# PostgreSQL
# 初心者向けセミナー

日本 PostgreSQL ユーザ会 名古屋支部
沢田 潔

# PostgreSQL の特徴

# PostgreSQL とは

- 1996年から正式に PostgreSQLとなる
  - Ingress カリフォルニア大学(UCB)
  - Postgres UCB Ver4.2
  - Postgre95 ～1995年まで
- インターネットを利用したボランティアでの開発体制
  - 中心となる開発者は、全世界に30人程度
- http://www.postgresql.org/ （本家）
- http://www.postgresql.jp/ （日本 PostgreSQL ユーザ会）

# 日本 PostgreSQL ユーザ会


PostgreSQL

# バージョン番号規則

8 . 3 . 3

メジャーバージョン　　マイナーバージョン

- メジャーバージョンは仕様の追加･変更がある
  - 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は主にバグ修正
- 2008年8月現在の最新バージョンは
  - 8.33, 8.2.9 , 8.1.13 , 8.0.17 , 7.4.21

# PostgreSQL の主な特徴

- フリー＆オープンソース
  - BSDライセンス
  - 利用、コピー、配布の自由
  - 義務は著作権表示のみ
    - 広告情報は無し（いわゆる修正BSDライセンス）
  - 免責
- SQL92/99/2003 のサポート
  - SQL文法の充実度が高い

# PostgreSQL の主な特徴

- 豊富なプラットフォームに対応
  - Linuxを含むほとんどのUNIX系システム
  - Windows（インストーラ有り）
  - Mac OS X
- 本格的なDBMS
  - トランザクション、オプティマイザ、マルチバイト
  - サブクエリー、外部キー、手続き言語、トリガー
  - トランザクションログ、テーブルスペース、PITR
  - テーブル継承

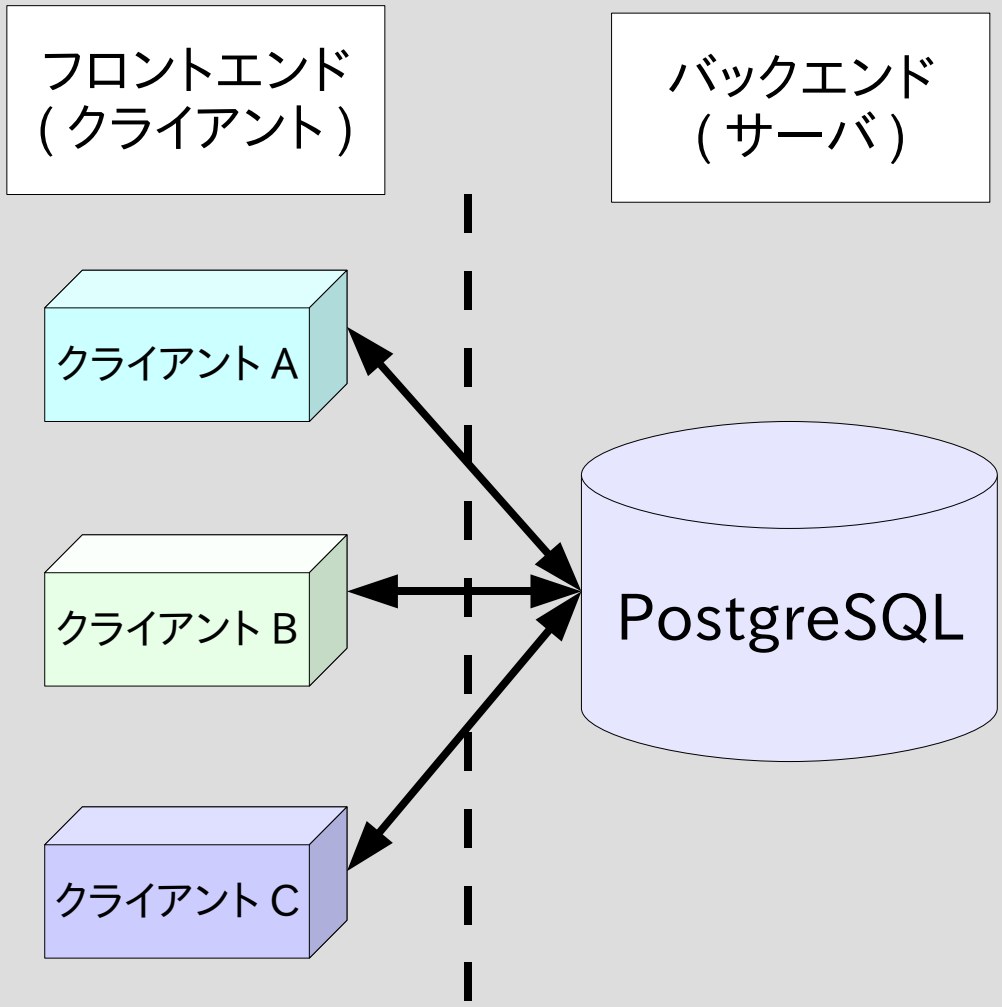
# PostgreSQL の主な特徴

- 商用データベースと比べてコンパクト
- 機能でも勝らずとも劣らず
- 他のOSS-DBと比較しても性能に遜色はない
  - テーブル上のレコード数が数千万件でも問題なし
- 無料で利用できる
- 個人マシンなどに簡単にインストールできる

リレーショナルデータベースの学習に最適!

# クライアント / サーバ 構成

- ネットワーク対応
- 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
- データベースエンジンの変更に影響されにくい



# PostgreSQL に接続可能な言語

| | API |
|---|---|
| C | ○ |
| ecpg (CにSQLを埋めこむプリプロセッサ) | ○ |
| Java | ○ |
| Tcl/TK | ○ |
| Python | ○ |
| C++ | ○ |
| Perl | ○ |
| PHP | ○ |
| Ruby | ○ |
| ODBC | ○ |
| .Net Data Provider | ○ |
| VisualWorks Smalltalk | ○ |

# マルチバイト対応

- 文字エンコーディング
  - データベースごとに指定ができる
  - クライアントごとに通信するエンコーディングを指定することができる
- 自動的に変換することで
  - EUC_JP のデータベースに SJIS クライアントで通信したり
  - UNICODE のデータベースに EUC_JP で通信することができる
- JIS X 0213 (JIS 2004) にも対応

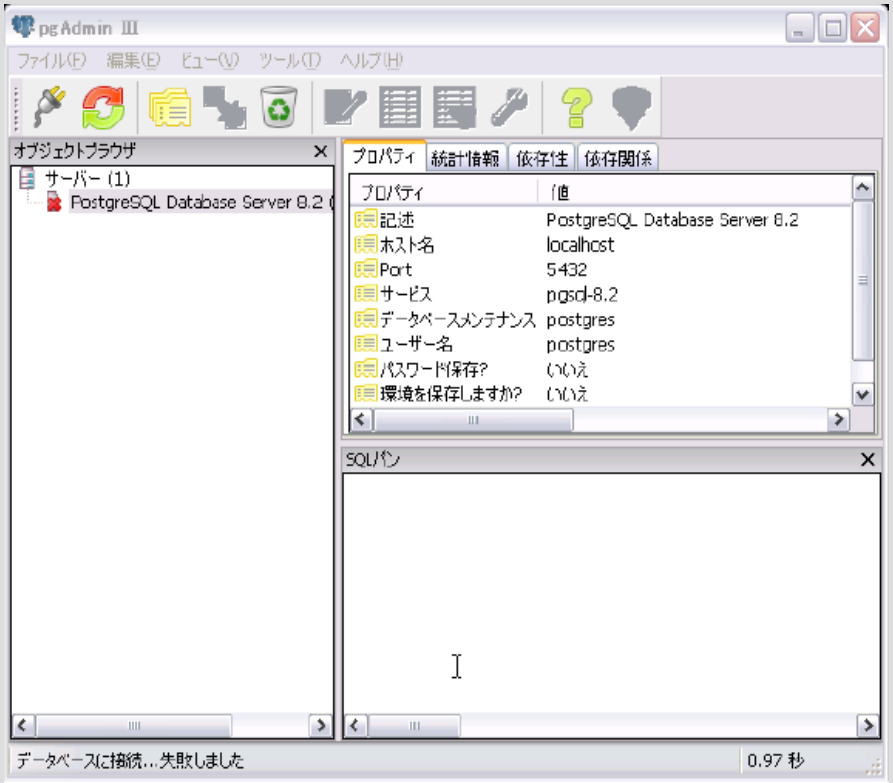
# PostgreSQL を動かす

# PostgreSQL を動かす (Linux)

- ディストリビューションパッケージから
  - 動作が比較的に安定している
  - ただし最新版では無い ( Ver 8.1, 8.2)
    - Debian 系：apt-get
    - RedHat 系： rpm, yum
  - 初期インストールでは ,DB クラスタの DIR が決められている
- ソースコードからコンパイルしてインストール
  - ./configure → make → make install
  - 最近の Linux ディストリビューションであれば特別な手順はなくインストールできる
    - ソースからインストールは、他の RPM パッケージと競合しないよう注意が必要
  - ./configure する際に、パラメータでインストール先などを指定できる
    - ./configure --help でヘルプが表示

# PostgreSQL を動かす (Windows)

- 日本語版インストーラから一発インストール
- サービスの登録、管理ユーザ作成なども自動化
- データベースを操作する GUI ツールも付属


PostgreSQL

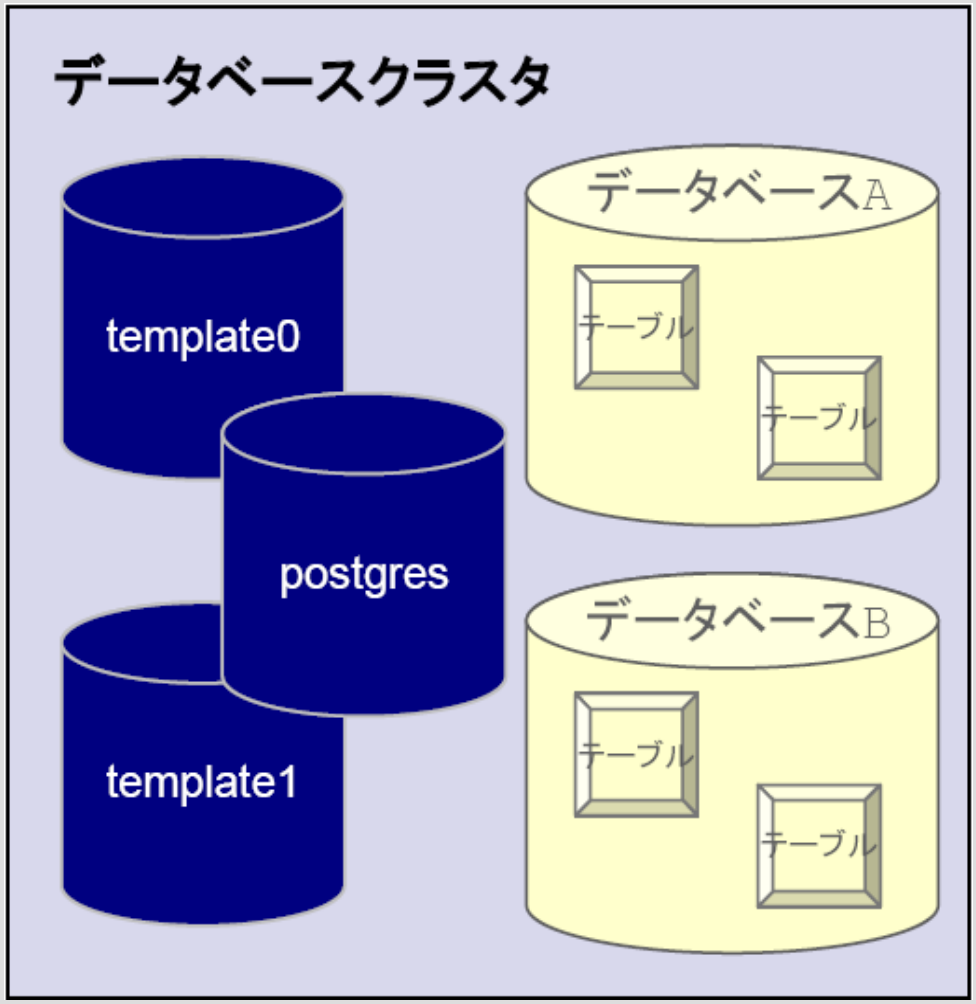
# PostgreSQLを動かす
# インストールされるコマンド

| | | | |
|---|---|---|---|
| createdb | データベース作成 | pg_config | インストール情報 |
| dropdb | データベース抹消 | pg_ctl | PostgreSQLの起動 |
| createuser | ユーザ登録 | pg_dump | データベースのバックアップ |
| dropuser | ユーザ抹消 | pg_dumpall | データベース全体の<br>バックアップ |
| createlang | プログラミング言語追加 | pg_restore | データベース復旧 |
| droplang | プログラミング言語削除 | psql | SQLインタプリタ |
| initdb | データベースクラスタ初期化 | vacuumdb | データベースの<br>ガベージコレクション |

# データベースクラスタ

PostgreSQL データ全てが格納されるディレクトリ

- **initdb** コマンドで最初の1回だけ実行
- システムデータベースがデフォルトで作成される
- 環境変数 $PGDATA で指定
- 教科書的には、
  - /usr/local/pgsql/data



# 起動･停止

## pg_ctl コマンド

root で実行できない

- 起動

```
pg_ctl -D $PGDATA -w start
```

- 再起動

```
pg_ctl -D $PGDATA restart
```

- 停止

```
pg_ctl -D $PGDATA -m f stop
```

-w
wait の略。処理が完了したらプロンプトを戻す。停止の時は、デフォルトで有効になっている。

-m { s | f | i }
mode の略。
smart　全てのクライアントが切断するまで待つ（デフォルト）
fast　クライアントが切断するまで待たない）
immediate
クリーンアップ処理なしで、全サーバプロセスを中断

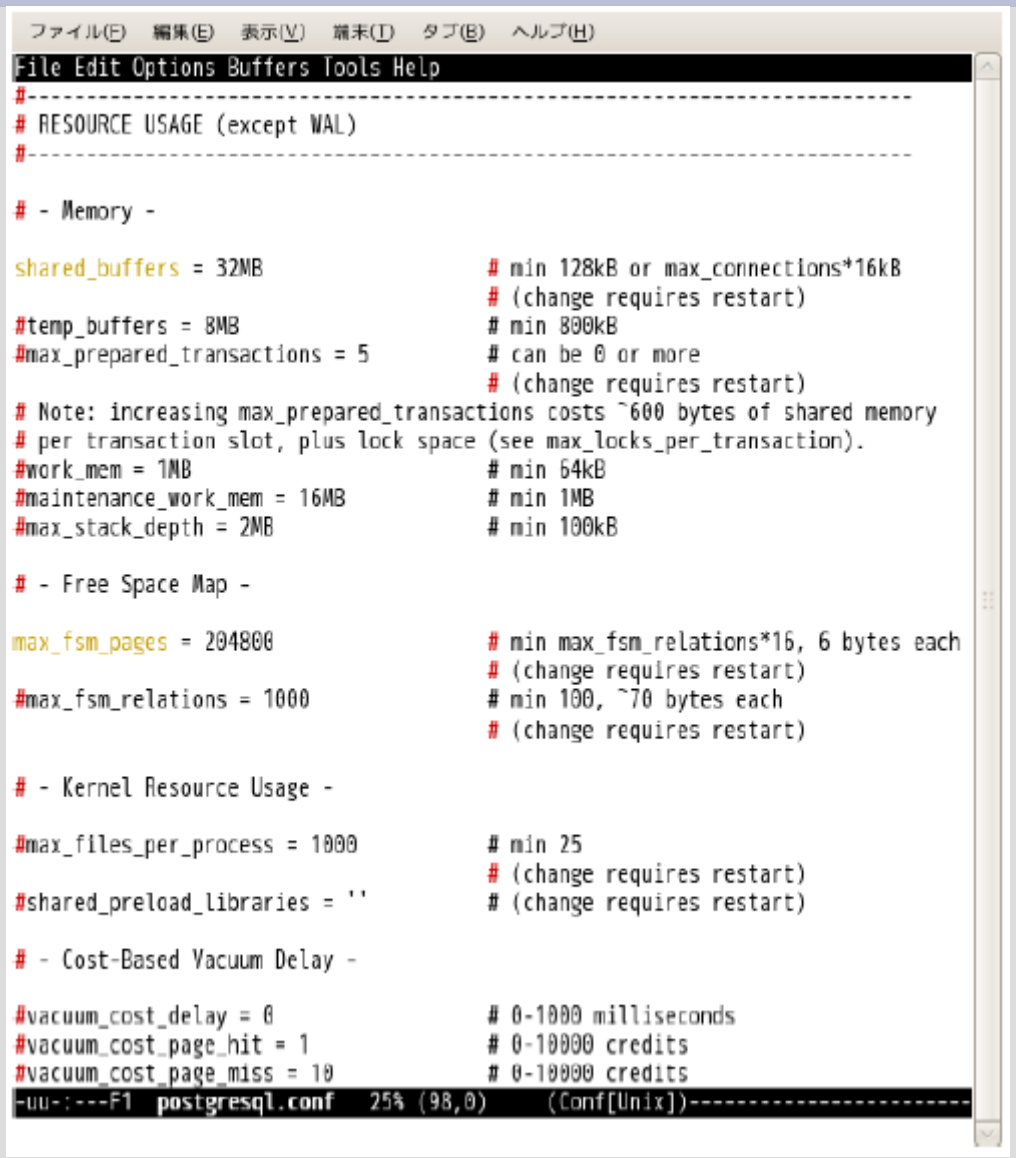
# 設定ファイル

## postgresql.conf

- $PGDATA配下にある
- 書式
  - 「#」で始まる行はコメント
  - パラメータ ＝ 値
- 設定を反映させる為には

```
pg_ctl -D $PGDATA restart
```

```
pg_ctl -D $PGDATA reload
```



# 設定ファイル　主なパラメータ

| | |
|---|---|
| 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。 |
| max_connections | データベースサーバへの同時接続の最大数。 |
| silent_mode | trueの場合、サーバはバックグラウンドで起動し、制御端末は切り離される。 |
| port | 接続ポート番号。デフォルトは5432。 |
| shared_buffers | 共有メモリバッファをページ数で指定。デフォルトは1000。 |
| max_files_per_process | プロセスあたりのファイルオープン最大数。 |
| log_destination | ログの出力先を指定。stderr、syslog、eventlog |
| log_connections | クライアントが接続したことをログに取るようにする。 |
| redirect_stderr | stderrに出力したエラーを $PGDATA/pg_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。 |
| 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。 |

# PostgreSQL の設定

- PostgreSQL の動作、リソースに関しては全て postgresql.conf で設定する
- インストール直後（デフォルトの設定）の postgresql.conf は非力なマシンでも動作するように設定されている
- PostgreSQL の性能を引き出すためには、個々のマシンに沿った設定が必要！
  - いわゆる「チューニング」！
  - JPUG名古屋支部(NPUG)の勉強会に、是非ご期待

# PostgreSQL を使う

# ユーザ作成

## createuser コマンド

- PostgreSQL 8.1 からは「ロール」でユーザとグループを管理する
- SQL の「 CREATE ROLE 」文も利用可能

```
[postgres]$ createuser taro
Shall the new role be a superuser? (y/n) n
Shall the new role be allowed to create databases? (y/n) y
Shall the new role be allowed to create more new roles? (y/n) n
CREATE ROLE
```

# データベース作成

## createdb コマンド

- データベースクラスタ（$PGDATA）には複数のデータベースを作成できる
- データベースごとに所有者を決められる
- SQL の「CREATE DATABASE」文も利用可能

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

# データベース / ユーザ削除

- アンインストールはデータベースクラスタとインストールディレクトリを削除するだけ

```
[postgres]$ rm -rf $PGDATA
[postgres]$ rm -rf /usr/local/pgsql
```

- データベース削除
  - **dropdb** コマンド
  - DROP DATABASE 文 (SQL文)
- ユーザ削除
  - **dropuser** コマンド
  - DROP ROLE 文 (SQL文)

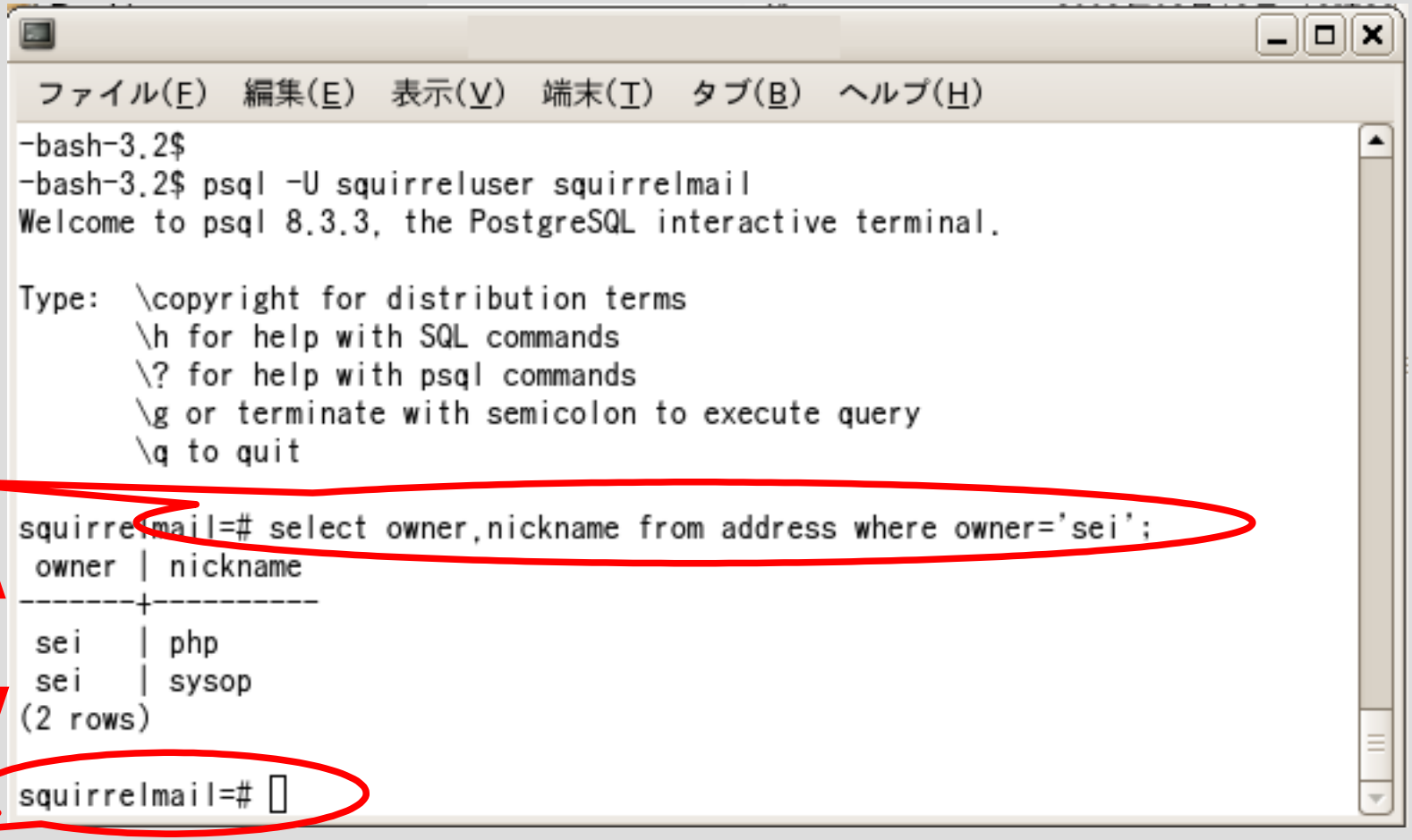
# SQL の実行

## psql コマンド

- 対話型 SQL インタプリタ　（クライアントアプリ）
- ユーザが入力した SQL をデータベースサーバ（バックエンド）に接続して送信する
- SQL の処理結果を受け取りユーザに表示する

```
psql ［オプション］..　［DB名 ［ユーザ名］］
```

# psql コマンド操作例




SQL 入力

バックエンドからの結果表示

接続中のデータベース

# SQL 文

- 標準SQL に準拠しているので、通常の SQL は問題なく使える
  - SELECT, UPDATE, INSERT, DELETE
  - CREATE TABLE, JOIN, OUTER JOIN
  - インデックス、ビュー、外部キー、スキーマ
  - サブクエリー、トリガ、シーケンス、カーソル ...
- 一部方言、独自拡張がある

# データ型

- 1レコード1カラムあたり、1GBまで
- 豊富な組み込みデータ型
- ユーザがデータ型を作れる
  - ユーザ定義データ型
- 主な組み込みデータ型
  - 数値　　日付　　ネットワークアドレス　　配列
  - 通貨　　真偽　　ビット列　　XML
  - 文字　　幾何　　ラージオブジェクト

# ログ

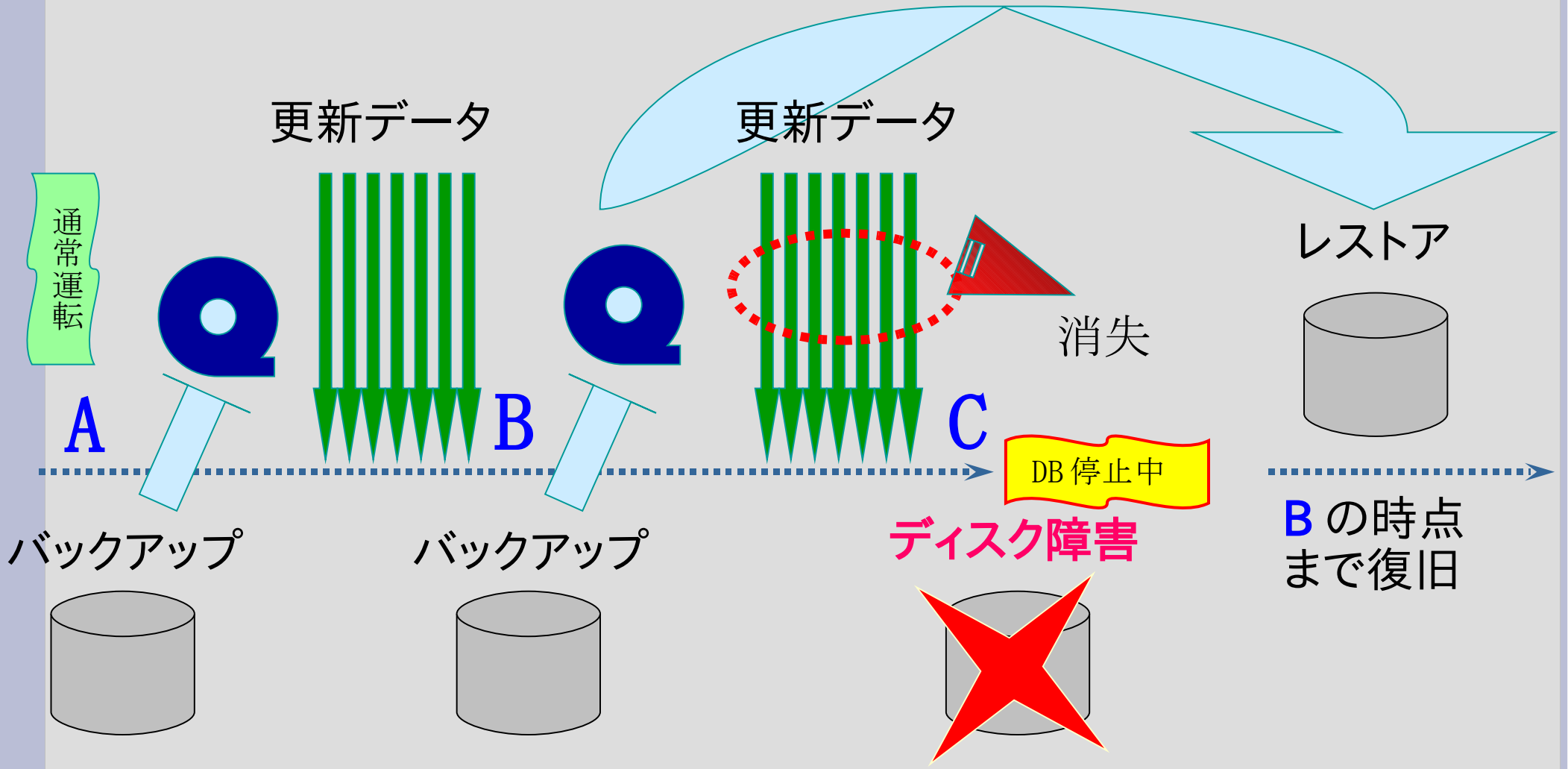
- 独自のログ機能で取得、ローテーションなどを行える
  - syslog にも出力可能
- postgresql.confで設定する
  - ログ出力先
  - ログ出力内容
    - 時刻、PID、SQL文、ユーザ名、DB名など
  - ログのローテェーション サイズとその時間

# PostgreSQL をさらに
# 安心・安全に使うために

# バックアップ / リストア

- pg_dump コマンド
  - データベース単位でバックアップを取得
- pg_dumpall コマンド
  - データベースクラスタ全体のバックアップを取得
- デフォルトは SQL 文で論理バックアップを作成する
- dumpの為にデータベースを停止する必要はない
- リストアは psql を使い、SQL 文から復元する
- ディスク障害などがあり、この Dump ファイルから復旧（リストア）する場合、dump した時点までの分しか復旧できない。

# ディスク障害による更新データの消失

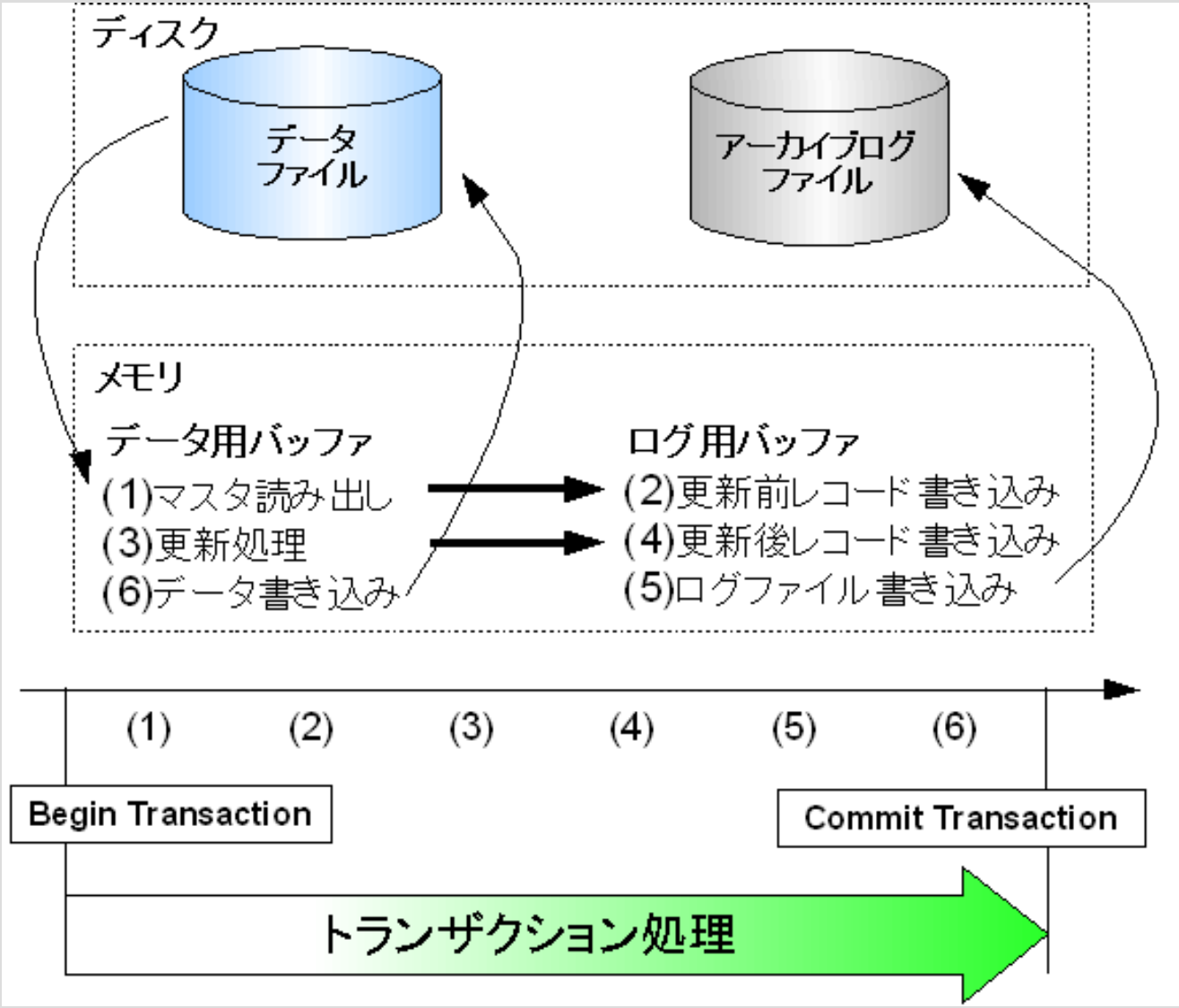



PostgreSQL

# トランザクションについて

トランザクションとは？

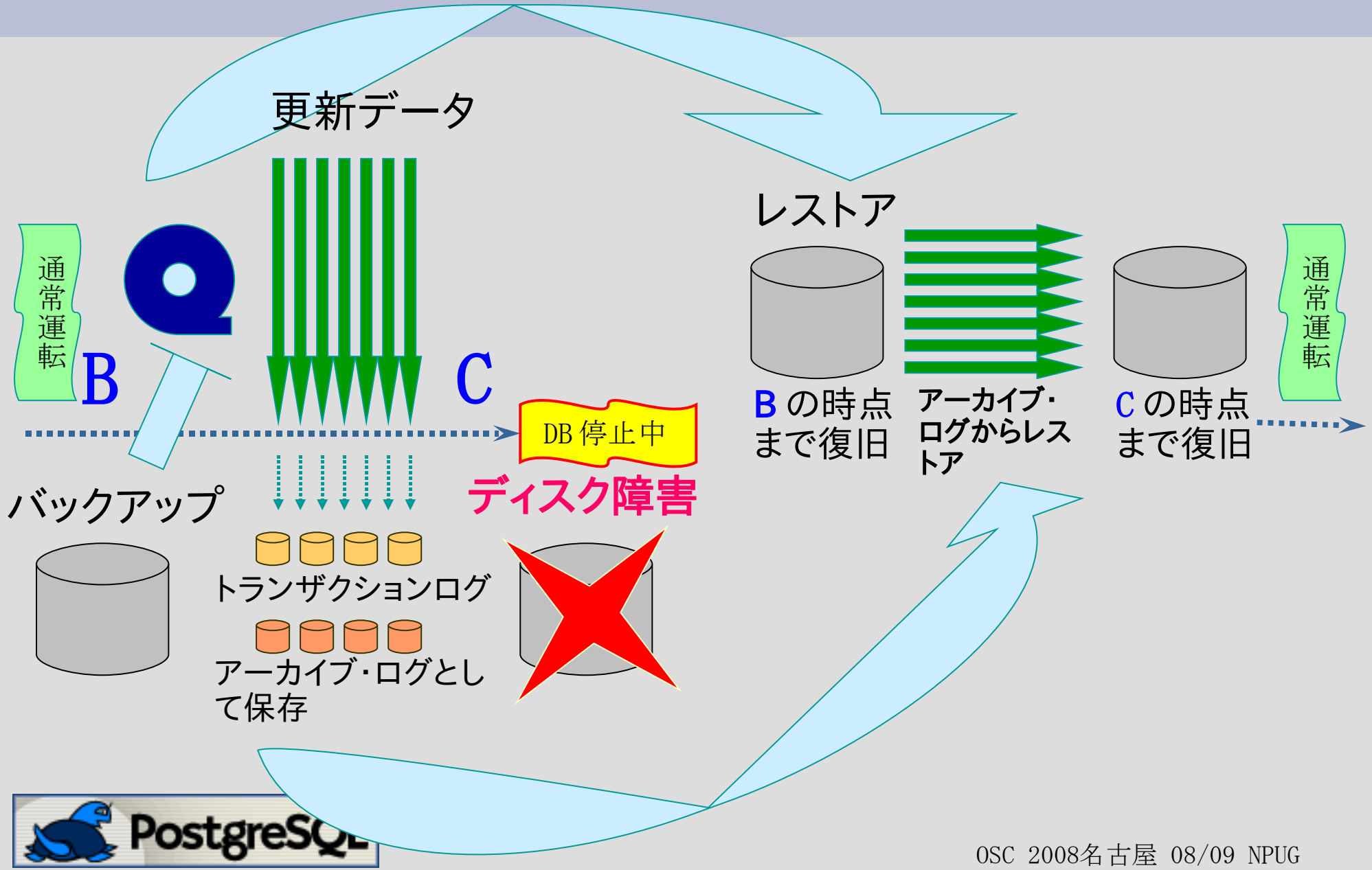
アプリケーションやユーザから見て、「ひと纏まり」のデータベース操作の処理単位



# データリカバリ

- ポイント・イン・タイム・リカバリ (Point In Time Recovery)
  - DB 本体がクラッシュなどで失われても、バックアップとトランザクションログから、最新の状態に復元することが可能
  - CheckPoint のタイミングで捨てていたトランザクションログ (WAL ログ ) を、アーカイブ・ログとして保存しておく
  - DB 本体が、メディア障害によって失われるような重大な障害の際にも、障害直前の状態まで DB の復旧ができる
  - 最新の状態に復元するだけでなく、任意の時点へ戻すことも可能

# PITR リカバリ動作


更新データ
通常運転
B
C
DB 停止中
ディスク障害
バックアップ
トランザクションログ
アーカイブ・ログとして保存
レストア
B の時点まで復旧
アーカイブ・ログからレストア
C の時点まで復旧
通常運転

# セキュリティ

- ホストによる認証
  - pg_hba.conf 設定ファイル
  - パスワード認証、ident認証、pam認証などを、IPアドレス、ユーザ名、データベース名ごとに設定できる
- ユーザの権限
  - テーブル、ビュー、シーケンスといったオブジェクトに、検索、更新、削除などの権限を設定できる
  - GRANT / REVOKE 文

# PostgreSQL を
# もっともっと使う

# 高機能いろいろ

- テーブルスペース
  - データベースクラスタとは別のディスクにテーブルを構築
- PL/pgSQL
  - SQL 手続き言語でユーザが関数を定義できる
- 二相コミット
  - 分散されたトランザクション間での同期をとることができる
- テーブル継承
  - 親テーブルを継承する子テーブルを複数作成し、データをクラスタ化できる
- SQL/XML標準をサポート
  - 新しい演算子(xml関数)とXMLデータ型

# PostgreSQL を使ってみよう!

JPUG 名古屋支部 (NPUG) の仲間と
一緒に勉強しましょう!

ご清聴ありがとうございました