WhereGroup

FOSS4G BUCHAREST 2019
26-31 August 44.43555, 26.102347

Fun with Foreign Data Wrappers

Get access to other Sources via FDW

Astrid Emde, WhereGroup

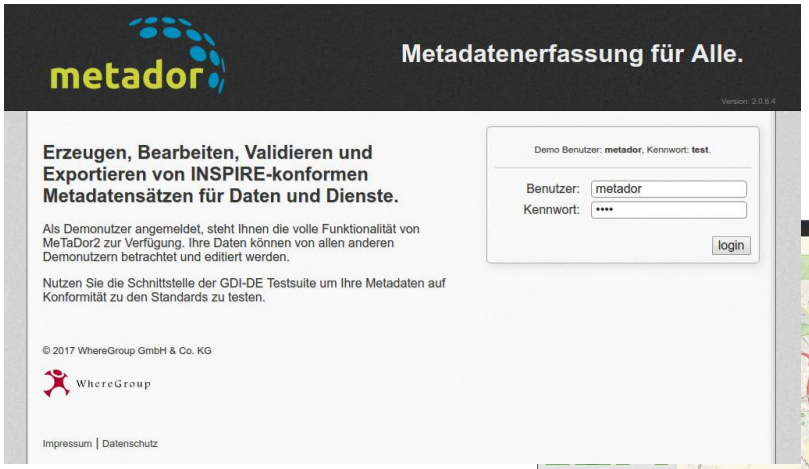
FOSS4G 2019 Bucharest (Romania)

Astrid Emde

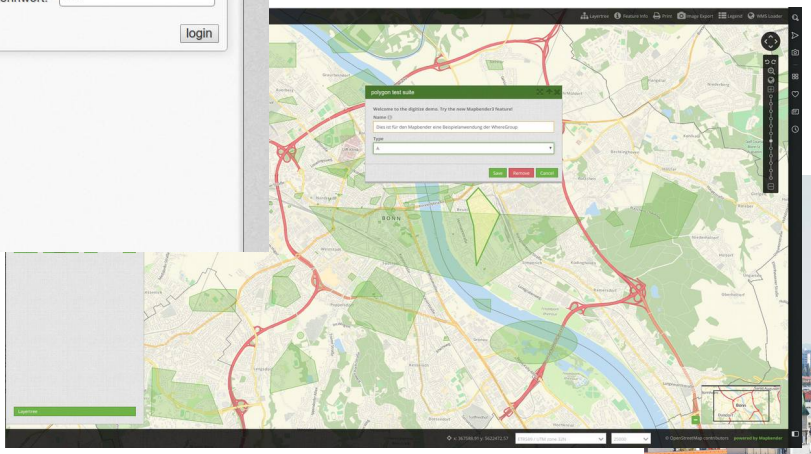
- **Mapbender Team & PSC member**
- WhereGroup **Bonn (Germany)**
- **Project management and less development, trainings for MapServer, PostgreSQL/PostGIS, Mapbender, GeoServer**
- **Mapbender Concept, Testing, Documentation**
- **OSGeoLive PSC since 2017**
- **OSGeo Board since 2017**

WhereGroup

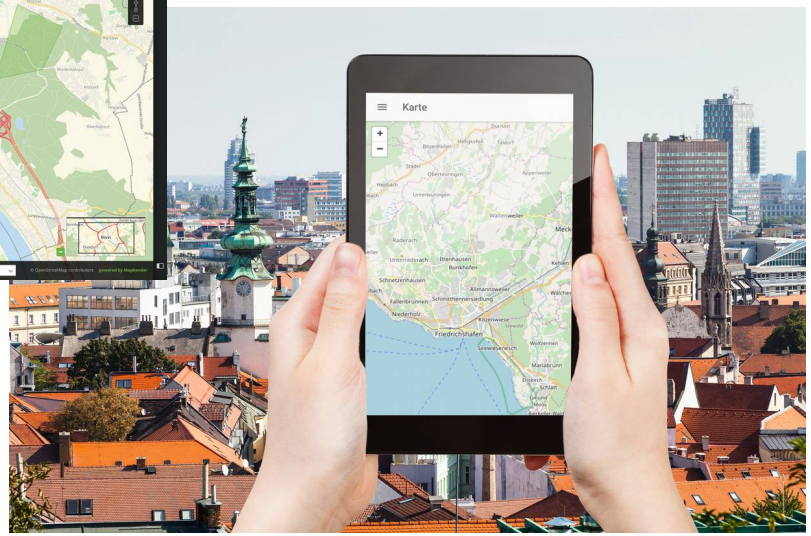
Metador



Mapbender



Mops



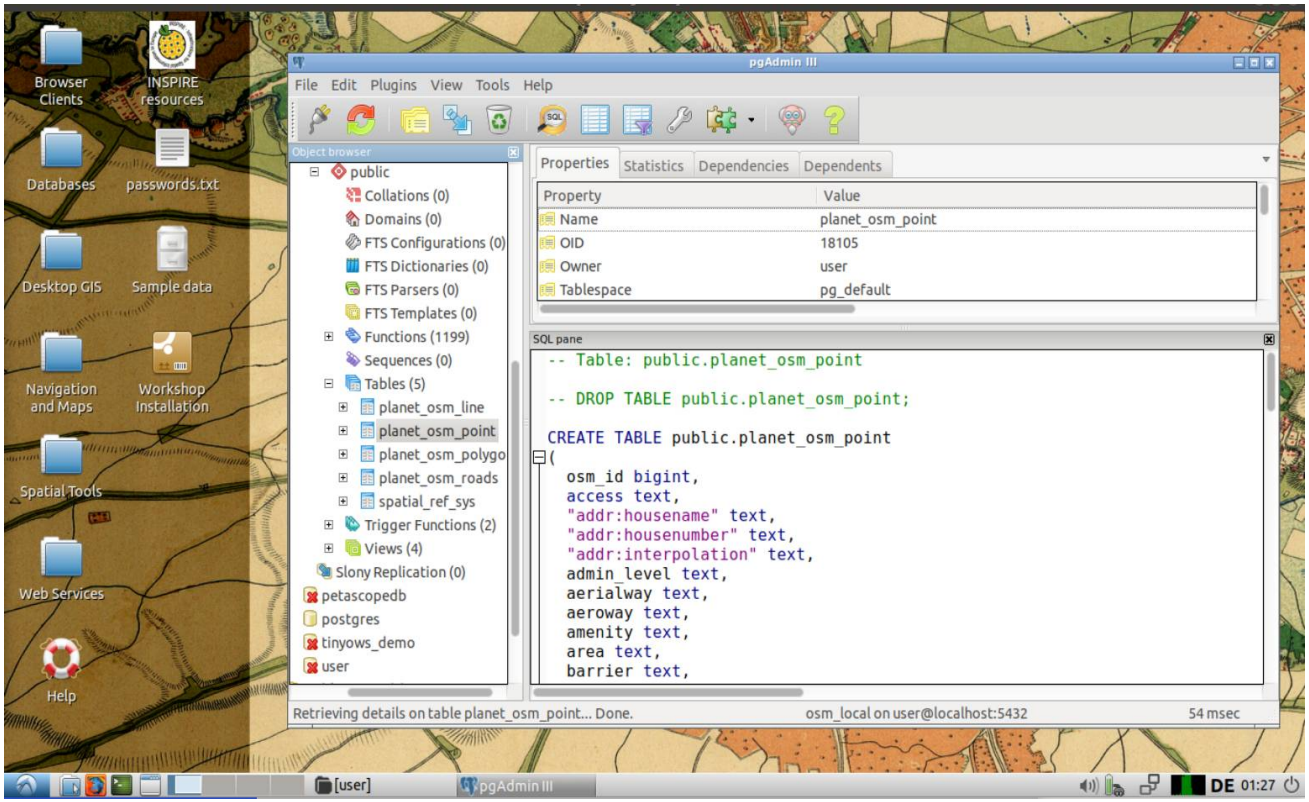
FOSS4G Community Sprint 2019

- **Don‘t miss it!**
- **Saturday 31.8.2019**
- **https://wiki.osgeo.org/wiki/FOSS4G_2019_Community_Sprint**

Try the presentation with OSGeoLive

- **Get pgAdmin 4**

  sudo apt-get install pgadmin4



What are Foreign Data Wrappers - FDW

- & why is it fun to work with them?

Why FDW?

- … but we already have dblink

DBLINK

- **... but we already have dblink**

```
  CREATE EXTENSION dblink;

  SELECT *
  FROM dblink('dbname=osm_local',

'SELECT osm_id, name, way as geom

FROM public.planet_osm_point

WHERE amenity = ''cafe''')
 AS foo
 ( osm_id int8, name text, geom geometry(point,4326));
```

SQL/MED Standard

- Based on the SQL/MED Standard
- SQL Management of External Data
- SQL Standard from by ISO/IEC 9075-9:2008
- Defines how a database can connect to external datasources
- https://wiki.postgresql.org/wiki/SQL/MED

FDW in general

- FDW in general
- not many implementations
- MariaDB CONNECT - but not following the standard
- IBM/DB2

FDW in PostgreSQL

- added in 2011 with read-only support (9.1)
- 2013 write support (9.3)

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

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		
Cassandra2	Native	MIT	Github			
Cassandra	Multicorn	PostgreSQL	Github			
ClickHouse	Multicorn	BSD	Github		README	
ClickHouse	Native	Apache	Github		README	
CouchDB	Native	PostgreSQL	Github	PGXN		Original version
CouchDB	Native	PostgreSQL	Github			golgauth version (9.1 - 9.2+ compatible)
GridDB	Native	PostgreSQL	Github		README	
InfluxDB	Native	PostgreSQL	Github		README	
Kafka	Native	PostgreSQL	GitHub		README	
Kyoto Tycoon	Native	MIT	Github			

PostgreSQL FDW

- many other sources are supported
- other SQL databases
- flat files
- geospatial data sources
- Twitter
- https://wiki.postgresql.org/wiki/Foreign_data_wrappers

Different installation, some with PGEX PostgreSQL Extension Network

Connect PostgreSQL to to PostgreSQL Database

Connect natural_earth2 with osm_local

- Load Extension postgres_fdw
- Create a foreign Server
- Create a foreign User
- Create a foreign Table
- Have Fun!

Load Extension

```
CREATE EXTENSION postgres_fdw;
```

Foreign Server

```
CREATE SERVER fdw_pg_server_osm_local
FOREIGN DATA WRAPPER postgres_fdw
OPTIONS (host '127.0.0.1', port '5432', dbname 'osm_local');
```

User Mapping

```
CREATE USER MAPPING FOR user
SERVER fdw_pg_server_osm_local
OPTIONS (user 'user', password 'user');
```

CREATE FOREIGN TABLE

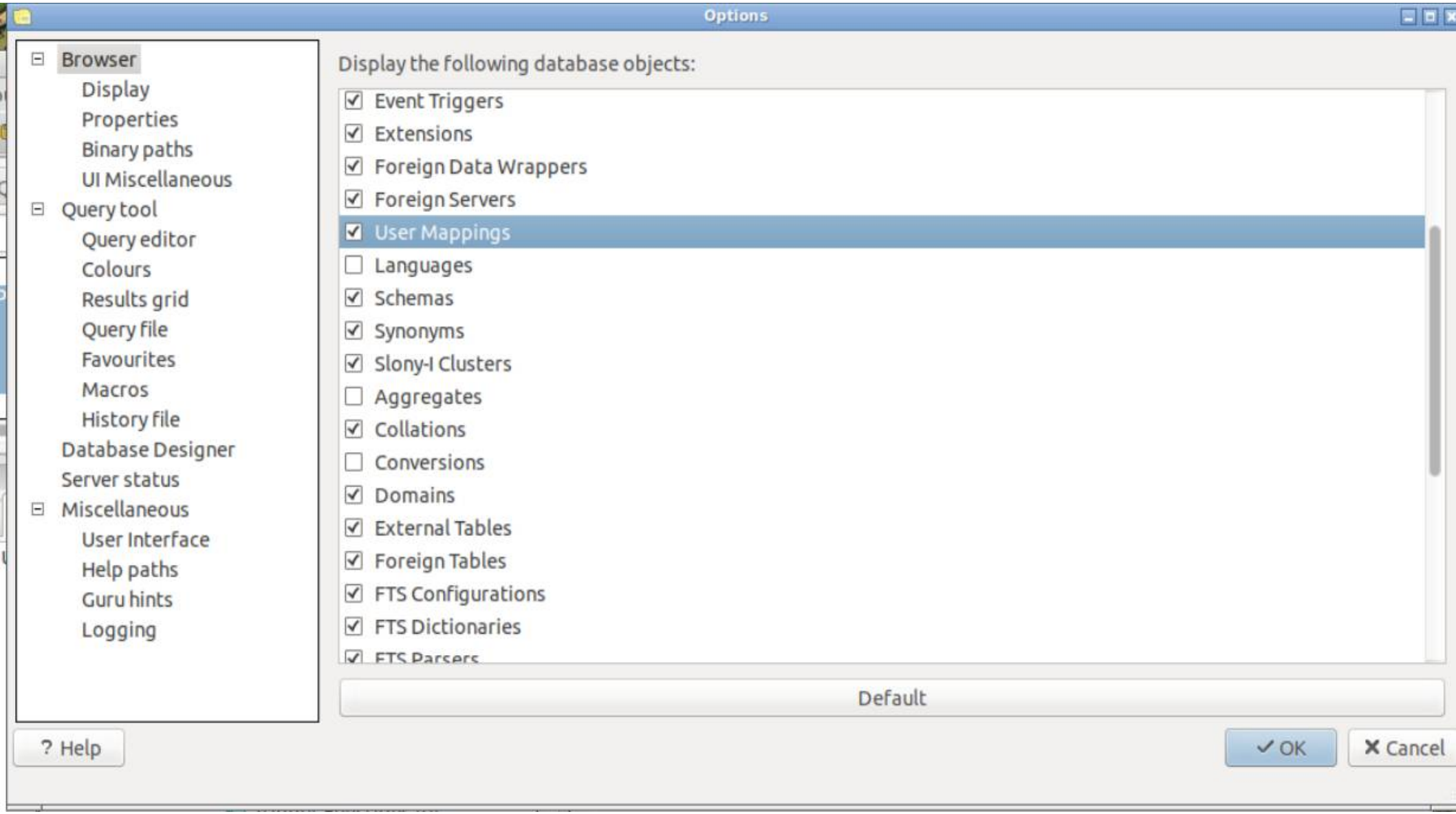
```
Create schema osm_fdw_pg;

IMPORT FOREIGN SCHEMA public
    LIMIT TO (planet_osm_polygon, planet_osm_point)
    FROM SERVER fdw_pg_server_osm_local
    INTO osm_fdw_pg;
```

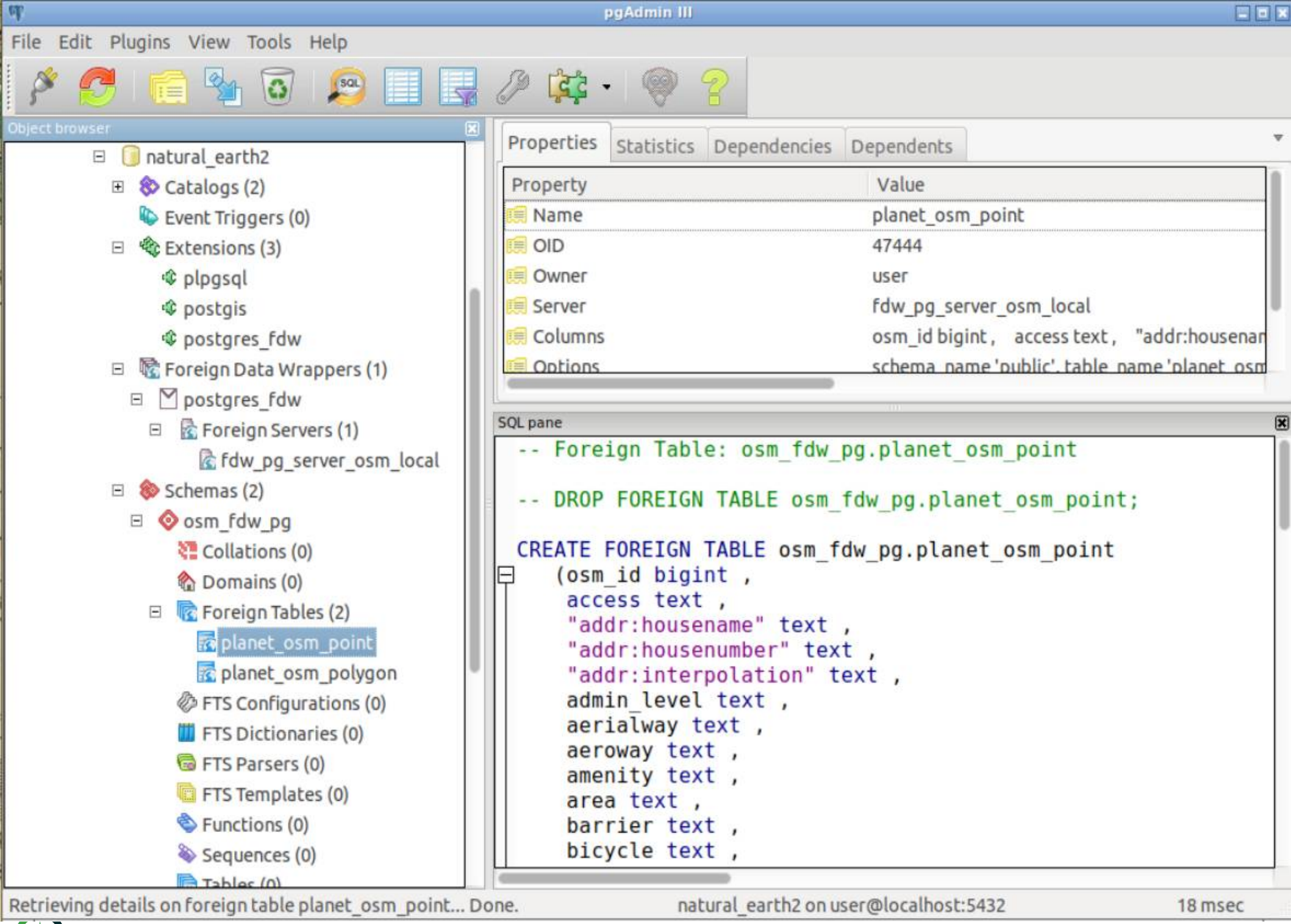
LIMIT TO (tablename, tablename)

EXCEPT (tablename, tablename)

Combine tables as if there are in your database

```
SELECT count(*)
  FROM
    ne_10m_admin_1_states_provinces_shp p
    osm_fdw_pg.planet_osm_point o,
  WHERE
    ST_Distance(o.way, p.the_geom) = 0
    AND amenity = 'cafe'
    AND p.name = 'Bucharest';

  count
  ---------
  174
```

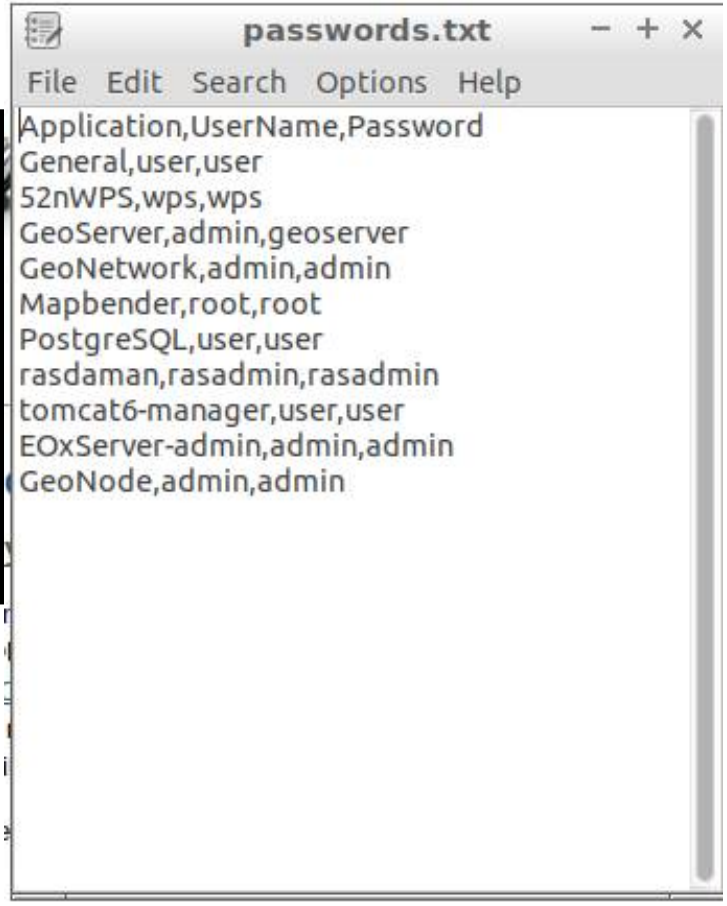

Options
Browser
Display
Properties
Binary paths
UI Miscellaneous
Query tool
Query editor
Colours
Results grid
Query file
Favourites
Macros
History file
Database Designer
Server status
Miscellaneous
User Interface
Help paths
Guru hints
Logging
Display the following database objects:
Event Triggers
Extensions
Foreign Data Wrappers
Foreign Servers
User Mappings
Languages
Schemas
Synonyms
Slony-I Clusters
Aggregates
Collations
Conversions
Domains
External Tables
Foreign Tables
FTS Configurations
FTS Dictionaries
FTS Parsers
Default
? Help
OK
Cancel


pgAdmin III
File Edit Plugins View Tools Help
Object browser
natural_earth2
Catalogs (2)
Event Triggers (0)
Extensions (3)
plpgsql
postgis
postgres_fdw
Foreign Data Wrappers (1)
postgres_fdw
Foreign Servers (1)
fdw_pg_server_osm_local
Schemas (2)
osm_fdw_pg
Collations (0)
Domains (0)
Foreign Tables (2)
planet_osm_point
planet_osm_polygon
FTS Configurations (0)
FTS Dictionaries (0)
FTS Parsers (0)
FTS Templates (0)
Functions (0)
Sequences (0)
Properties
Statistics
Dependencies
Dependents
Property
Value
Name
planet_osm_point
OID
47444
Owner
user
Server
fdw_pg_server_osm_local
Columns
osm_id bigint , access text , "addr:housenar
Options
schema_name 'public', table_name 'planet_osm
SQL pane
-- Foreign Table: osm_fdw_pg.planet_osm_point
-- DROP FOREIGN TABLE osm_fdw_pg.planet_osm_point;
CREATE FOREIGN TABLE osm_fdw_pg.planet_osm_point
(osm_id bigint ,
access text ,
"addr:housename" text ,
"addr:housenumber" text ,
"addr:interpolation" text ,
admin_level text ,
aerialway text ,
aeroway text ,
amenity text ,
area text ,
barrier text ,
bicycle text ,
Retrieving details on foreign table planet_osm_point... Done.
natural_earth2 on user@localhost:5432
18 msec

file_fdw

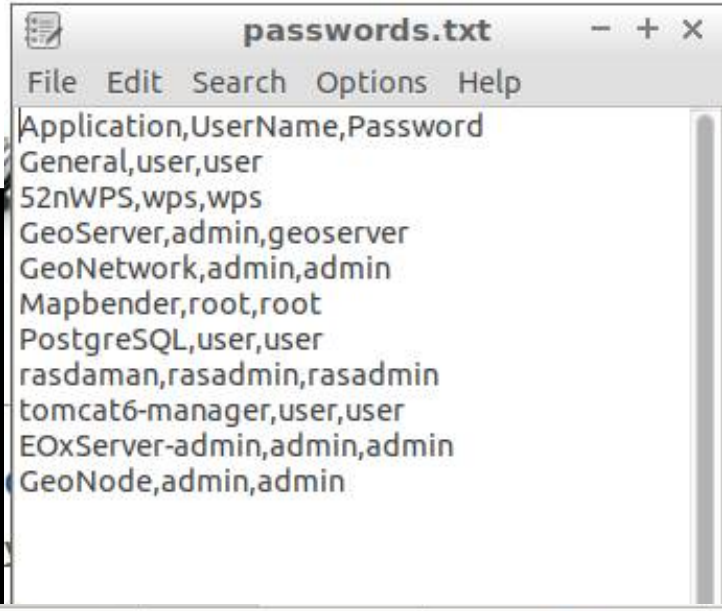
- **https://www.postgresql.org/docs/current/file-fdw.html**

```
CREATE EXTENSION file_fdw;

/home/user/Desktop/passwort.txt

CREATE SERVER pglog
FOREIGN DATA WRAPPER file_fdw;
```



file_fdw

```
CREATE FOREIGN TABLE passwords (
  fid bigint,
  application varchar,
  username varchar,
  password varchar
) SERVER myserver_csv
OPTIONS (layer 'passwords');

Select * from passwords;
```



	fid bigint	application character varying	username character varying	password character varying
1	1	General	user	user
2	2	52nWPS	wps	wps
3	3	GeoServer	admin	geoserver
4	4	GeoNetwork	admin	admin
5	5	Mapbender	root	root
6	6	PostgreSQL	user	user
7	7	rasdaman	rasadmin	rasadmin
8	8	tomcat6-manager	user	user
9	9	EOxServer-admin	admin	admin
10	10	GeoNode	admin	admin

Connect from PostgreSQL to ORACLE Database

- Load Extension oracle_fdw
- Create a foreign Server
- Create a foreign User
- Create a foreign Table
- Have Fun!

oracle_fdw

```
CREATE EXTENSION oracle_fdw;
```

```
CREATE EXTENSION oracle_fdw;
CREATE SERVER oradb
    FOREIGN DATA WRAPPER oracle_fdw
    OPTIONS (dbserver '//dbserver.mydomain.com:1521/ORADB');
GRANT USAGE ON FOREIGN SERVER oradb TO pguser;
```

```
CREATE USER MAPPING FOR pguser SERVER oradb
   OPTIONS (user 'orauser', password 'orapwd');

CREATE FOREIGN TABLE oratab (
        id       integer          OPTIONS (key 'true')  NOT NULL,
        text      character varying(30),
        floating  double precision  NOT NULL
    ) SERVER oradb OPTIONS (schema 'ORAUSER', table 'ORATAB');
```

oracle_fdw

- Combine tables as if there are in your database
- Pushdown of WHERE Clause
- Pushdown of requierd columns
- EXPLAIN Support

OGR FDW

- FDW by Paul Ramsey
- Many formats like Geopackage, WFS, OSM, ESRI Shape, KML, …
- Even WFS 3.0

How to install OGR FDW?

- Download & compile ogr_fdw on yourself
- or use OSGeoLive 13.0
- packages available
- https://packages.ubuntu.com/source/bionic/pgsql-ogr-fdw
- https://packages.debian.org/sid/postgresql-10-ogr-fdw

```
sudo apt-get install postgresql-10-ogr-fdw
```

OGR FDW Formats

- **See availabe formats**

```
/usr/lib/postgresql/10/bin/ogr_fdw_info -f

Supported Formats:
-> "OGR_GRASS" (readonly)
-> "PCIDSK" (read/write)
-> "netCDF" (read/write)
-> "JP2OpenJPEG" (readonly)
-> "PDF" (read/write)
-> "MBTiles" (read/write)
-> "EEDA" (readonly)
-> "ESRI Shapefile" (read/write)
-> "MapInfo File" (read/write)
-> "UK .NTF" (readonly)
-> "OGR_SDTS" (readonly)
-> "S57" (read/write)
```

OGR FDW and ESRI Shape

Connect natural_earth2 ESRI SHP

- Load Extension ogr_fdw
- Create a foreign Server
- ~~Create a foreign User~~
- Create a foreign Table
- Have Fun!

Load Extension

- **CREATE EXTENSION ogr_fdw;**

Have a look at your data

```
cd /usr/lib/postgresql/10/bin/
./ogr_fdw_info -s /home/user/data/natural_earth2/
        Layers:
        ne_10m_geography_marine_polys
        ne_10m_geography_regions_points
        ne_10m_urban_areas
        ne_10m_populated_places
        ne_10m_admin_0_countries
        ne_10m_geography_regions_polys
        ne_10m_admin_1_states_provinces_shp
        ne_10m_geography_regions_elevation_points
        ne_10m_lakes
        ne_10m_ocean
        ne_10m_rivers_lake_centerlines
        ne_10m_land
```

Create Foreign Server

```
CREATE SERVER myserver
  FOREIGN DATA WRAPPER ogr_fdw
  OPTIONS (
    datasource '/home/user/data/natural_earth2/',
    format 'ESRI Shapefile' );
```

Create Foreign Table

```
CREATE FOREIGN TABLE ne_10m_populated_places
(
  fid bigint,
  geom Geometry(Point,4326),
  scalerank integer,
  natscale integer,
  labelrank integer,
  featurecla varchar,
  name varchar,
  namepar varchar,
  namealt varchar,
  diffascii integer,
.....
  pop2010 real,
  pop2015 real,
  pop2020 real,
  pop2025 real,
  pop2050 real,
  cityalt varchar
) SERVER myserver
OPTIONS (layer 'ne_10m_populated_places');
```

Encod?ng

- **'utf-8' codec can't decode byte 0xed in position 1: invalid continuation byte**

```
CREATE SERVER myserver_latin1
  FOREIGN DATA WRAPPER ogr_fdw
  OPTIONS (
    datasource
'/home/user/data/natural_earth2/',
format 'ESRI Shapefile',
config_options
'SHAPE_ENCODING=LATIN1');


Set client_encoding to UNICODE;
```

OGR FDW und OSM

- **Examine your data**

```
/usr/lib/postgresql/10/bin/ogr_fdw_info -s
/home/user/feature_city.osm
Layers:
  points
  lines
  multilinestrings
  multipolygons
  other_relations
```

ogr_fdw_info writes SQL for you

```
/usr/lib/postgresql/10/bin/ogr_fdw_info -s /home/user/feature_city.osm -l points

CREATE SERVER myserver_osm
  FOREIGN DATA WRAPPER ogr_fdw
  OPTIONS (
    datasource '/home/user/feature_city.osm',
    format 'OSM' );

CREATE FOREIGN TABLE points (
  fid bigint,
  geom Geometry(Point,4326),
  osm_id varchar, name varchar,
  barrier varchar,  highway varchar,
  ref varchar,  address varchar,
  is_in varchar,  place varchar,
  man_made varchar,
  other_tags varchar
) SERVER myserver_osm
OPTIONS (layer 'points');
```

ogr_fdw_info writes SQL for you

```
/usr/lib/postgresql/10/bin/ogr_fdw_info -s /home/user/feature_city.osm -l points

CREATE SERVER myserver_osm
  FOREIGN DATA WRAPPER ogr_fdw
  OPTIONS (
    datasource '/home/user/feature_city.osm',
    format 'OSM' );

CREATE FOREIGN TABLE points (
  fid bigint,
  geom Geometry(Point,4326),
  osm_id varchar, name varchar,
  barrier varchar,  highway varchar,
  ref varchar,  address varchar,
  is_in varchar,  place varchar,
  man_made varchar,
  other_tags varchar
) SERVER myserver_osm
OPTIONS (layer 'points');
```

FDW and EXPLAIN ANALYZE

```
Total points: 77003


Select count(*) from points where highway = 'traffic_signals';
-------
600

EXPLAIN ANALYZE
Select count(*) from points where highway = 'traffic_signals';

"Aggregate  (cost=1027.50..1027.51 rows=1 width=8) (actual
time=2586.355..2586.355 rows=1 loops=1)"
"  ->  Foreign Scan on points  (cost=25.00..1025.00 rows=1000 width=0) (actual
time=3.047..2586.076 rows=600 loops=1)"
"        Filter: ((highway)::text = 'traffic_signals'::text)"
"Planning time: 34.801 ms"
"Execution time: 2637.603 ms"
```

OGR FDW netCDF Support

```
/usr/lib/postgresql/10/bin/ogr_fdw_info -s
/home/user/data/netcdf/rx5dayETCCDI_yr_MIROC5_historical_r2i1p1_1850-
2012.nc
Layers:
  rx5dayETCCDI_yr_MIROC5_historical_r2i1p1_1850-2012

CREATE SERVER myserver_netcdf
  FOREIGN DATA WRAPPER ogr_fdw
  OPTIONS (
    datasource
'/home/user/data/netcdf/rx5dayETCCDI_yr_MIROC5_historical_r2i1p1_1850-
2012.nc',
    format 'netCDF' );

CREATE FOREIGN TABLE rx5dayetccdi_yr_miroc5_historical_r2i1p1_1850_2012
(
  fid bigint,
  time real
) SERVER myserver_netcdf
OPTIONS (layer 'rx5dayETCCDI_yr_MIROC5_historical_r2i1p1_1850-2012');

Select * from rx5dayetccdi_yr_miroc5_historical_r2i1p1_1850_2012;
```

OGR FDW and WFS Support

- **WFS is supported**
- **Readonly**

```
Drop server myserver_wfs_qgis_server Cascade;

CREATE SERVER myserver_wfs_qgis_server
  FOREIGN DATA WRAPPER ogr_fdw
  OPTIONS (
    datasource
'WFS:http://localhost/cgi-bin/qgis_mapserv.fcgi?map=/ho
me/user/world.qgz',
    format 'WFS',
    config_options 'CPL_DEBUG=ON');
```

OGR FDW and WFS

```
Create schema fdw_wfs_qgis_server;

IMPORT FOREIGN SCHEMA ogr_all
FROM server myserver_wfs_qgis_server
INTO fdw_wfs_qgis_server;

Show client_min_messages;
SET client_min_messages=debug2;

Select * from
fdw_wfs_qgis_server.ne_10m_admin_0_countries;
```

Thank you

Astrid Emde

astrid.emde@wheregroup.com