外部データラッパによるPostgreSQLの拡張

株式会社メトロシステムズ
花田茂

セッションの流れ

外部データラッパの概要

- 外部データラッパとは？
- 外部データラッパの実例

外部データアクセスの流れ

- 外部データラッパに関わるオブジェクト
- 外部データへのアクセス

外部データラッパの作成

- 外部データラッパに必要なもの
- 外部データラッパ作成のTips

PostgreSQL 9.2に向けて

- 追加された機能や提案中の機能

自己紹介

氏名　：花田　茂

所属　：株式会社メトロシステムズ

メール：shigeru.hanada@gmail.com

twitter：@s87

ブログ：http://d.hatena.ne.jp/s87/

セッションの流れ

外部データラッパの概要

- 外部データラッパとは？
- 外部データラッパの実例

外部データアクセスの流れ

- 外部データラッパに関わるオブジェクト
- 外部データへのアクセス

外部データラッパの作成

- 外部データラッパに必要なもの
- 外部データラッパ作成のTips

PostgreSQL 9.2に向けて

- 追加された機能や提案中の機能

外部データラッパとは？

PostgreSQLにいくつかある
機能拡張の仕組みのひとつ

- PostgreSQLに備わった機能拡張の仕組み

	概要	PostgreSQLに含まれる同種のもの
関数	SQLから呼び出す関数を追加	pg_sleep()、dblink_query()など
演算子	データ型に応じた演算子を追加	幾何演算など
データ型	業務データを表現するデータ型を追加	IPアドレス型、幾何型、XML型など
インデックス	データ特性に適したインデックスを追加	GIN、GiSTなど
手続き言語	関数を実装する新しい言語を追加	PL/Perl、PL/pgSQLなど
フック関数	PostgreSQLの一部の処理に割り込み、独自の処理を追加/置換	auto_explainでの自動EXPLAIN実行処理の追加など
外部データラッパ	外部データを通常の表と同様に検索できる	CSVファイルなどを読み込むfile_fdw

外部データラッパとは？

PostgreSQLの外部にあるデータに

普通のSELECT文で

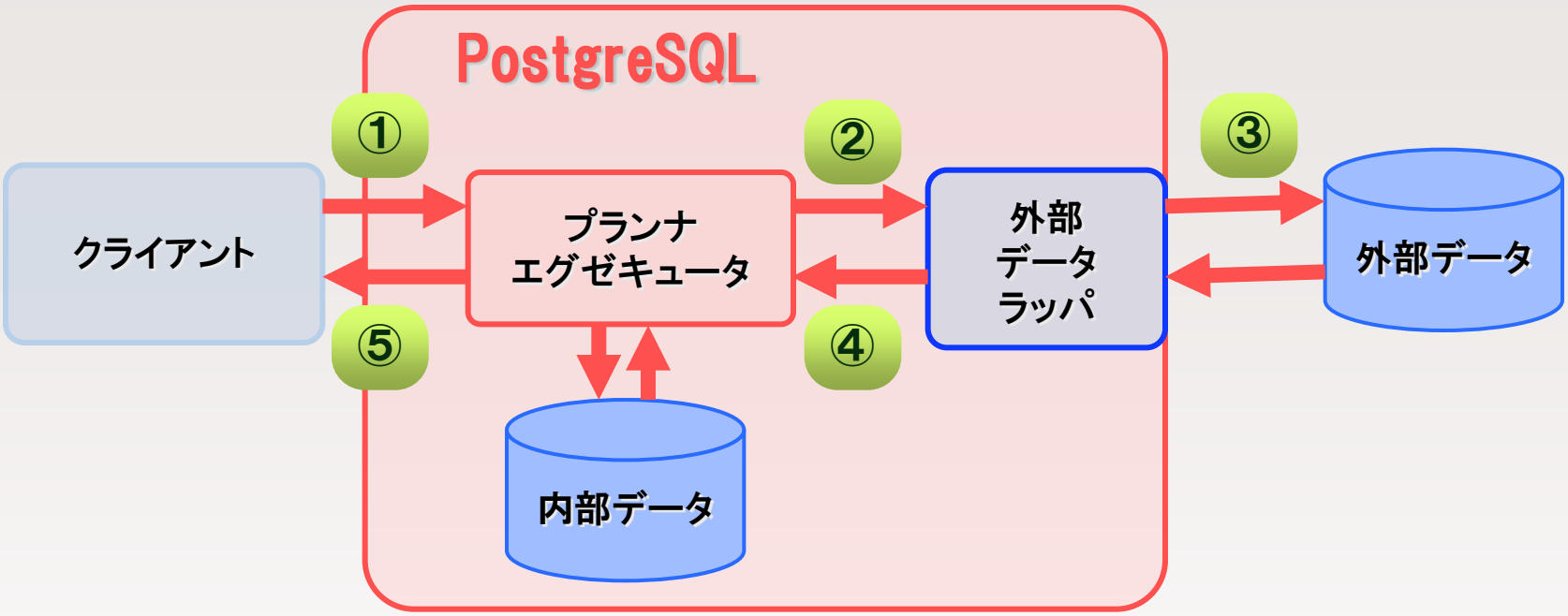
アクセスするためのしくみ

```
postgres=# SELECT * FROM ex_users WHERE id = 1;
 id | name |  birthday
----+------+------------
  1 | foo  | 2001-01-01
(1 row)
```

普通の表に見えるが
実は外部にあるデータ

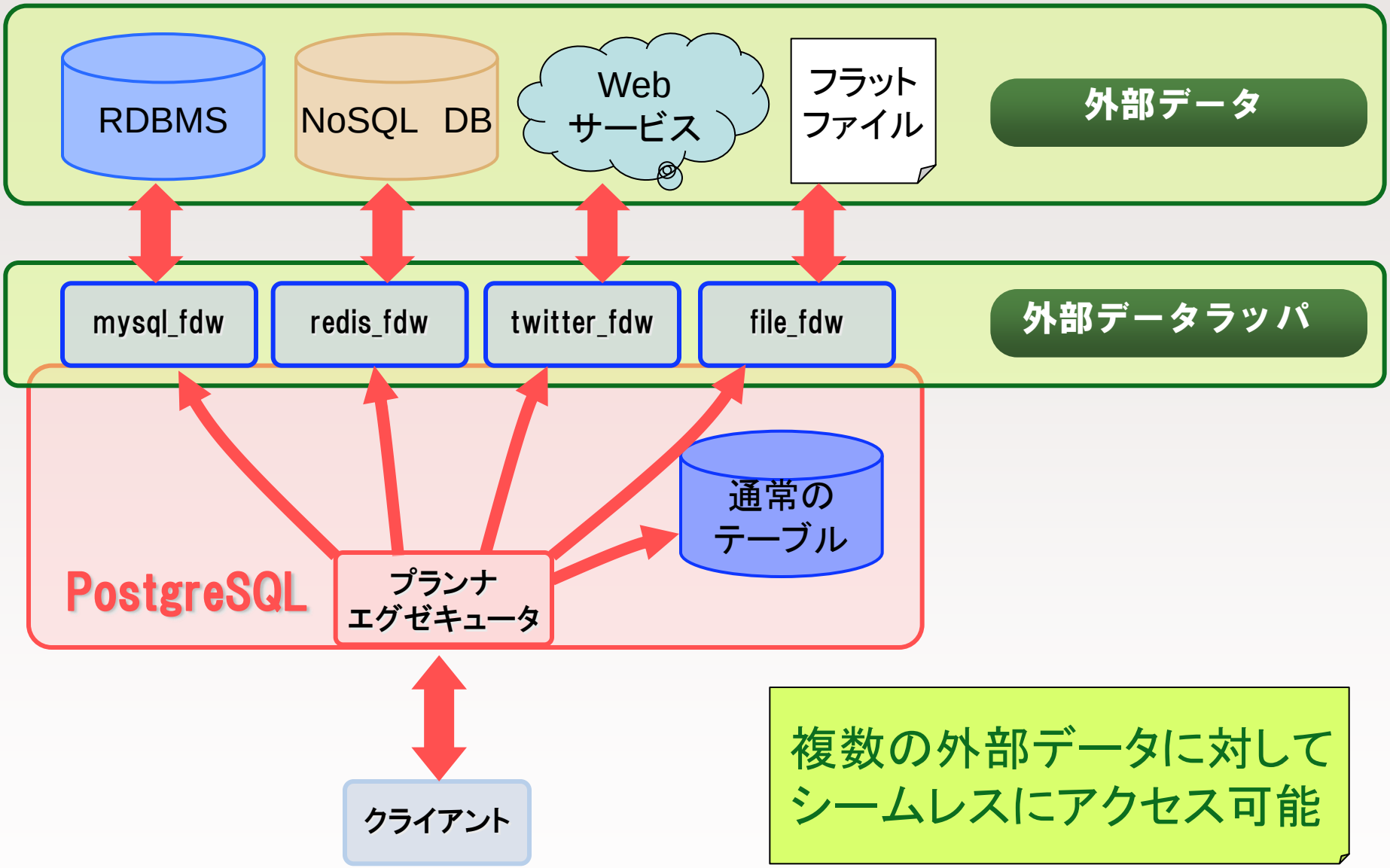
- 標準SQLの一部であるSQL/MED規格に準拠

外部データ検索の流れ



① クライアントがSELECT文を発行
② FROM句のテーブルに応じて外部データラッパに検索処理を委譲
③ FDWが検索内容に応じて外部データを取得
④ 読み取った結果をPostgreSQLの内部表現に変換して返却
⑤ クライアントに結果を返却

情報ハブとしてのPostgreSQL



複数の外部データに対して
シームレスにアクセス可能

外部データラッパの実例

mysql_fdw

- MySQLのテーブルにアクセス
- pgsnake(Dave Page)氏がPGXNにおいてEXTENSIONとして公開
- PostgreSQLからC言語ライブラリ経由でMySQLサーバにアクセス
- 他のRDBMSにあるデータをPostgreSQLでのSQL文で参照可能

使用例

```
-- MySQL にあるテーブルを検索
postgres=# select id, name FROM mysql_user;
 id |    name
----+------------
  1 | PostgreSQL
  2 | MySQL
  3 | Firebird
(3 rows)

postgres=#
```

```
-- MySQL 側での実行結果
mysql> select * From user;
+------+------------+
| id   | name       |
+------+------------+
|    1 | PostgreSQL |
|    2 | MySQL      |
|    3 | Firebird   |
+------+------------+
3 rows in set (0.00 sec)

mysql>
```

MySQLのデータが
PostgreSQLの検索結果に

外部データラッパの実例

redis_fdw

- Key-ValueストアであるRedisにアクセス
- pgsnake(Dave Page)氏がPGXNにおいてEXTENSIONとして公開
- PostgreSQLからRedisサーバにアクセス
- NoSQL DBにあるデータをPostgreSQLでのSQL文で参照可能

使用例

```
-- Redis に格納されたデータから key が 1～3 のものを検索
postgres=# select * from redis WHERE key BETWEEN '1' AND '3' ORDER BY key;
NOTICE:  Got qual key = 1
NOTICE:  Got qual key = 3
 key |   value
-----+------------
 1   | PostgreSQL
 2   | MySQL
 3   | Firebird
(3 行)

postgres=#
```

Redisからは不定順で返却されるので、PostgreSQL側でソートしている

外部データラッパの実例

twitter_fdw

- Twitterのつぶやきを条件指定で検索
- umitanuki氏がPGXNにおいてEXTENSIONとして公開
- PostgreSQLからcurl＋twitter API経由でTwitterサービスにアクセス
- Webサービスのデータを直接参照可能

使用例

```
--  “PostgreSQL” を含むつぶやきの id と 内容の先頭を検索
postgres=# SELECT id, substr(text, 1, 30) AS "text" FROM twitter
postgres-# WHERE q = 'PostgreSQL' ORDER BY created_at LIMIT 3;
         id         |                text
--------------------+--------------------------------
 165256379513307136 | Tengo que retomar PostgreSQL ,
 165243243322286080 | Develop your PostgreSQL Skills
 165237194234142720 | PostgreSQLコンファレンス
(3 rows)

postgres=#
```

LIMITを記述することで
PostgreSQL側で
件数制限が可能

外部データラッパの実例

file_fdw

- PostgreSQLが動作するサーバ上のファイルにアクセスするFDW
- contrib/file_fdwとして、PostgreSQLに同梱(9.1から)
- COPYコマンドと同じファイルフォーマットに対応
- 外部ファイルのデータをテーブルにロードせずに随時参照可能

使用例

```
-- LOG レベルのサーバをログを CSV フォーマットのログファイルから検索
postgres=# select log_time, message From postgres_log_current WHERE error_severity = 'LOG';
          log_time          |                              message
----------------------------+-------------------------------------------------------------
 2012-02-03 10:21:26.929+09 | database system was shut down at 2012-02-03 10:21:25 JST
 2012-02-03 10:21:27.097+09 | database system is ready to accept connections
 2012-02-03 10:21:27.099+09 | autovacuum launcher started
(3 rows)
postgres=#
```

WHERE句を書くことで絞込みが可能

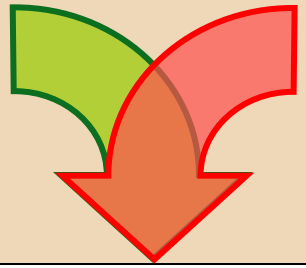
外部データラッパを組み合わせると

MySQLとRedisのデータを結合して検索

- MySQLにはデータベースのIDと名称を格納
- Redisにはデータベースの名称とバージョンを格納

```
mysql> select * From db_name;
+------+------------+
| id   | name       |
+------+------------+
|    1 | PostgreSQL |
|    2 | MySQL      |
|    3 | Firebird   |
+------+------------+
```

```
redis>get PostgreSQL
"9.1.2"
redis>get MySQL
"5.5.18"
redis>get Firebird
"2.5.1"
```



```
postgres=# SELECT m.id, m.name, r.value AS version
postgres-# FROM redis_db_version r JOIN mysql_db_name m ON m.name = r.key;
 id |    name    | version
----+------------+---------
  1 | PostgreSQL | 9.1.2
  2 | MySQL      | 5.5.18
  3 | Firebird   | 2.5.1
(3 行)
```

結合結果をPostgreSQLの結果セットとして利用可能

セッションの流れ

外部データラッパの概要

- 外部データラッパとは？
- 外部データラッパの実例

外部データアクセスの流れ

- 外部データラッパに関わるオブジェクト
- 外部データへのアクセス

外部データラッパの作成

- 外部データラッパに必要なもの
- 外部データラッパ作成のTips

PostgreSQL 9.2に向けて

- 追加された機能や提案中の機能

外部データラッパに関わるオブジェクト

外部データラッパに関連するオブジェクトは階層構造
上位のオブジェクトから順に作成していく

外部データラッパ（FOREIGN DATA WRAPPER）

外部データの種類ごとに作成
例）ファイル、MySQL、Redis、etc.

```
CREATE FOREIGN DATA WRAPPER fdw_name
    HANDLER handler_func
    VALIDATOR validator_func
    OPTIONS (key 'value');
```

外部サーバ（SERVER）

外部データの格納場所ごとに作成
例）mysql://host:port/database

```
CREATE SERVER server_name
    FOREIGN DATA WRAPPER fdw_name
    OPTIONS (key 'value);
```

ユーザマッピング（USER MAPPING）

ローカルユーザごとに作成
例）postgresユーザ、public（デフォルト）

```
CREATE USER MAPPING FOR user_name
    SERVER server_name
    OPTIONS (key 'value);
```

8.4

9.1

外部テーブル（FOREIGN TABLE）

表として見せるかたまりごとに作成
例）特定のCSVファイル、外部DBのテーブル

```
CREATE FOREIGN TABLE relname
    (attname type, ...)
    SERVER server_name
    OPTIONS (key, 'value');
```

オブジェクト：外部データラッパ

FOREIGN DATA WRAPPER

- PostgreSQLからFDW依存の処理を呼び出すためのハンドラ関数を指定
- 設定されたFDWオプションを検証するためのバリデータ関数を指定
- 外部データラッパ単位で保持するFDWオプションを指定
- 通常一つだけ必要なので、インストール時に自動的に作成されることも

例

```
postgres=# CREATE FOREIGN DATA WRAPPER mysql_fdw
postgres-# HANDLER mysql_fdw_handler
postgres-# VALIDATOR mysql_fdw_validator;
CREATE FOREIGN DATA WRAPPER
postgres=# ¥dew
                        List of foreign-data wrappers
    Name    |  Owner   |      Handler       |     Validator
------------+----------+--------------------+--------------------
 mysql_fdw | postgres | mysql_fdw_handler | mysql_fdw_validator
(1 row)
```

ハンドラ関数（事前にSQL関数を定義）

バリデータ関数（事前にSQL関数を定義）

psqlの¥dewコマンドで確認可能

オブジェクト：外部サーバ

SERVER(FOREIGN SERVER)

- 使用する外部データラッパを指定
- 外部サーバ単位で保持するFDWオプションを指定
- 一つの外部データラッパにつき複数を作成することが可能
- 接続先がRDBMS向けの場合、データベースごとなどに作成

例

```
postgres=# CREATE SERVER asset_db
postgres-# FOREIGN DATA WRAPPER mysql_fdw
postgres-# OPTIONS (address '192.168.1.101', port '3306');
CREATE SERVER
postgres=# ¥des
          List of foreign servers
   Name    |  Owner   | Foreign-data wrapper
-----------+----------+----------------------
 asset_db | postgres | mysql_fdw
(1 row)
```

psqlの¥desコマンドで確認可能

オブジェクト：ユーザマッピング

USER MAPPING

- 外部データにアクセスするローカルユーザを指定(「public」も指定可)
- 接続先外部サーバを指定
- ユーザマッピングごとに保持するFDWオプションを指定
- FDWの実装によっては作成不要(file_fdwなど)

例

```
postgres=# CREATE USER MAPPING FOR postgres
postgres-# SERVER asset_db
postgres-# OPTIONS (username 'mysql', password 'secret');
CREATE USER MAPPING
postgres=# ¥deu
List of user mappings
  Server   | User name
----------+-----------
 asset_db | postgres
(1 row)
```

psqlの¥deuコマンドで確認可能

オブジェクト：外部テーブル

FOREIGN TABLE

- 外部データの構造を表す列定義を指定
- 外部データが存在する外部サーバを指定
- 外部テーブルごとに保持するFDWオプションを指定
- この外部テーブルをSELECTすることで外部データラッパが外部データを取得

例

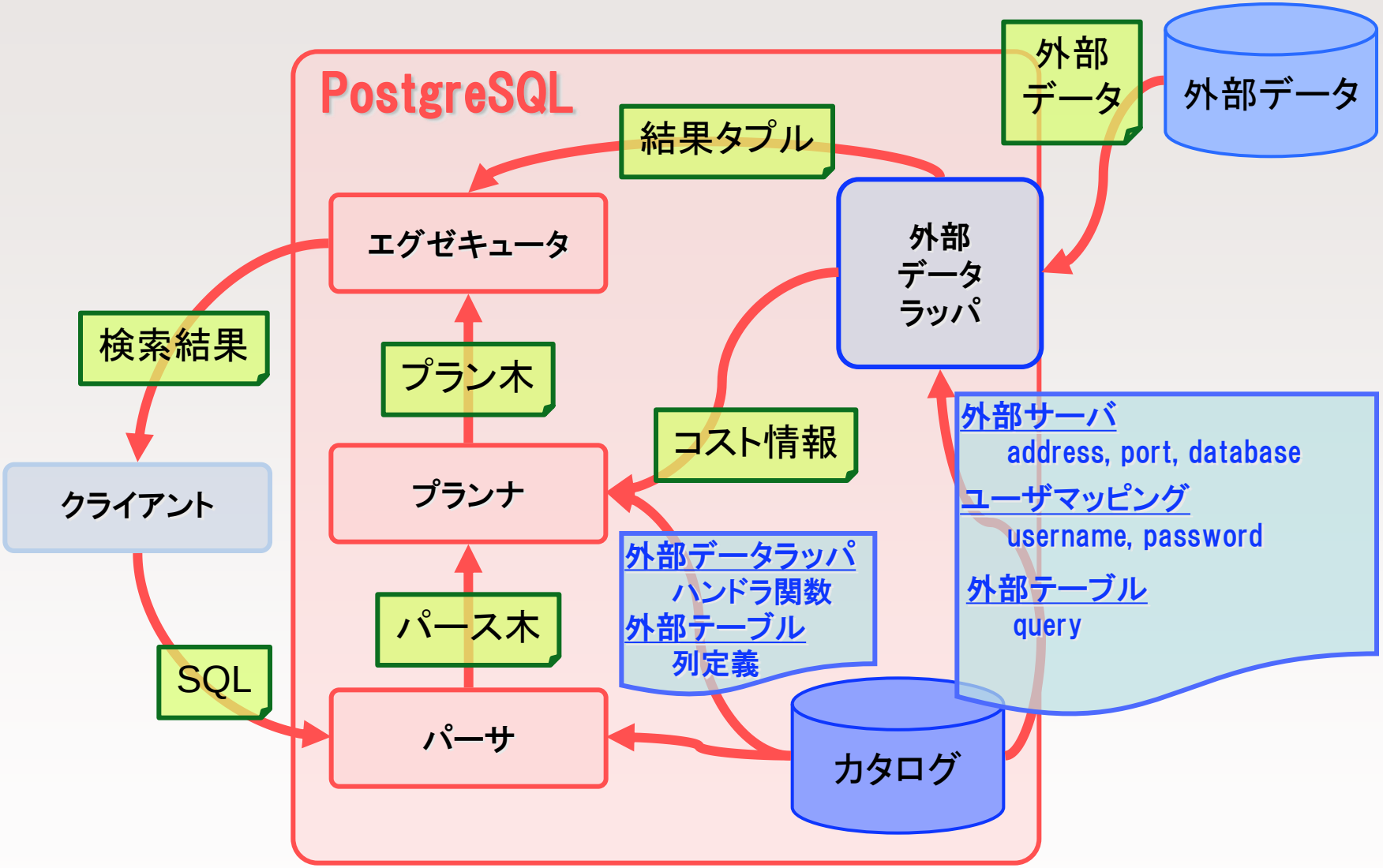
```
postgres=# CREATE FOREIGN TABLE assets (
postgres-#     id int NOT NULL,
postgres-#     name text NOT NULL,
postgres-#     buy_date date,
postgres-#     note text
postgres-# ) SERVER asset_db
postgres-# OPTIONS (query 'SELECT * FROM assets');
CREATE FOREIGN TABLE
```

```
postgres=# ¥det
   List of foreign tables
 Schema | Table  |  Server
--------+--------+----------
 public | assets | asset_db
(1 row)

postgres=#
```

psqlの¥detコマンドや
¥dコマンドで確認可能

外部データへのアクセス



外部テーブル検索のプラン

MySQLとRedisのデータを結合して検索

```
postgres=# EXPLAIN SELECT m.id, m.name, r.value AS version
postgres-# FROM redis_db_version r JOIN mysql_db_name m ON m.name = r.key
postgres-# ORDER BY m.id;
                                        QUERY PLAN
-------------------------------------------------------------------------------------------
 Sort  (cost=26.15..26.16 rows=1 width=68)
   Sort Key: m.id
   ->  Nested Loop  (cost=20.00..26.14 rows=1 width=68)
         Join Filter: (r.key = m.name)
         ->  Foreign Scan on redis_db_version r  (cost=10.00..13.00 rows=3 width=64)
               Foreign Redis Database Size: 3
         ->  Materialize  (cost=10.00..13.02 rows=3 width=36)
               ->  Foreign Scan on mysql_db_name m  (cost=10.00..13.00 rows=3 width=36)
                     Local server startup cost: 10
                     MySQL query: SELECT * FROM db_name
(10 行)
```

【検索結果】

```
 id |    name    | version
----+------------+---------
  1 | PostgreSQL | 9.1.2
  2 | MySQL      | 5.5.18
  3 | Firebird   | 2.5.1
(3 行)
```

セッションの流れ

外部データラッパの概要

- 外部データラッパとは？
- 外部データラッパの実例

外部データへのアクセス

- 外部データラッパに関わるオブジェクト
- 外部データへのアクセス

外部データラッパの作成

- **外部データラッパに必要なもの**
- **外部データラッパ作成のTips**

PostgreSQL 9.2に向けて

- 追加された機能や提案中の機能

外部データラッパに必要なもの

C言語で外部データにアクセスできれば、外部データラッパを作れます！

ハンドラ関数

プランナやエグゼキュータが呼び出すC言語関数のポインタを保持するFdwRoutine構造体を返すSQL関数です。

バリデータ関数

各種の外部データラッパ関連オブジェクトに設定されたFDWオプションの妥当性を検査するSQL関数です。検査が不要ならば省略可能です。

ほんの少しの元気と勇気

現時点ではPostgreSQLの内部構造に関する知識がある程度必要です。

ハンドラ関数が必要とするC言語関数

関数名	呼び出しタイミング	必要な処理
PlanForeignScan	プランナが外部テーブル単体のスキャンのコストを知りたいとき	外部テーブルのスキャンコストを見積もり、スキャン実行時に必要な情報と一緒にFdwPlan構造体に格納する。
BeginForeignScan	エグゼキュータでプランを実行し始めるとき	外部テーブルのスキャンに必要な情報をカタログなどから取得し、以降の関数で使用できるようにFofeignScanStateに格納する。
IterateForeignScan	エグゼキュータが次の行を必要とするとき	外部データから一行分のデータを読み取り、Tupleを生成して返す。スキャンが末尾に到達した場合はそれを通知する。
ReScanForeignScan	Nested LoopのInner Scanがリセットされるとき(Outer Scanが次のタプルに進んだとき)	外部データからの読み取り位置を先頭にリセットする。
EndForeignScan	エグゼキュータでプラン実行が完了したとき	使用したリソースを解放する。
ExplainForeignScan	外部テーブルを含むSELECT文のEXPLAINコマンドが実行されたとき	外部データラッパ特有のプラン情報(リモートに発行するクエリなど)がある場合は、ExplainPropertyText()などで追加する。

ハンドラ関数の仕様

引数

- なし

戻り値

- FdwRoutineオブジェクトのポインタを返す
- SQL関数の戻り値はfdw_handler型(これにより、SQLからは呼べなくなる)

```
【SQL関数】
CREATE FUNCTION pgsql_fdw_handler()
RETURNS fdw_handler AS
'<ライブラリファイル名>' LANGUAGE C STRICT;


【C言語関数】
Datum
pgsql_fdw_handler(PG_FUNCTION_ARGS)
{
    PG_RETURN_POINTER(&fdwroutine);
}
```

```
static FdwRoutine fdwroutine = {
    T_FdwRoutine,
    pgsqlPlanForeignScan,
    pgsqlExplainForeignScan,
    pgsqlBeginForeignScan,
    pgsqlIterateForeignScan,
    pgsqlReScanForeignScan,
    pgsqlEndForeignScan,
};
```

バリデータ関数の仕様

引数

- DDLで指定されたオプションの名前/値ペア(textの配列型)
- DDLで指定されたオブジェクトを管理するカタログ種別(pg_classのOID)

戻り値

- なし(void)
- 全て妥当なオプションであれば、呼び出し元に制御を戻す
- 不正なオプションがあった場合は、内部でereport()などでエラーにする

【SQL関数】

```
CREATE FUNCTION pgsql_fdw_validator(text[], oid)
RETURNS void AS
'<ライブラリファイル名>' LANGUAGE C STRICT;
```

【C言語関数】

```
Datum
pgsql_fdw_validator(PG_FUNCTION_ARGS)
{
    ...
    if (invalid_option_found)
        elog(ERROR, "invalid option");
    ...
}
```

外部データラッパ作成のTips

手始めに

- まずは既存の外部データラッパをコピーしていろいろ改造
- 慣れてきたら、独自の外部データにアクセス

C言語関数の実装での注意点

- 関数間での情報の受け渡しはfdw_private(List)で
- ただし、ここで受け渡せる情報はNode構造体を継承したもののみ
- 非Node構造体はC言語文字列化してmakeString()でValueに変換、とか

リソース管理

- 動的メモリの確保は必ずpalloc()で！
- 独自のメモリコンテキストを作成するのもあり
- 外部リソースはResourceOwnerにコールバックを登録してエラー時に解放

外部データラッパの実例(番外)

multicorn

- Pythonで外部データラッパを実装するためのフレームワーク
- http://multicorn.orgで開発、PGXNで配布中
- PostgreSQL内部の知識が不要で、Pythonのライブラリが利用可能
- ForeignDataWrapperクラスを継承しexecuteメソッドを書くだけ！

使用例

```
...
class ProcessFdw(ForeignDataWrapper):
    ...

    def execute(self, quals, columns):
        # statgrab already returns its data in a format suitable
        # for Multicorn: a list (iterable) of dicts.
        # `quals` is ignored, PostgreSQL will do the filtering itself.
        return statgrab.sg_get_process_stats()
```

実質この一行だけで、プロセス一覧がSELECT文で取得可能

セッションの流れ

外部データラッパの概要

- 外部データラッパとは？
- 外部データラッパの実例

外部データアクセスの流れ

- 外部データラッパに関わるオブジェクト
- 外部データへのアクセス

外部データラッパの作成

- 外部データラッパに必要なもの
- 外部データラッパ作成のTips

PostgreSQL 9.2に向けて

- 追加された機能や提案中の機能

FDWオブジェクトの強化

- 外部テーブルの各列にFDWオプションが指定可能
- psqlの¥d*+でFDWオプションをDDLと同じ書式で表示→コピペ可能
- 外部データラッパ、外部サーバがCOMMENT ON文に対応
- これらの機能は、9.2での採用が決定しています！

使用例

```
-- 列単位のFDWオプションを設定
postgres=# CREATE FOREIGN TABLE remote_users (
postgres-#     id int NOT NULL OPTIONS (colname 'user_id'),
postgres-#     name text NOT NULL OPTIONS (colname 'user_name'),
postgres-#     ...
postgres-# ) SERVER pgsql_fdw OPTIONS (relname 'users');
postgres-# ¥det+
```

リモート側の列名などを
適切な単位で設定可能

PostgreSQL 9.2に向けて

pgsql_fdw

- 外部のPostgreSQLサーバのデータにアクセス
- PostgreSQL本体のcontribモジュールとして提案中
- PostgreSQLからlibpq経由で外部サーバにアクセス
- 一部のWHERE句をリモート側で評価し転送量を削減可能！

使用例

```
postgres=# EXPLAIN SELECT tid, tbalance FROM pgbench_tellers WHERE bid = 1;
                                                                QUERY PLAN
------------------------------------------------------------------------------------------------------------------------
-----------------------------------
 Foreign Scan on pgbench_tellers  (cost=100.00..137.38 rows=10 width=12)
   Remote SQL: DECLARE pgsql_fdw_cursor_0 SCROLL CURSOR FOR SELECT tid, bid, tbalance, NULL FROM
public.pgbench_tellers WHERE (bid = 1)
(2 rows)
```

WHERE句をリモート側で評価し
データ転送量を削減

外部テーブルのANALYZE対応

- 外部テーブルの統計情報をANALYZEコマンドで収集可能に
- PostgreSQL本体の追加機能として提案中
- 統計情報の収集処理は各外部データラッパに委譲
- 検索条件やJOIN条件に関するコスト見積もりの精度が向上！

使用例

```
postgres=# EXPLAIN ANALYZE SELECT tid, tbalance FROM pgbench_tellers WHERE bid = 1;
                                                              QUERY PLAN
---------------------------------------------------------------------------------------------------------------------------------------
 Foreign Scan on pgbench_tellers  (cost=100.00..137.96 rows=10 width=12) (actual time=46.363..46.740
rows=10 loops=1)
   Filter: (bid = 1)
   Rows Removed by Filter: 20
   Remote SQL: DECLARE pgsql_fdw_cursor_2 SCROLL CURSOR FOR SELECT tid, bid, tbalance, …
 Total runtime: 48.167 ms
(5 rows)
```

見積もり精度が向上

外部データラッパに関する情報はこんなところに

PostgreSQL Wiki

http://wiki.postgresql.org/wiki/Foreign_data_wrappers

開発中のものも含めて、様々な外部データラッパがリストアップされ、配布場所やソースリポジトリなどがリンクされています。

PGXN

http://pgxn.org/

本セッションで紹介した外部データラッパは、主にこのサイトで配布されています。PostgreSQLのEXTENSIONをシェルコマンドで管理するツールもあります。