configure の実行でいくつかエラーが出るが、
  これを自力で解決できる人には、ソースからのインストールを勧める。

ソースコードからインストールした後の初期設定

OSS-DB

- make install は、プログラムを /usr/local/pgsql の下にインストールするだけなので、その後の初期設定をすべて実行する必要がある。

- postgres ユーザの作成

```
# useradd postgres
```

- 環境変数の設定 (~postgres/.bash_profile、およびPostgreSQLを利用するユーザの~/.bash_profile に追記)

```
export PATH=$PATH:/usr/local/pgsql/bin
export PGDATA=/usr/local/pgsql/data
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/pgsql/lib
export MANPATH=$MANPATH:/usr/local/pgsql/share/man
```

- データベース用ディレクトリの作成

```
# mkdir /usr/local/pgsql/data
# chown postgres /usr/local/pgsql/data
# chmod 700 /usr/local/pgsql/data
```

ソースコードからインストールした後の初期設定

OSS-DB

- データベースの初期化と起動 (postgresユーザで実行)

```
$ initdb -E UTF8 --no-locale
$ pg_ctl start
```

- 自動起動の設定 (RedHat系)

contrib/start-scripts/linux を /etc/rc.d/init.d/postgresql-9.0 にコピー

```
# chkconfig --add postgresql-9.0
# chkconfig postgresql-9.0 on
```

- 自動起動の設定 (Debian系)

contrib/start-scripts/linux を /etc/init.d/postgresql-9.0 にコピー

```
$ sudo update-rc.d postgresql-9.0 defaults 98 02
```

SQLによるデータ操作

SQLによるデータ操作

OSS-DB

SQLとは？

- Structured Query Language
- RDBMSにアクセス（データの検索および更新）するときに使われる言語
- ANSI/ISOにより標準化されており、どのRDBMSでも共通に利用できるが、製品により多少の違い（方言と呼ばれることが多い）がある

RDBMS、SQLで重要な概念

- 表（table）
- 列（column, field）
- 行（row, record）

表、列、行の概念

OSS-DB

CANDIDATE(受験者表) ←表名

CID (受験者番号)	NAME (氏名)
C001	小沢次郎
C002	石原伸子
C003	戌井玄太郎
C004	山本花子

←列名

←行

↑
列

表と列はデータを入れるための器、行がデータ。

**表や列の名前に日本語 (漢字) を使用しても問題なく動作することが多いが、
一般的には望ましくないので、表名、列名には英数字のみを使うことを推奨する。**

表、列、行の概念

OSS-DB

EXAM(試験結果表)

EID (試験ID)	CID (受験者番号)	EXAM_NAME (試験名)	EXAM_DATE (受験日)	SCORE (得点)	GRADE (試験結果)
1	C001	Silver	2011/7/1	80	Pass
2	C002	Silver	2011/7/1	75	Pass
3	C003	Silver	2011/7/2	50	Fail
4	C001	Gold	2011/7/4	40	Fail
5	C004	Silver	2011/7/10	90	Pass
6	C002	Gold	2011/7/12	85	Pass
7	C003	Silver	2011/7/12	30	Fail
8	C001	Gold	2011/7/14	70	Pass

EXAM表のCIDはCANDIDATE表のCIDを参照している。

データベースへの接続（psqlコマンド）

OSS-DB

- SQLを使ってデータの操作（検索、更新）を行うには、psqlコマンドを使う。

- 簡単な使い方：
  $ psql［-U ユーザ名］［-d 接続先DB名］

- 初期設定の直後は、postgres ユーザと、postgres という名前のDBができており、
  $ psql -U postgres -d postgres
  とすることで接続できる。

- （必要もないのに）管理者ユーザでDBの操作をするのは望ましくないので、可能であれば、createuserコマンドで一般ユーザを作成、createdbコマンドで一般ユーザ用のDBを作成しておく。
  （これらのコマンドは後述）

表の作成(CREATE TABLE文)

OSS-DB

基本的な使い方

- CREATE TABLE 表名 (列名 データ型, 列名 データ型...);

このセミナーで使用するテーブルの作成

- CREATE TABLE candidate
  (cid VARCHAR(10), name VARCHAR(30));
- CREATE TABLE exam
  (eid INTEGER,
    cid VARCHAR(10),
    exam_name VARCHAR(10),
    exam_date DATE,
    score INTEGER,
    grade VARCHAR(10));

補足

- VARCHARは可変長文字列型、INTEGERは整数型、DATEは日付型
- SQL文は";"(セミコロン)で終了する、改行では終わらない
- SQL文では、クォーテーションで囲まれた文字列を除き、
  大文字・小文字は区別されない

データの検索(SELECT文)

OSS-DB

基本的な使い方

- SELECT 列名, 列名... FROM 表名 WHERE 検索条件;

表示する列の選択

- すべての列を表示
  SELECT * FROM candidate;
- 指定した列だけを表示
  SELECT eid, exam_date, grade FROM exam;

指定した条件に合致したデータの検索

- SELECT * FROM exam WHERE grade = 'Pass';

データの集計

- データの件数
  SELECT COUNT(*) FROM exam
  WHERE score >= 50 AND exam_name = 'Silver';
- データの平均値
  SELECT exam_name, AVG(score) FROM exam GROUP BY exam_name;

データの検索 (SELECT文)

OSS-DB

テーブルの結合

- SELECT name, exam_name, exam_date, grade
  FROM exam e
  JOIN candidate c ON c.cid = e.cid; または
- SELECT name, exam_name, exam_date, grade
  FROM exam e, candidate c
  WHERE c.cid = e.cid;

順序付けて出力

- SELECT name, exam_name, exam_date, grade
  FROM exam e
  JOIN candidate c ON c.cid = e.cid
  ORDER BY c.cid;
- 表は行の集合であって、行と行の間に順序関係はない。
  ORDER BY がないときの出力順は不定であることに注意

データの追加(INSERT文)

OSS-DB

基本的な使い方

- INSERT INTO 表名(列名, 列名...) VALUES (値, 値...);

このセミナーで使用するサンプルデータ

- INSERT INTO candidate(cid, name) VALUES
  ('C001', '小沢次郎'), ('C002', '石原伸子'), ('C003', '戌井玄太郎'),
  ('C004', '山本花子');
- INSERT INTO exam(eid, cid, exam_name, exam_date, score, grade)
  VALUES
  (1, 'C001', 'Silver', '2011-07-01', 80, 'Pass'),
  (2, 'C002', 'Silver', '2011-07-01', 75, 'Pass'),
  (3, 'C003', 'Silver', '2011-07-02', 50, 'Fail'),
  (4, 'C001', 'Gold', '2011-07-04', 40, 'Fail'),
  (5, 'C004', 'Silver', '2011-07-10', 90, 'Pass'),
  (6, 'C002', 'Gold', '2011-07-12', 85, 'Pass'),
  (7, 'C003', 'Silver', '2011-07-12', 30, 'Fail'),
  (8, 'C001', 'Gold', '2011-07-14', 70, 'Pass');

データの更新(UPDATE文)

OSS-DB

基本的な使い方

- UPDATE 表名 SET 列名 = 値 WHERE 検索条件;

使い方の例

- UPDATE candidate SET name = '小沢三郎' WHERE cid = 'C001';
- UPDATE candidate SET name = '鈴木花子' WHERE name = '山本花子';
- UPDATE exam SET score = score*2 WHERE exam_name = 'Silver';
- UPDATE exam SET grade = 'Fail' WHERE score <= 70;

注意事項

- WHERE句をつけないと、全件更新されてしまうので注意

データの削除(DELETE文)

OSS-DB

基本的な使い方

- DELETE FROM 表名 WHERE 検索条件;

使い方の例

- DELETE FROM candidate;（全件削除！！）
- DELETE FROM exam WHERE exam_date = '2011-07-14';

注意事項

- WHERE句を忘れると全件削除されてしまうので注意
- INSERT/UPDATE/DELETEがデフォルトでは即時実行される
  (COMMIT不要、というよりROLLBACKできない)ことに注意
  (特に、Oracle/DB2でSQLを学んだ人)

psql

OSS-DB

- psqlは、SQL文の実行以外にも様々な機能がある。
- $ psql –l （データベース一覧の表示）
- 主なpsqlコマンド（'=>' はpsqlのプロンプト）
  - => ¥d （テーブル一覧の表示）
  - => ¥d 表名 （指定した表の列名、データ型の表示）
  - => ¥? （psql で使える各種コマンドに関するヘルプの表示）
  - => ¥h （SQL に関するヘルプの表示）
    - => ¥h SELECT （SELECTの使い方に関するヘルプの表示）
  - => ¥! OSコマンド （OSコマンドの実行）
    - => ¥! ls （カレントディレクトリのファイル一覧の表示）
  - => ¥q （終了）
- '¥'で始まるのはpsqlの独自コマンドで、psqlツールによって処理される。
  それ以外のものはSQL文と判断され、データベースに送信される。

データベース
運用管理の基礎

データベース運用管理の基礎

OSS-DB

運用管理の目的

- 必要な人に、適切なDBサービスを提供すること（セキュリティ管理）
  - 必要ない人にはサービスを提供しない
  - 不正なアクセスを拒絶する
  - 設定と監視
- サービスレベルの維持
  - 定められた水準のサービスを提供し続けること
- トラブルシューティング（予防と対処）
  - DBに接続できない
  - DBが遅い
  - DBが起動しない
  - ディスク、ファイル、データの破損
  - バックアップ、リストア、リカバリ

GUIツールとコマンドライン操作

- 便利なGUIツールもあるが、コマンドラインで操作するスキルがあることが基本

データベース運用管理の基礎

OSS-DB

- データベースの初期化
- 設定ファイル
- ユーザ管理
- データベースのバックアップ

データベースの初期化

OSS-DB

データベースの初期化（データベースクラスタの作成）

- PostgreSQLが管理するデータを記憶する領域をデータベースクラスタと呼ぶ。（他のDBでは違う意味で使われるので注意）
- データベースクラスタ内には以下のものなどが含まれる。
  - （複数の）データベース
  - ユーザ情報などのグローバルデータ
  - 設定ファイル
- インストール方法によっては、自動的にデータベースクラスタが作成される。
- 手動でデータベースクラスタを作成するにはデータベース管理者ユーザ（通常は postgres ユーザ）で initdb コマンドを実行する。例えば
  - $ initdb -E UTF8 --no-locale -D /usr/local/pgsql/data
- 実行直後には、template0, template1, postgres の3つのデータベースが作られる。
- PostgreSQLサーバを起動する前に、データベースクラスタを作成する必要がある。

データベースの初期化

OSS-DB

■データベースサーバの起動

- データベースサーバが起動していなければ、psql コマンドなどでデータベースにアクセスすることができない。
- 多くの場合、データベースサーバは自動的に起動されるように設定されているが、手作業で起動するときは、データベース管理者ユーザで、pg_ctl コマンドを使う。
  - $ pg_ctl start -D /usr/local/pgsql/data (PostgreSQLサーバの起動)
  - $ pg_ctl stop -D /usr/local/pgsql/data (PostgreSQLサーバの停止)

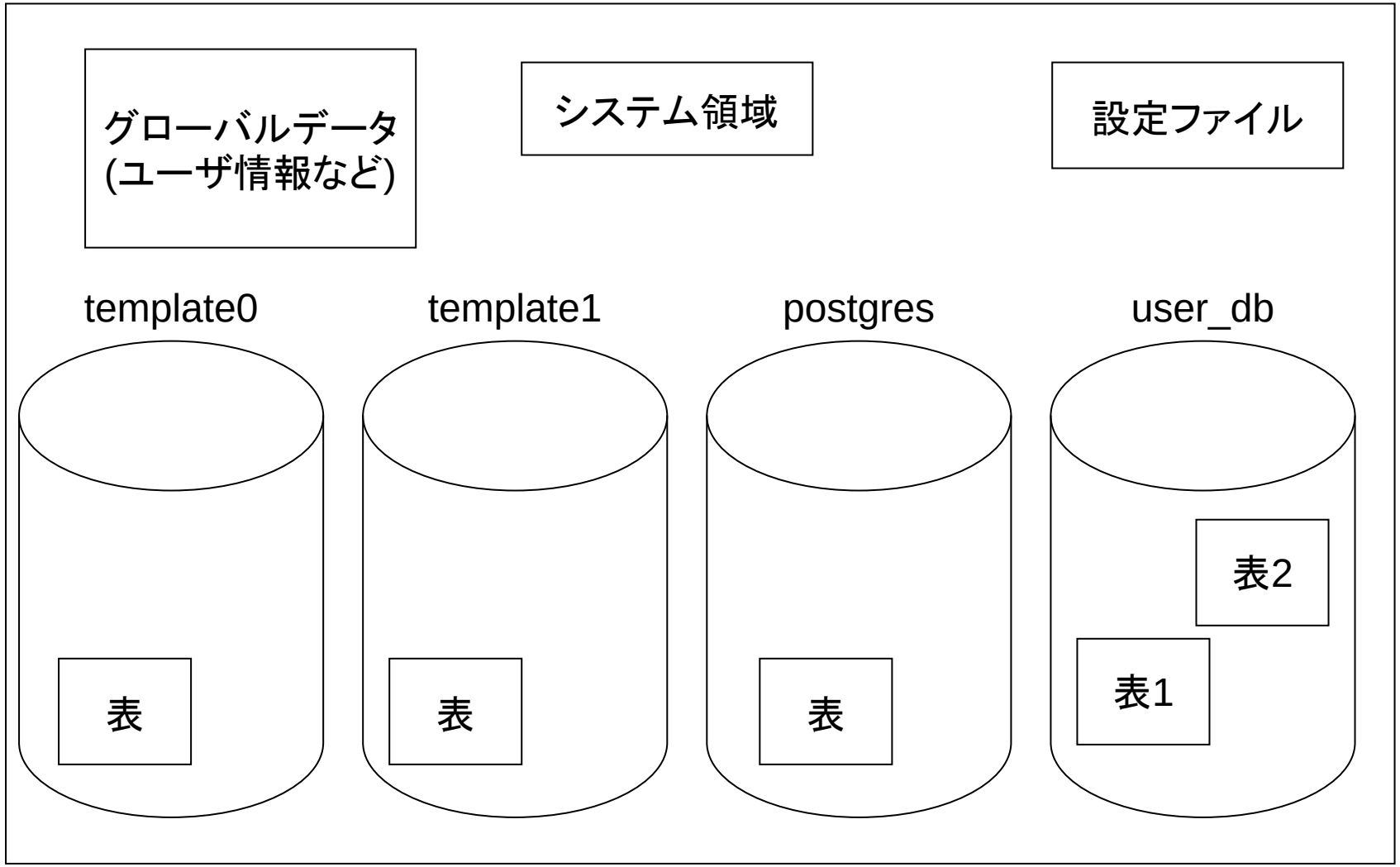
■データベースの作成

- initdb で作成される3つのデータベース(template0, template1, postgres)はいずれもユーザ用ではない。
  各種データの格納のためには、ユーザ用のデータベースを作成する。
- postgres ユーザで createdb コマンドを実行するか、psql でデータベースに接続して CREATE DATABASE 文を使う。
  - $ createdb –O ownername dbname
  - =# CREATE DATABASE dbname OWNER ownername;

データベースクラスタのイメージ

OSS-DB

データベースクラスタ



設定ファイル (postgresql.conf)

OSS-DB

- DBサーバーのリソースなど、各種パラメータの設定をするファイル
  - '#'で始まる行はコメント
  - "パラメータ名 = 値" という形式でパラメータを設定
  - 主なパラメータと設定の例
    - listen_address = '*' (TCP接続を許可する)
    - log_destination = 'syslog' (サーバーのログをsyslogに出力する)
    - log_line_prefix = '%t %p' (ログ出力時に、時刻とプロセスIDを付加
  - この他、パフォーマンスチューニングなどのための多数のパラメータが設定できるが、OSS-DB Silverの試験で問われるのは、以下の4つ (数字はマニュアルの節番号)
    - 記述方法 (18.1)
    - 接続と認証 (18.3)
    - クライアント接続デフォルト (18.10)
    - エラー報告とログ取得 (18.7)

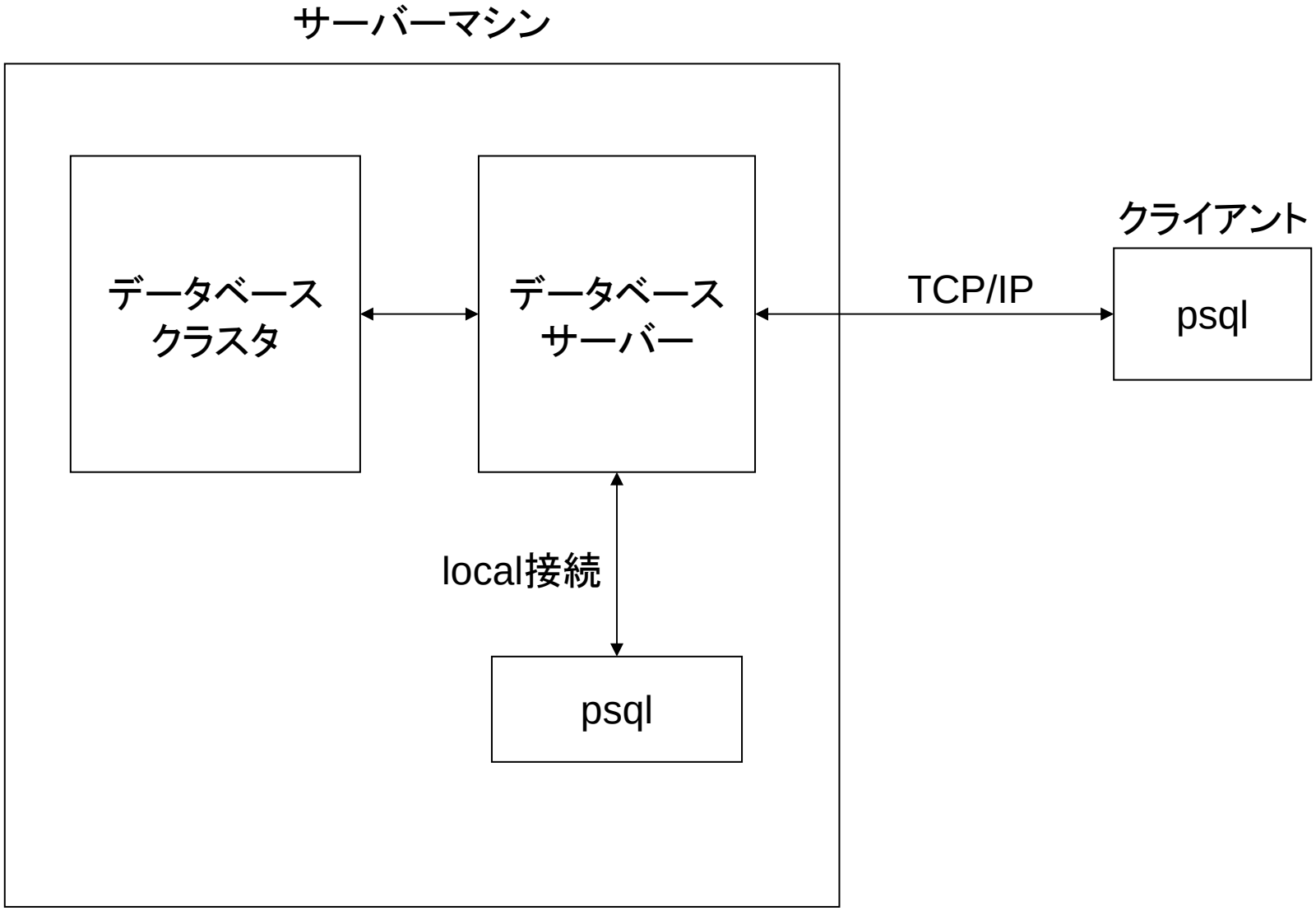
設定ファイル(pg_hba.conf)

OSS-DB

- HBA=Host Based Authentication
- DBへの接続を許可(あるいは拒否)する接続元、データベース、ユーザの組み合わせを設定
  - 先頭行から順に調べて、マッチする組み合わせが見つかったところで終了
  - マッチする組み合わせが見つからなければ、接続拒否
- 記述形式
  - local database名 ユーザ名 認証方法
  - host database名 ユーザ名 接続元IPアドレス 認証方法
- 記述例
  - local all postgres md5 (postgresユーザでの接続はパスワードを要求)
  - local all all ident (OSのユーザ名とDBのユーザ名が一致すれば接続可)
  - host all all 127.0.0.1/32 trust (ローカルホストからは接続可)
  - host db1 all 192.168.0.0/24 reject
    (192.168.0.1-255からdb1には接続不可)
  - host all all 192.168.0.0/24 trust
    (192.168.0.1-255から接続可)

データベースサーバーとクライアント

OSS-DB



ユーザ管理

OSS-DB

一般ユーザと管理者ユーザ (スーパーユーザ)

- OSに一般ユーザと管理者ユーザがあるのと同じように、データベースにも一般ユーザと管理者ユーザがある。
- 一般ユーザには限られた権限しかないが、管理者ユーザにはすべての権限がある。
- OSの管理者ユーザと、データベースの管理者ユーザは異なる。
  例えば、root で pg_ctl コマンドを実行することはできない。

権限とは？

- 多くの種類の権限があるが、例えば
  - 新規にテーブルを作成する権限、あるいは削除する権限
  - テーブルからデータを検索 (SELECT) する権限
  - テーブルのデータを更新 (UPDATE) する権限
- デフォルトでは、テーブルの所有者 (作成者) だけが、そのテーブルに対するSELECT/UPDATEなどの権限を持つ (管理者ユーザは別)。
  つまり、権限を与えられなければ、他人のDBやテーブルを参照/更新できない。

ユーザ管理

OSS-DB

ユーザ作成

- postgres ユーザで createuser コマンドを使う。
  - $ createuser [option] [username]
- オプションで指定しなかった場合、以下を対話的に入力する。
  - 新規ユーザ名
  - 新規ユーザを管理者ユーザとするかどうか
  - 新規ユーザにデータベース作成の権限を与えるかどうか
  - 新規ユーザにユーザ作成の権限を与えるかどうか

権限管理

- テーブルなどのオブジェクトに対する権限の付与・剥奪には、SQLのGRANTとREVOKEを使う。
- 個々のユーザに対して、GRANT/REVOKEすることもできるが、ユーザ名としてpublicを指定すれば、全ユーザに対してGRANT/REVOKEも可能。
  - => GRANT SELECT ON candidate TO public;
  - => GRANT SELECT, UPDATE ON exam TO foobar;
  - => REVOKE DELETE ON exam FROM public;

データベースのバックアップ

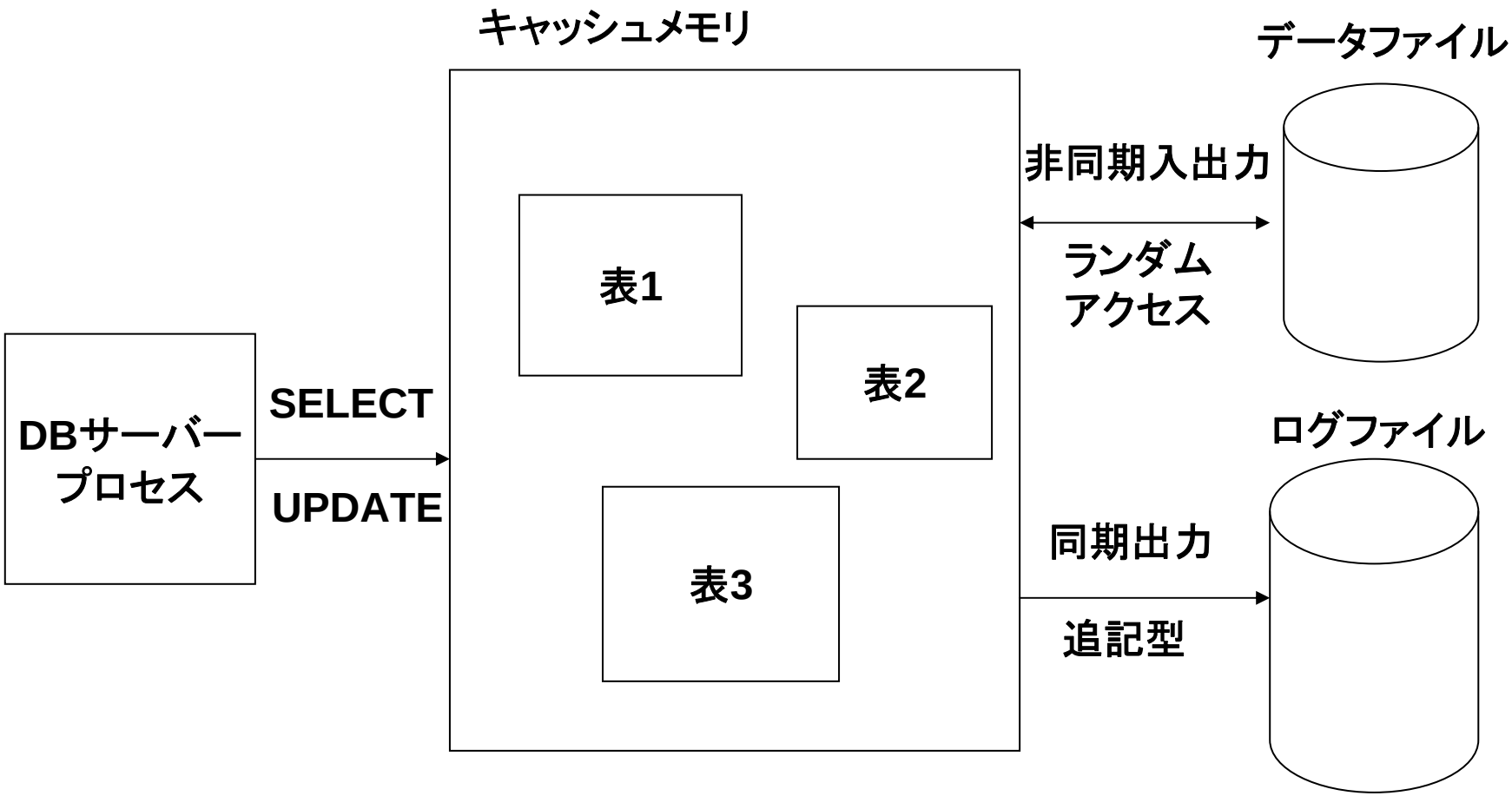
OSS-DB

- データベースでは重要なデータを管理している。ディスクの故障などによるデータの損失に備え、バックアップを取得することが重要。

- データベースのファイルは常に更新され続けている。メモリ上のデータ（キャッシュ）とディスク上のデータファイルの内容が一致するとは限らない。つまり、OSコマンドを使ってファイルをコピーしてもバックアップにはならない。
- データベースのバックアップには特殊な方法が必要。

- データベースがクラッシュしたとき、一週間前のバックアップからデータベースが復元（リストア）できても、ありがたくないかもしれない。
- クラッシュ直前の状態にデータを復旧（リカバリ）する手段がある。

キャッシュメモリとファイルの関係



データベースのバックアップ

OSS-DB

簡単なバックアップ方法

- pg_dump コマンドでデータベース単位のバックアップを取得。
  - $ pg_dump [options] –f dumpfilename dbname　あるいは
  - $ pg_dump [options] dbname > dumpfilename
- オプションで、出力形式（テキスト/バイナリ）などを指定できる。
- データベースクラスタ内のすべてのデータベースのバックアップを取得するには、pg_dumpall コマンドを使う。（出力形式はテキストのみ）

バックアップからのリストア方法

- テキスト形式のバックアップは psql コマンドで、バイナリ形式のバックアップは pg_restore コマンドでリストアする。
  - $ psql –f dumpfilename dbname　あるいは
  - $ psql dbname < dumpfilename
  - $ pg_restore –d dbname dumpfilename

データベースのバックアップ

OSS-DB

ディレクトリコピーによるバックアップ

- データベースを停止すれば、物理的なデータファイルをディレクトリごとコピーすることでバックアップを作成できる。(コールドバックアップ)
- コピーの方法は自由に選んで良い。(tar, cpio, zip...)
- バックアップを、同じ構成の別のマシンにコピーして動かすこともできる。

PITR (Point In Time Recovery)

- 障害の直前の状態までデータを復旧(リカバリ)できる。
- 間違ってデータを削除した場合でも、任意の時点まで戻すことができる。

PITRの仕掛け

- WAL(Write Ahead Logging)により、データファイルへの書き込み前に、変更操作についてログ出力される。(トランザクションログ)
- 最後のバックアップ以後、障害発生直前までのWALを、バックアップに適用することで、データを復旧できる。

CSVファイルの入出力

OSS-DB

- psql の ¥copy コマンドを使うと、データベースのテーブルと、OSファイルシステム上のファイル (CSVなど) の間で入出力ができる。

- 基本的な使い方
  - => ¥copy table_name to file_name [options]
  - => ¥copy table_name from file_name [options]
  - デフォルトではタブ区切りのテキストファイル、オプションに"csv"と指定すれば、カンマ区切りのCSVファイルになる。

- SQLのCOPYコマンド (PostgreSQLの独自拡張機能) もあるが、¥copy との使い方の違いに注意。
  - =# COPY table_name TO 'file_name' [options];
  - =# COPY table_name FROM 'file_name' [options];
  - ¥copyはクライアント上のファイル、COPYはサーバ上のファイルの入出力。
  - COPYによるファイル入出力は、データベース管理者ユーザのみ実行できる。

例題解説

例題解説1

OSS-DB

■一般知識 – コミュニティと情報収集

PostgreSQLの開発元が運用しているメーリングリストについて、
正しいものを2つ選びなさい。

A. 目的別に複数のメーリングリストが運用されている。
B. 日本語、英語、フランス語など参加者の母国語で情報交換が可能である。
C. ソースコードの改変などに参加している開発者だけが登録可能である。
D. 開発者でなくても参加できるが、開発者からの招待がなければ登録できない。
E. 過去のメールのやり取りがすべて公開されている。

例題解説2

OSS-DB

■運用管理 – バックアップ方法

以下のコマンドについての記述から、正しいものを2つ選びなさい。

```
$ pg_dump -F c -U a p > q
```

A. テキスト形式のバックアップを出力している
B. バックアップ中のエラーメッセージがファイル q に出力される
C. データベース p のバックアップを取得している
D. ユーザ a でデータベースに接続する
E. このコマンドで作成されたバックアップをリストアするには psql コマンドを使う

例題解説3

OSS-DB

■開発 – SELECT文、JOIN

以下のような行をもつテーブルt1, t2がある。いずれもid列はINTEGER型、name列はVARCHAR(10)型である。

```
 t1                t2
 id | name         id | name
----+-------     ----+-------
  1 | abcde         1 | abcde
  2 | fghij         1 | xyzvw
  3 | klmno         2 | uvwxy
  4 | pqrst         3 | abcde
  5 | uvwxy         6 | zzzzz
```

このとき、次のSQL文を実行した結果の行数は何行か。

```
SELECT * FROM t1 FULL JOIN t2 ON t1.id = t2.id;
```

試験対策

OSS-DB

- 出題範囲詳細に載っている項目すべてについて、マニュアルなどで調査、実際に試す、などして理解する

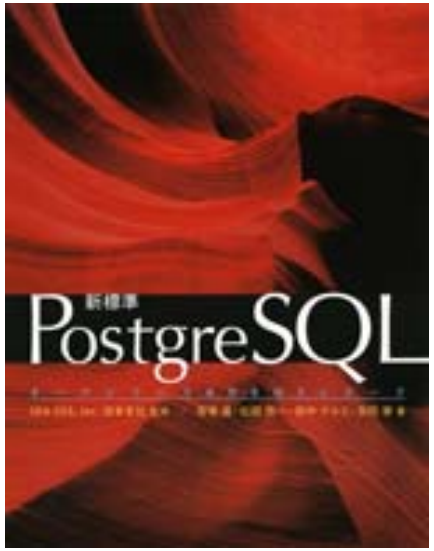
- 以下は出題範囲からの抜粋
  - インストール方法
    - initdbコマンド
    - データベースクラスタの概念
    - テンプレートデータベース
  - 標準付属ツールの使い方
    - pg_ctl, createuser, dropuser, createdb, dropdb, createlang, droplang, psql
  - 設定ファイル
    - postgresql.conf…

- Silverの合格基準は、各機能やコマンドについて
  - その目的を正しく理解していること
  - 利用法を正しく理解していること

参考資料

OSS-DB

- PostgreSQL徹底入門
  - PostgreSQL 9.0対応
  - 9.0.1のインストーラ、ソースコード
- 新標準PostgreSQL
  - PostgreSQL 8.4対応
- SQLポケットリファレンス
  - 他のDBやANSI標準との比較
- オンラインマニュアル
  http://www.postgresql.jp/document/9.0/html/
- 日本PostgreSQLユーザ会
  http://www.postgresql.jp/
- Let's Postgres
  http://lets.postgresql.jp/



ご清聴ありがとうございました。

■お問い合わせ■
LPI-Japan
テクノロジー・マネージャー
松田　神一
matsuda@lpi.or.jp