YOUR ONE-STOP DATA SHOP - AN INTRO TO FOREIGN DATA WRAPPERS IN POSTGRES

Sam Bail @spbail

PostgresOpen 2019

HI, I'M SAM

PhD in Semantic Web, knowledge representation, automated reasoning

Spent 5 ½ years at Flatiron Health in NYC analyzing oncology data

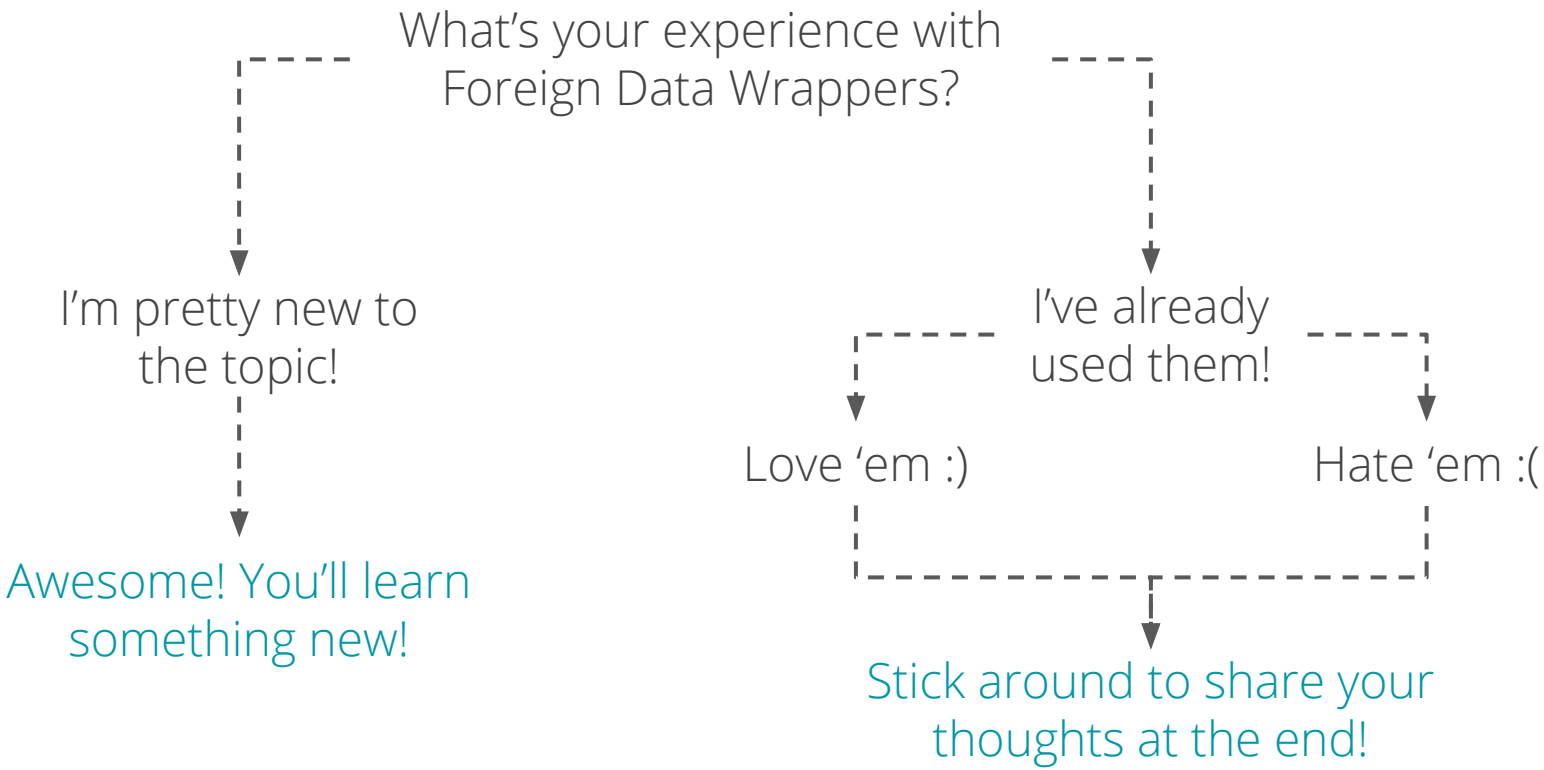
Less big data, more artisanal handcrafted data

Used PostgreSQL, SQL Server, in-house ETL tooling, lots of Python

Looking at PostgreSQL topics from a user perspective

Twitter: @spbail (mostly data/tech, New York rants, and bunny pictures)

IS THIS TALK FOR ME?



OUTLINE

1 Why and what?

2 FDW in the wild

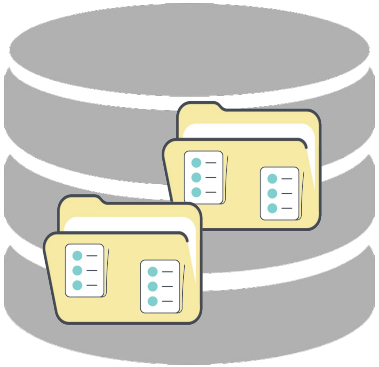
3 BYO FDW!

4 Wrap-up

1 - WHY AND WHAT

Your one-stop data shop

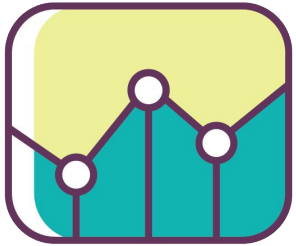
THE PROBLEM - PT1



Data in db1

```
postgres=# select * from zoo order by nickname;
 id  | nickname  |              type              | gender | age
-----+-----------+--------------------------------+--------+-----
  45 | Adey      | Wallaby, bennett's             | Female |   5
  41 | Aidan     | Insect, stick                  | Female |  16
  73 | Aliza     | Little cormorant               | Female |  25
  68 | Alonzo    | Red and blue macaw             | Male   |   3
  83 | Alysa     | Spurfowl, yellow-necked        | Female |   3
  35 | Alyse     | Kongoni                        | Female |  20
  37 | Amalita   | Bohor reedbuck                 | Female |   6
  91 | Andrus    | Red-breasted cockatoo          | Male   |  10
  14 | Auguste   | Ring-necked pheasant           | Female |   7
  92 | Baillie   | Greater kudu                   | Male   |  19
  55 | Base      | White-rumped vulture           | Male   |   2
  33 | Bethina   | North American red fox         | Female |  21
  17 | Bette-ann | Peacock, blue                  | Female |  25
  28 | Birdie    | Arctic tern                    | Female |  22
   4 | Bondy     | Eland, common                  | Male   |  11
```

SQL queries



Insights,
apps, etc.

THE PROBLEM - PT2

db1

db2

```
postgres=# select * from zoo order by nickname;
 id  | nickname  |              type              | gender | age
-----+-----------+--------------------------------+--------+-----
  45 | Adey      | Wallaby, bennett's             | Female |   5
  41 | Aidan     | Insect, stick                  | Female |  16
  73 | Aliza     | Little cormorant               | Female |  25
  68 | Alonzo    | Red and blue macaw             | Male   |   3
  83 | Alysa     | Spurfowl, yellow-necked        | Female |   3
  35 | Alyse     | Kongoni                        | Female |  20
  37 | Amalita   | Bohor reedbuck                 | Female |   6
  91 | Andrus    | Red-breasted cockatoo          | Male   |  10
  14 | Auguste   | Ring-necked pheasant           | Female |   7
  92 | Baillie   | Greater kudu                   | Male   |  19
  55 | Base      | White-rumped vulture           | Male   |   2
  33 | Bethina   | North American red fox         | Female |  21
  17 | Bette-ann | Peacock, blue                  | Female |  25
  28 | Birdie    | Arctic tern                    | Female |  22
   4 | Bondy     | Eland, common                  | Male   |  11
```

SQL queries

Insights,
apps, etc.

THE PROBLEM - PT2

Cross-database querying doesn't
work out of the box!
Do we need an ETL pipeline?

db1

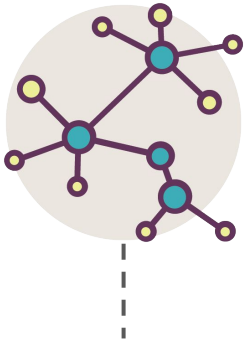
db2

SQL queries

Insights,
apps, etc.

* AFAIK, SQL Server supports cross-DB and cross-server queries, MySQL uses "federated storage engine", Oracle has "database link"...

A SOLUTION - POSTGRES EXTENSIONS



Pre-PostgreSQL 9.1:
dblink

PostgreSQL 9.1+:
Read-only Foreign Data Wrapper (FDW)

PostgreSQL 9.3+:
INSERT/UPDATE/DELETE with FDW

SAMPLE DATA

```
db1=# SELECT * FROM zoo LIMIT 5;
 id | nickname |          type          | gender | age
----+----------+------------------------+--------+-----
  1 | Tadd     | Turaco, violet-crested | Male   |   7
  2 | Zuzana   | Vervet monkey          | Female |  17
  3 | Taryn    | Phascogale, red-tailed | Female |   8
  4 | Bondy    | Eland, common          | Male   |  11
  5 | Janette  | Python, carpet         | Female |  25
(5 rows)
```

```
db2=# SELECT * FROM animal_lookup LIMIT 5;
 id |            common_name            |       scientific_name
----+-----------------------------------+------------------------------
  1 | Yellow-necked spurfowl            | Francolinus leucoscepus
  2 | Tiger snake                       | Notechis semmiannulatus
  3 | Hornbill, yellow-billed           | Tockus flavirostris
  4 | Malagasy ground boa               | Acrantophis madagascariensis
  5 | Tarantula, salmon pink bird eater | Lasiodora parahybana
(5 rows)
```

What if I want Janette the carpet python's scientific name from db2? We can't just join tables across DBs...

DBLINK EXAMPLE - PT1

```
db1=# CREATE EXTENSION dblink;

db1=# SELECT *
db1-# FROM dblink('dbname=db2', 'SELECT common_name, scientific_name FROM animal_lookup')
db1-# AS (common_name TEXT, scientific_name TEXT);
             common_name              |           scientific_name
--------------------------------------+-------------------------------------
 Yellow-necked spurfowl               | Francolinus leucoscepus
 Tiger snake                          | Notechis semmiannulatus
 Hornbill, yellow-billed              | Tockus flavirostris
 Malagasy ground boa                  | Acrantophis madagascariensis
 Tarantula, salmon pink bird eater    | Lasiodora parahybana
 Stork, white                         | Ciconia ciconia
 Scarlet macaw                        | Ara macao
 Dunnart, fat-tailed                  | Smithopsis crassicaudata
 Common boubou shrike                 | Laniarius ferrugineus
 Dusky rattlesnake                    | Crotalus triseriatus
...
```

DBLINK EXAMPLE - PT2

```
db1=# SELECT z.*, a.scientific_name
FROM zoo z
LEFT JOIN (
SELECT *
FROM dblink('dbname=db2', 'SELECT common_name, scientific_name FROM animal_lookup')
AS (common_name TEXT, scientific_name TEXT)
) a ON z.type = a.common_name;
 id  | nickname  |             type             | gender | age |        scientific_name
-----+-----------+------------------------------+--------+-----+------------------------------
  22 | Giacopo   | Common boubou shrike         | Male   |  10 | Laniarius ferrugineus
  30 | Florida   | Gecko, ring-tailed           | Female |  14 | Cyrtodactylus louisiadensis
  13 | Joe       | Wallaby, red-necked          | Male   |  21 | Macropus rufogriseus
  41 | Aidan     | Insect, stick                | Female |  16 | Leprocaulinus vipera
  93 | Remus     | Bat, little brown            | Male   |  19 | Myotis lucifugus
  22 | Giacopo   | Common boubou shrike         | Male   |  10 | Laniarius ferrugineus
  20 | Ingmar    | Mongoose, eastern dwarf      | Male   |  25 | Helogale undulata
  36 | Lawton    | Jackal, silver-backed        | Male   |  22 | Canis mesomelas
  68 | Alonzo    | Red and blue macaw           | Male   |   3 | Ara chloroptera
...
```

FDW EXAMPLE - PT1

```
db1=# CREATE EXTENSION postgres_fdw;

db1=# CREATE SERVER db2foreign
db1-# FOREIGN DATA WRAPPER postgres_fdw
db1-# OPTIONS (host 'localhost', dbname 'db2');

db1=# CREATE USER MAPPING FOR sam
db1-# SERVER db2foreign
db1-# OPTIONS (user 'sam');

db1=# CREATE FOREIGN TABLE animal_lookup (
db1(#   id INTEGER,
db1(#   common_name TEXT,
db1(#   scientific_name TEXT)
db1-# SERVER db2foreign
db1-# OPTIONS (table_name 'animal_lookup');
```

Load (the built-in) extension - needs to be done only once per db

Create foreign server - needs to be done only once per foreign data source

Create a user mapping from user in db1 to user in db2 (can be different user names)

Create the foreign table (do this once for each table - alternatively, import a foreign schema to create the foreign tables)

FDW EXAMPLE - PT2

```
db1=# SELECT z.*, a.scientific_name
db1-# FROM zoo z
db1-# LEFT JOIN animal_lookup a ON (z.type = a.common_name);
 id  | nickname  |             type             | gender | age |        scientific_name
-----+-----------+------------------------------+--------+-----+-------------------------------
  22 | Giacopo   | Common boubou shrike         | Male   |  10 | Laniarius ferrugineus
  30 | Florida   | Gecko, ring-tailed           | Female |  14 | Cyrtodactylus louisiadensis
  13 | Joe       | Wallaby, red-necked          | Male   |  21 | Macropus rufogriseus
  41 | Aidan     | Insect, stick                | Female |  16 | Leprocaulinus vipera
  93 | Remus     | Bat, little brown            | Male   |  19 | Myotis lucifugus
  22 | Giacopo   | Common boubou shrike         | Male   |  10 | Laniarius ferrugineus
  20 | Ingmar    | Mongoose, eastern dwarf      | Male   |  25 | Helogale undulata
  36 | Lawton    | Jackal, silver-backed        | Male   |  22 | Canis mesomelas
  68 | Alonzo    | Red and blue macaw           | Male   |   3 | Ara chloroptera
  46 | Shaine    | Cat, european wild           | Female |   9 | Felis silvestris lybica
  36 | Lawton    | Jackal, silver-backed        | Male   |  22 | Canis mesomelas
  26 | Rees      | Civet, common palm           | Male   |  11 | Paradoxurus hermaphroditus
  37 | Amalita   | Bohor reedbuck               | Female |   6 | Redunca redunca
...
```

FDW UNDER THE HOOD

PostgreSQL Source Code git master

Several K lines of C code

Main Page | Namespaces | Data Structures | Files

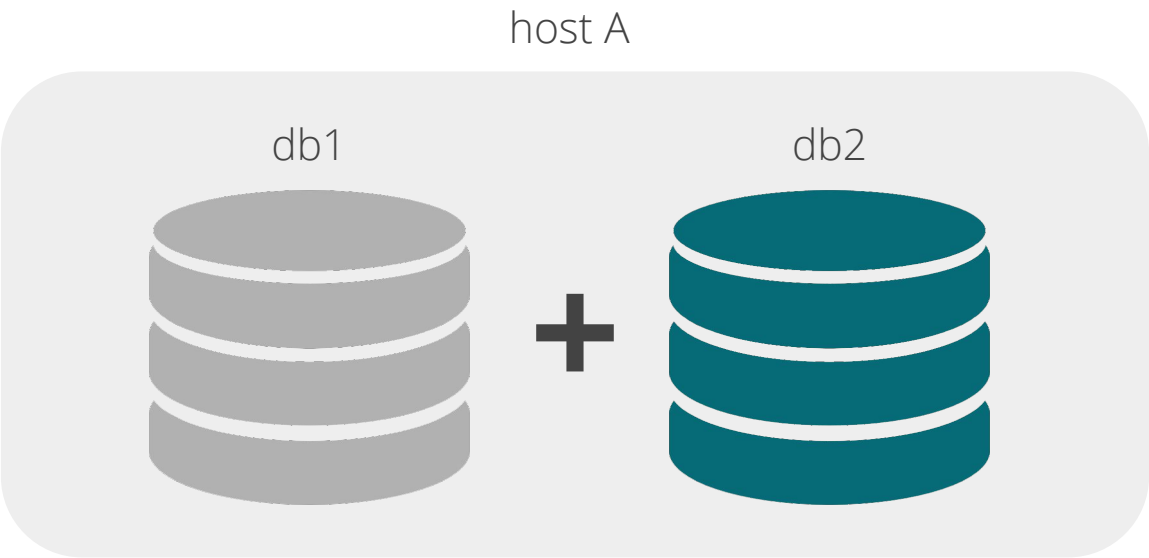
- passwordcheck
- pg_buffercache
- pg_freespacemap
- pg_prewarm
- pg_standby
- pg_stat_statements
- pg_trgm
- pg_visibility
- pgcrypto
- pgrowlocks
- pgstattuple
- postgres_fdw
  - connection.c
  - deparse.c
  - option.c
  - postgres_fdw.c
  - postgres_fdw.h
  - shippable.c
- seg
- sepgsql
- spi
- sslinfo
- tablefunc
- tcn

```c
    /* Look up foreign-table catalog info. */
    fpinfo->table = GetForeignTable(foreigntableid);
    fpinfo->server = GetForeignServer(fpinfo->table->serverid);

    /*
     * Extract user-settable option values.  Note that per-table setting of
     * use_remote_estimate overrides per-server setting.
     */
    fpinfo->use_remote_estimate = false;
    fpinfo->fdw_startup_cost = DEFAULT_FDW_STARTUP_COST;
    fpinfo->fdw_tuple_cost = DEFAULT_FDW_TUPLE_COST;
    fpinfo->shippable_extensions = NIL;
    fpinfo->fetch_size = 100;

    apply_server_options(fpinfo);
    apply_table_options(fpinfo);

    /*
     * If the table or the server is configured to use remote estimates,
     * identify which user to do remote access as during planning.  This
     * should match what ExecCheckRTEPerms() does.  If we fail due to lack of
     * permissions, the query would have failed at runtime anyway.
     */
    if (fpinfo->use_remote_estimate)
    {
        Oid         userid = rte->checkAsUser ? rte->checkAsUser : GetUserId();

        fpinfo->user = GetUserMapping(userid, fpinfo->server->serverid);
    }
    else
        fpinfo->user = NULL;

    /*
     * Identify which baserestrictinfo clauses can be sent to the remote
     * server and which can't.
     */
    classifyConditions(root, baserel, baserel->baserestrictinfo,
                       &fpinfo->remote_conds, &fpinfo->local_conds);
```

2 - FDW IN THE WILD

Some examples of Foreign Data Wrappers in action

EXAMPLE 1: SAME HOST



Multiple **Postgres** DBs on the **same** host (already demo'd)

EXAMPLE 2: DIFFERENT HOSTS

Multiple **Postgres** DBs on
different hosts

EXAMPLE 2: DIFFERENT HOSTS

RNAcentral

Search by gene, species, accession, or any keyword

Search

Examples: KCNQ1OT1, Mus musculus, snoRNA, miRBase, hsa-mir-126

How to search

v12 Databases Tools API Downloads Browse About Help Feedback

Public Postgres database

In addition to downloadable files, an API, and the text search, RNAcentral provides a public Postgres database that can be used to query the data using SQL syntax. The database is updated with every RNAcentral release and contains a copy of the data available through the RNAcentral website.

Connection details

- Hostname: `hh-pgsql-public.ebi.ac.uk`
- Port: `5432`
- Database: `pfmegrnargs`
- User: `reader`
- Password: `NWDMCE5xdipIjRrp`

Help topics

About RNAcentral

- Frequently Asked Questions
- Scientific Advisory Board
- Training resources
- Data growth timeline
- Linking to RNAcentral

Data Access

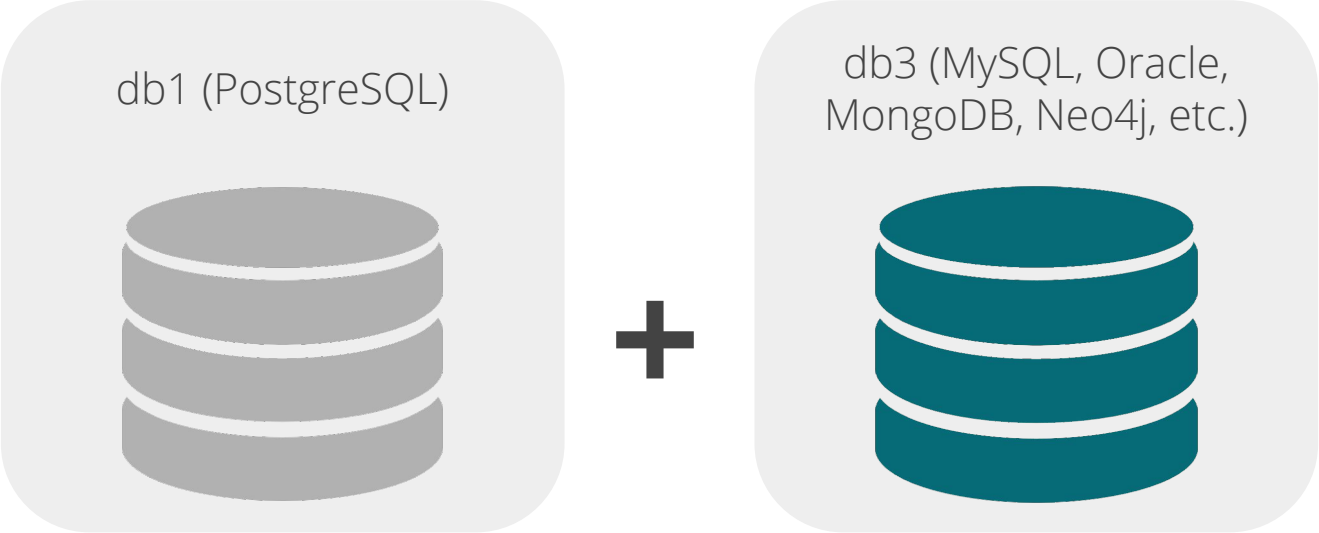
https://rnacentral.org/help/public-database

EXAMPLE 2: DIFFERENT HOSTS

```
db1=# CREATE SERVER ebiserver
db1-# FOREIGN DATA WRAPPER postgres_fdw
db1-# OPTIONS (host 'hh-pgsql-public.ebi.ac.uk', dbname 'pfmegrnargs', port '5432');

db1=# CREATE USER MAPPING FOR sam
db1-# SERVER ebiserver
db1-# OPTIONS (user 'reader', password 'NWDMCE5xdipIjRrp');

db1=# CREATE FOREIGN TABLE protein_info (
db1(#   protein_accession TEXT,
db1(#   description TEXT,
db1(#   label TEXT,
db1(#   synonyms TEXT)
db1-# SERVER ebiserver
db1-# OPTIONS (table_name 'protein_info', schema_name 'rnacen');
```

Example of user mapping to a different user

EXAMPLE 3: PG TO OTHER DBMS



Multiple **different** relational or NoSQL DBs (on same or different hosts)

I WANT TO INTEGRATE MORE GENE DATA

EnsemblGenomes

About us | Genomes | Data types | Data access | FAQs

Bacteria | Protists | Fungi | Plants | Metazoa | Vertebrates

Data access

- Overview
  - Data export
  - Data download
    - Data archives
  - BioMart
  - Using your own data
  - Public Track Hubs
  - Programmatic access
    - REST service
    - Ensembl Perl API
      - Ensembl Genomes Perl API
    - MySQL database access
  - Linking to Ensembl Genomes
  - Building an Ensembl Genomes mirror
    - Virtual Machine (deprecated)

MySQL database access

Ensembl Genomes and the Ensembl software platform use the MySQL relational database management system to store data. MySQL databases are used by the web browser and REST service, and can be used with the Ensembl Perl API or directly with a MySQL client (see below). The schema used by the Ensembl platform are described in the Ensembl API documentation.

MySQL databases are also used by the BioMart data warehouse interface, although we recommend that you use the web interface to access data in BioMart, as the mart schema contains many tables of denormalised data. Data can also be retrieved from BioMart programmatically, using the XML-based martservice.

The Ensembl Genomes public MySQL Servers

How do we connect from pg to a MySQL db!?

Ensembl Genomes operates a MySQL server for public use which contains all databases from the last 1 ed for the BioMart interface. This server can be used in conjuction with the Ensembl public MySQL servers (though or servers). Details of Ensembl and Ensembl Genomes servers are shown below, and all servers can be accessed using the user 'anonymous' (no password required):

Dataset	Server	Port
Ensembl Genomes, all databases	mysql-eg-publicsql.ebi.ac.uk	4157
Ensembl	ensembldb.ensembl.org	5306
Ensembl Mart	martdb.ensembl.org	5316

Note: Ensembl Genomes and Ensembl MySQL servers are located at different URLs. Ensembl Genomes databases from all five divisions are located on the same server. Ensembl BioMart is on a different server. Not all MySQL instances use the default port, so please ensure that you specify the correct port when trying to connect!

The MySQL server is provided 'as-is', though scheduled downtime will be publicised on the mailing lists and on ensemblgenomes.org. Please note that if processes that use this service excessively to the detriment of other users may be terminated without warning to preserve the service functionality. For intensive use of this server, an alternative is to set up a local MySQL database with copies of Ensembl Genomes data.

http://ensemblgenomes.org/info/access/mysql

TO THE FDW WIKI!

Generic SQL Database Wrappers

Data Source	Type	License	Code	Install	Doc	Notes
ODBC	Native		github			CartoDB took over active development of the ODBC FDW for PG 9.5+
JDBC	Native		github			Not maintained ?
JDBC2	Native		github			
SQL_Alchemy	Multicorn	PostgreSQL	GitHub	PGXN	documentation	Can be used to access data stored in any database supported by the sqlalchemy python toolkit.
VirtDB	Native	GPL	GitHub			A generic FDW to access VirtDB data sources (SAP ERP, Oracle RDBMS)

Specific SQL Database Wrappers

Data Source	Type	License	Code	Install	Doc	Notes
PostgreSQL	Native	PostgreSQL	git.postgresql.org		documentation	
Oracle	Native	PostgreSQL	github	PGXN	website	
MySQL	Native		github	PGXN	example	FDW for MySQL
Informix	Native	PostgreSQL	github			
Firebird	Native	PostgreSQL	github	PGXN	README	version 1.1 released (2019-05)
SQLite	Native		github			An FDW for SQLite3 (read-only)
SQLite	Native	PostgreSQL	github	PGXN	README	An FDW for SQLite3 (write support and several pushdown optimization)
Sybase / MS SQL Server	Native		github	PGXN		An FDW for Sybase and Microsoft SQL server
MonetDB	Native		github			

NoSQL Database Wrappers

Data Source	Type	License	Code	Install	Doc	Notes
BigTable or HBase	Native Rust Binding (RPGFFI)	MIT	Github			
Cassandra	Multicorn	MIT	Github	Rankactive		

https://wiki.postgresql.org/wiki/Foreign_data_wrappers

FDW INSTALLATION

MySQL Foreign Data Wrapper for PostgreSQL

This PostgreSQL extension implements a Foreign Data Wrapper (FDW) for MySQL.

Please note that this version of mysql_fdw only works with PostgreSQL Version 9.3 and greater, for previous version support please download from PG_92 branch.

We have added a number of significant enhancements to the mysql fdw, some of the major enhancements are listed in the "enhancements" section of this document.

1. Installation

To compile the MySQL foreign data wrapper, MySQL's C client library is needed. This library can be downloaded from the official MySQL website.

1. To build on POSIX-compliant systems you need to ensure the `pg_config` executable is in your path when you run `make`. This executable is typically in your PostgreSQL installation's `bin` directory. For example:

   ```
   $ export PATH=/usr/local/pgsql/bin/:$PATH
   ```

2. The `mysql_config` must also be in the path, it resides in the MySQL `bin` directory.

   ```
   $ export PATH=/usr/local/mysql/bin/:$PATH
   ```

3. Compile the code using make.

   ```
   $ make USE_PGXS=1
   ```

4. Finally install the foreign data wrapper.

   ```
   $ make USE_PGXS=1 install
   ```

Most FDW installs are pretty similar: download, compile, install*

* Then kick and repeat until it works

EXAMPLE 3: PG TO OTHER DBMS

```
db1=# CREATE EXTENSION mysql_fdw;

db1=# CREATE SERVER ensemblserver
db1-# FOREIGN DATA WRAPPER mysql_fdw
db1-# OPTIONS (host 'mysql-eg-publicsql.ebi.ac.uk', port '4157');

db1=# CREATE USER MAPPING FOR sam
db1-# SERVER ensemblserver
db1-# OPTIONS (username 'anonymous');

db1=# CREATE FOREIGN TABLE ncbi_taxa_name (
db1(#   taxon_id INTEGER,
db1(#   name TEXT,
db1(#   name_class TEXT)
db1-# SERVER ensemblserver
db1-# OPTIONS (table_name 'ncbi_taxa_name', dbname 'ncbi_taxonomy');
```

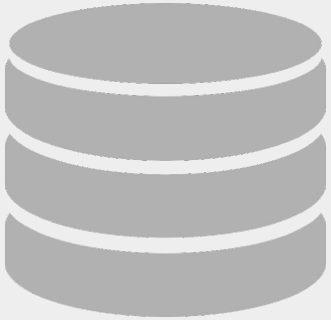
mysql_fdw takes dbname option in create foreign table statement

EXAMPLE 3: PG TO OTHER DBMS

```
db1=# SELECT * FROM ncbi_taxa_name;
 taxon_id |                                        name                                        |     name_class
----------+----------------------------------------------------------------------------------+---------------------
        1 | all                                                                              | synonym
        1 | root                                                                             | scientific name
        2 | Bacteria                                                                         | scientific name
        2 | Monera                                                                           | in-part
        2 | Procaryotae                                                                      | in-part
        2 | Prokaryota                                                                       | in-part
        2 | Prokaryotae                                                                      | in-part
        2 | bacteria                                                                         | blast name
        2 | eubacteria                                                                       | genbank common name
        2 | not Bacteria Haeckel 1894                                                        | synonym
        2 | prokaryote                                                                       | in-part
        2 | prokaryotes                                                                      | in-part
        6 | Azorhizobium                                                                     | scientific name
        6 | Azotirhizobium                                                                   | misspelling
...
```

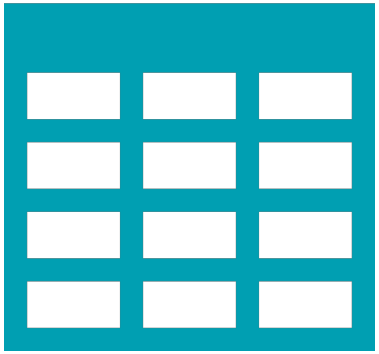
EXAMPLE 4: PG TO… ANYTHING

db1 (PostgreSQL)

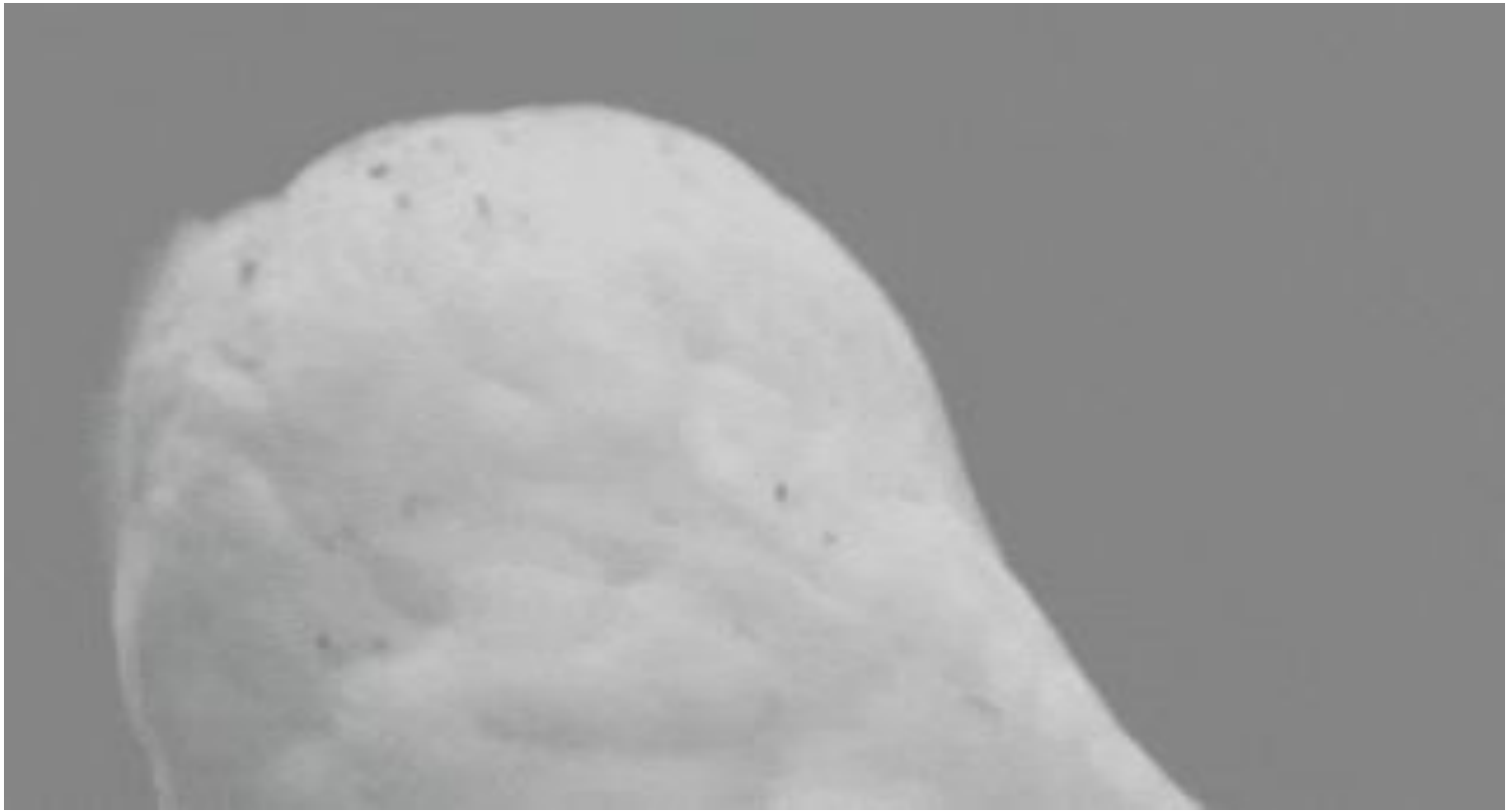


+

Anything that can return tabular data!



EXAMPLE 4: PG TO… ANYTHING



EXAMPLE 4: PG TO… ANYTHING

animal_lookup

File Edit View Insert Format Data Tools Add-ons Help All changes saved in Drive

Python, carpet

	A	B	C
1	id	common_name	scientific_name
2	1	Yellow-necked spurfowl	Francolinus leucoscepus
3	2	Tiger snake	Notechis semmiannulatus
4	3	Hornbill, yellow-billed	Tockus flavirostris
5	4	Malagasy ground boa	Acrantophis madagascariensis
6	5	Tarantula, salmon pink bird eater	Lasiodora parahybana
7	6	Stork, white	Ciconia ciconia
8	7	Scarlet macaw	Ara macao
9	8	Dunnart, fat-tailed	Smithopsis crassicaudata
10	9	Common boubou shrike	Laniarius ferrugineus
11	10	Dusky rattlesnake	Crotalus triseriatus
12	11	Ass, asiatic wild	Equus hemionus
13	12	Gecko, ring-tailed	Cyrtodactylus louisiadensis
14	13	Klipspringer	Oreotragus oreotragus
15	14	Great horned owl	Bubo virginianus
16	15	Cottonmouth	Agkistrodon piscivorus

What if our zoologists maintain this lookup table in a Google spreadsheet?

THE FDW WIKI - TO INFINITY AND BEYOND

File Wrappers

Data Source	Type	License	Code	Install	Doc	Notes
CSV	Native	PostgreSQL	git.postgresql.org		documentation	Delivered as an official extension of PostgreSQL 9.1 / example / Another example
CSV	Multicorn	PostgreSQL	GitHub	PGXN	documentation	Each column defined in the table will be mapped, in order, against columns in the CSV file.
CSV / Text Array	Native		GitHub		How to	Another CSV wrapper
CSV / Fixed-length	Native		GitHub			
CSV / gzipped	Multicorn		GitHub			PostgreSQL Foreign Data Wrapper for gzipped cvs file
Compressed File	Native		GitHub			
Document Collection	Native	PostgreSQL	GitHub		wiki	
JSON	Native	GPL3	GitHub		Example	
Multi-File	Multicorn	PostgreSQL	GitHub	PGXN	doc	Access data stored in various files in a filesystem. The files are looked up based on a pattern, and parts of the file's path are mapped to various columns, as well as the file's content itself.
Multi CDR	Native	PostgreSQL	GitHub	PGXN		
Parquet	Native	PostgreSQL	GitHub			Foreign data wrapper for reading Parquet files using libarrow/libparquet
pg_dump	Native	New BSD	GitHub			Allows querying of data directly against Postgres custom format files created by pg_dump
TAR Files	Native		GitHub			
XML	Multicorn	PostgreSQL	GitHub	PGXN		
ZIP Files	Native		GitHub			

Geo Wrappers

Data Source	Type	License	Code	Install	Doc	Notes
GDAL/OGR	Native	MIT	GitHub			A wrapper for data sources with a GDAL/OGR driver, including databases like Oracle, Informix, SQLite, SQL Server, ODBC as well as file formats like Shape, FGDB, MapInfo, CSV, Excel, OpenOffice, OpenStreetMap PBF and XML, OGC WebServices, and more Spatial columns are linked in as PostGIS geometry if PostGIS is installed.
Geocode / GeoJSON	Multicorn	GPL	GitHub			a collection of PostGIS-related foreign data wrappers

THE FDW WIKI - TO INFINITY AND BEYOND

ICAL	Multicorn	PostgreSQL	GitHub		pdf	
IMAP	Multicorn	PostgreSQL	GitHub	PGXN	documentation	
RSS	Multicorn	PostgreSQL	GitHub	PGXN	documentation	This fdw can be used to access items from an rss feed.
www	Native	PostgreSQL	GitHub	PGXN	wiki	Allows to query different web services

Specific Web Wrappers

Data Source	Type	License	Code	Install	Doc	Notes
Database.com	Multicorn	BSD	GitHub			
Dun & Badstreet	Multicorn	PostgreSQL	GitHub			Access to the Data Universal Numbering System (DUNS)
DynamoDB	Multicorn	GPL	GitHub			
Facebook	Multicorn		GitHub			
Fixer.io	based on www_fdw		GitHub			
Google	Multicorn	PostgreSQL	GitHub	PGXN		
Heroku dataclips	Native	PostgreSQL	GitHub			
Keycloak	Multicorn	MIT	GitHub	PGXN	README	Direct database integration with the Keycloak open-source Identity/Access Management solution.
Mailchimp	Multicorn	PostgreSQL	GitHub			Beta
Parse	Multicorn	MIT	GitHub			
S3	Native	PostgreSQL	GitHub	PGXN		
S3CSV	Multicorn	GPL 3	GitHub			This is meant to replace s3_fdw that does is not supported on PostgreSQL version 9.2+
Telegram	Multicorn	PostgreSQL	GitHub			telegram_fdw is a Telegram BOT implemented using the PostgreSQL foreign data wrapper interface.
Twitter	Native	PostgreSQL	GitHub	PGXN		A wrapper fetching text messages from Twitter over the Internet and returning a table
Treasure Data	Native	Apache	GitHub	PGXN		A FDW for Treasure Data internally using a Rust library
Treasure Data	Multicorn	Apache	GitHub			
Google Spreadsheets	Multicorn	MIT	GitHub			

Why hello there!

EXAMPLE 4: PG TO… ANYTHING

I'll explain this in the next section!

```
db1=# CREATE EXTENSION multicorn;

db1=# CREATE SERVER multicorn_gspreadsheet
db1-# FOREIGN DATA WRAPPER multicorn
db1-# OPTIONS (
db1(#   wrapper 'gspreadsheet_fdw.GspreadsheetFdw'
db1(# );

db1=# CREATE FOREIGN TABLE animal_lookup_gsheet (
db1(#   id INTEGER,
db1(#   common_name TEXT,
db1(#   scientific_name TEXT)
db1-# SERVER multicorn_gspreadsheet
db1-# OPTIONS (
db1(#   gskey '1j3jhy2EWaHbdJ0STKpPc668lhOVW4iwEqYas93TYtBY',
db1(#   keyfile '/Users/sam/code/fdw_test_auth.json'
db1(# );
```

EXAMPLE 4: PG TO… ANYTHING

I just queried a Google sheet from PostgreSQL… woo!

```
db1=# SELECT * FROM animal_lookup_gsheet;
  id  |              common_name              |            scientific_name
------+---------------------------------------+-------------------------------
    1 | Yellow-necked spurfowl                | Francolinus leucoscepus
    2 | Tiger snake                           | Notechis semmiannulatus
    3 | Hornbill, yellow-billed               | Tockus flavirostris
    4 | Malagasy ground boa                   | Acrantophis madagascariensis
    5 | Tarantula, salmon pink bird eater     | Lasiodora parahybana
    6 | Stork, white                          | Ciconia ciconia
    7 | Scarlet macaw                         | Ara macao
    8 | Dunnart, fat-tailed                   | Smithopsis crassicaudata
    9 | Common boubou shrike                  | Laniarius ferrugineus
   10 | Dusky rattlesnake                     | Crotalus triseriatus
   11 | Ass, asiatic wild                     | Equus hemionus
   12 | Gecko, ring-tailed                    | Cyrtodactylus louisiadensis
   13 | Klipspringer                          | Oreotragus oreotragus
   14 | Great horned owl                      | Bubo virginianus
   15 | Cottonmouth                           | Agkistrodon piscivorus
...
```

FDW CONSIDERATIONS*

Benefits (over ETL pipelines)

Requires no additional tooling*, machines, and storage*

Single interface for users, easier to run ad-hoc queries (e.g. for development, QA, spot-checks)

Ability to distribute data over multiple servers

One-step data migrations from other sources

Some concerns

Version changes, upgrades, etc might be more brittle than with ETL tooling

No control over external data sources (but that's a problem either way)

Some security concerns, e.g. mapping to "too powerful" user

What's the data backup strategy?

Performance might be worse depending on use case

* This is a terrible slide, let's discuss later

3 - BYO FDW!

From zero to wrap in five minutes*

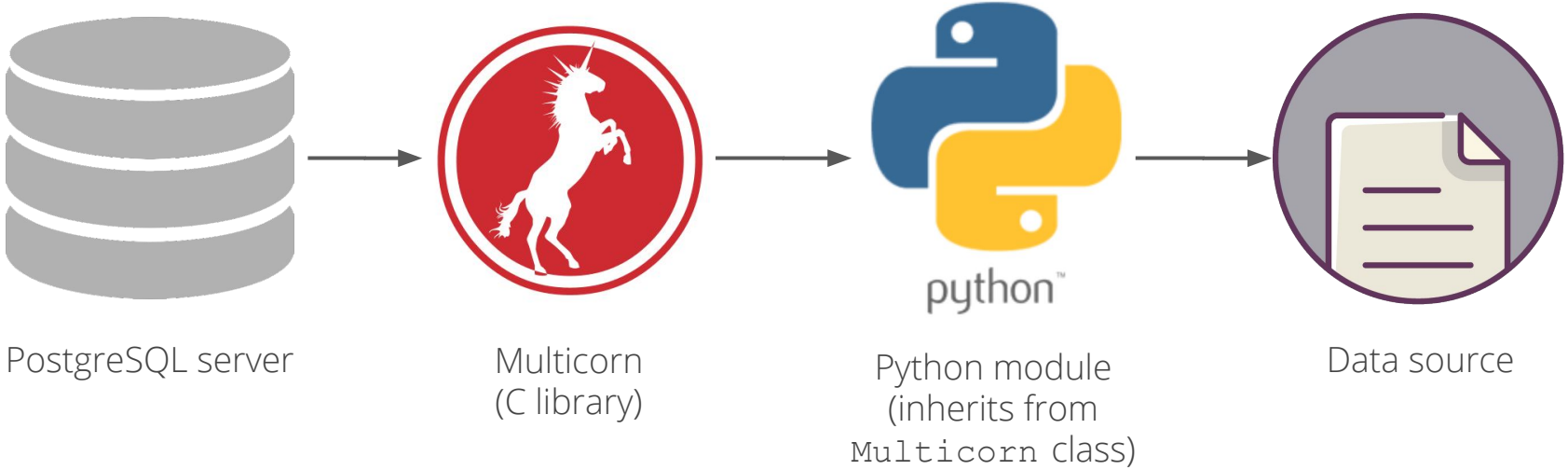
*after I spent several hours getting my old Mac back into shape for development

SIDEBAR: MULTICORN

I'll explain this now!

```
db1=# CREATE EXTENSION multicorn;

db1=# CREATE SERVER multicorn_gspreadsheet
db1-# FOREIGN DATA WRAPPER multicorn
db1-# OPTIONS (
db1(#   wrapper 'gspreadsheet_fdw.GspreadsheetFdw'
db1(# );

db1=# CREATE FOREIGN TABLE animal_lookup_gsheet (
db1(#   id INTEGER,
db1(#   common_name TEXT,
db1(#   scientific_name TEXT)
db1-# SERVER multicorn_gspreadsheet
db1-# OPTIONS (
db1(#   gskey '1j3jhy2EWaHbdJ0STKpPc668lhOVW4iwEqYas93TYtBY',
db1(#   keyfile '/Users/sam/code/fdw_test_auth.json'
db1(# );
```

SIDEBAR: MULTICORN***



***** Caveats**

1) Multicorn does not appear to be under active development. There are known issues with PostgreSQL 11. Check github issues and use with caution. Or help get it back in shape!

2) I haven't figured out how to point PostgreSQL to my virtual Python env, so I'm stuck with MacOS system Python (2.7)

SIDEBAR: MULTICORN

```
db1=# CREATE EXTENSION multicorn;

db1=# CREATE SERVER multicorn_gspreadsheet
db1-# FOREIGN DATA WRAPPER multicorn
db1-# OPTIONS (
db1(#   wrapper 'gspreadsheet_fdw.GspreadsheetFdw'
db1(# );

db1=# CREATE FOREIGN TABLE animal_lookup_gsheet (
db1(#   id INTEGER,
db1(#   common_name TEXT,
db1(#   scientific_name TEXT)
db1-# SERVER multicorn_gspreadsheet
db1-# OPTIONS (
db1(#   gskey '1j3jhy2EWaHbdJ0STKpPc668lhOVW4iwEqYas93TYtBY',
db1(#   keyfile '/Users/sam/code/fdw_test_auth.json'
db1(# );
```

This is where we specify the Python module

Options depend on the wrapper

A MULTICORN MODULE UNDER THE HOOD

lincolnturner / gspreadsheet_fdw

Watch 4 | Star 34 | Fork 4

Code | Issues 2 | Pull requests 0 | Projects 0 | Wiki | Security | Insights

Multicorn-based PostgreSQL foreign data wrapper for Google Spreadsheets

13 commits | 2 branches | 0 releases | 1 contributor | MIT

Branch: master | New pull request | Create new file | Upload files | Find File | Clone or download

lincolnturner Added files for type management, imported from pypq project — Latest commit a691733 on Jul 6, 2016

gspreadsheet_fdw	Silly type casting bug fixed	3 years ago
LICENSE	Initial commit	5 years ago
README.md	Seeing which branch this will push to	5 years ago
buildtypes.py	Added files for type management, imported from pypq project	3 years ago
datatypes.py	Added files for type management, imported from pypq project	3 years ago
errors.py	Added files for type management, imported from pypq project	3 years ago
setup.py	First released version for github	5 years ago
test.sql	First draft of documentation	5 years ago

A MULTICORN MODULE UNDER THE HOOD

Branch: master ▾ gspreadsheet_fdw / setup.py

Component 1: a setup file

Find file | Copy path

ldturner First released version for github c46940e on Feb 24, 2015

1 contributor

14 lines (11 sloc) | 310 Bytes

Raw | Blame | History

```
import subprocess
from setuptools import setup, find_packages, Extension

setup(
  name='gspreadsheet_fdw',
  version='0.0.1',
  author='Lincoln Turner',
  author_email='lincoln.turner@monash.edu',
  url='https://github.com/lincolnturner/gspreadsheet_fdw/',
  license='MIT',
  packages=['gspreadsheet_fdw']
)

```

A MULTICORN MODULE UNDER THE HOOD

Branch: master ▾ gspreadsheet_fdw / gspreadsheet_fdw / __init__.py Find file Copy path

lincolnturner Silly type casting bug fixed e4a350e on Jun 29, 2016

2 contributors

33 lines (28 sloc) | 1.75 KB Raw Blame History

```
1
2  from multicorn import ForeignDataWrapper
3  from multicorn.utils import log_to_postgres, ERROR, WARNING, DEBUG
4  from oauth2client.service_account import ServiceAccountCredentials
5  import gspread
6
7  class GspreadsheetFdw(ForeignDataWrapper):
8      def __init__(self, fdw_options, fdw_columns):
9          """A Google Spreadsheets Foreign Wrapper.
```

Component 2: a class that inherits from multicorn's **ForeignDataWrapper**

●●●

```
30     def execute(self, quals, columns):
31         return self.wks.get_all_records(head=int(self.headrow));
32
```

Component 3: implement "**execute**" method (return rows from data source)

** Optional: implement insert/update/delete

BYO FDW

BYO BORING FDW

```
Default (ruby)
copying sam_fdw.egg-info/top_level.txt -> build/bdist.macosx-10.14-intel/egg/EG
G-INFO
zip_safe flag not set; analyzing archive contents...
creating dist
creating 'dist/sam_fdw-0.0.1-py2.7.egg' and adding 'build/bdist.macosx-10.14-in
tel/egg' to it
removing 'build/bdist.macosx-10.14-intel/egg' (and everything under it)
Processing sam_fdw-0.0.1-py2.7.egg
Copying sam_fdw-0.0.1-py2.7.egg to /Library/Python/2.7/site-packages
Adding sam-fdw 0.0.1 to easy-install.pth file

Installed /Library/Python/2.7/site-packages/sam_fdw-0.0.1-py2.7.egg
Processing dependencies for sam-fdw==0.0.1
Finished processing dependencies for sam-fdw==0.0.1
DeLorean:sam_fdw sam$ brew services restart postgresql@10
Stopping `postgresql@10`... (might take a while)
==> Successfully stopped `postgresql@10` (label: homebrew.mxcl.postgresql@10)
==> Successfully started `postgresql@10` (label: homebrew.mxcl.postgresql@10)
DeLorean:sam_fdw sam$
```

4 - WRAP-UP

See what I did there?

QUICK SUMMARY

1

Why and what?

Cross-Postgres DB querying with dblink and FDW

postgresql_fdw under the hood

2

FDW in the wild

Cross-anything querying with FDW

Where to find existing FDW

FDW considerations

3

BYO FDW!

Multicorn overview

Caveats

End-to-end FDW building with Multicorn

THANK YOU!

Sam Bail @spbail
Slides will be on my github

Q&A

1. Have you (successfully) used FDW?
2. What are your anti-examples for FDW?
3. Do you have any questions for me?

REFERENCES

- https://wiki.postgresql.org/wiki/Foreign_data_wrappers
- https://www.postgresql.org/docs/9.6/dblink.html
- https://www.postgresql.org/docs/9.6/postgres-fdw.html
- EBI databases:
  - https://rnacentral.org/help/public-database
  - http://ensemblgenomes.org/info/access/mysql
- https://github.com/Kozea/Multicorn
- https://multicorn.org/implementing-a-fdw/

- **Thanks for guidance and feedback:** Renee Phillips, Jonathan Katz, James Dura