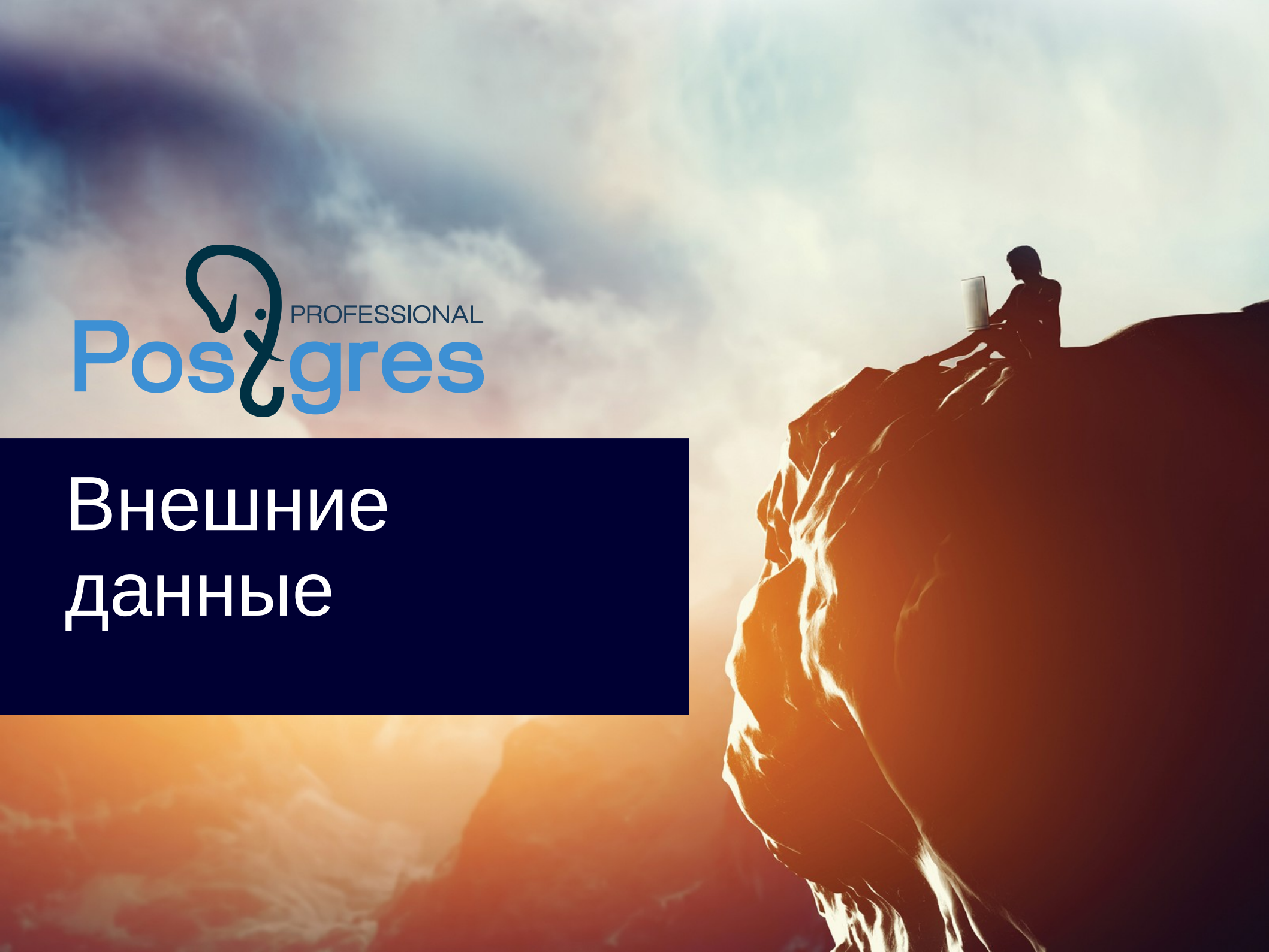
PROFESSIONAL Postgres

Внешние данные



Темы

Назначение

Настройка доступа к внешним данным

file_fdw

postgres_fdw

Доступные FDW

Назначение

Доступ к внешним данным

- источник данных вне текущей БД
- возможно как чтение, так и запись

Прозрачно для приложения

- внешние данные представлены в виде таблиц БД
- доступ через GRANT/REVOKE
- команды DML, триггеры

Стандарт ISO/IEC 9075-9 (SQL/MED)

Настройка доступа

Для получения доступа к внешним данным нужно создать несколько объектов:

1. Обертка сторонних данных
2. Внешний сервер
3. Сопоставление пользователей
4. Внешняя таблица

Обертка сторонних данных

Назначение

API для работы с внешним источником

Создание

CREATE FOREIGN DATA WRAPPER

CREATE EXTENSION

только суперпользователь

Информация

pg_foreign_data_wrapper \dew

Внешний сервер

Назначение

содержит информацию о подключении к внешнем источнику

Создание

CREATE SERVER

на один FDW может быть несколько серверов

например, для каждой БД, к которой нужно подключиться

указываются специфичные для сервера опции подключения

например, для postgres_fdw: OPTIONS (dbname 'postgres' host 'remotehost' port '5432')

привилегия USAGE на обертку сторонних данных

Информация

pg_foreign_server \des

Сопоставление ролей

Назначение

настройка соответствия между локальными ролями
и учетными записями на внешнем сервере

Создание

CREATE USER MAPPING

указываются дополнительные опции подключения к внешнему серверу, такие, как имя пользователя, пароль

например, для postgres_fdw: OPTIONS (user 'postgres' password 'qwerty')

владелец сервера, роль с привилегией USAGE на сервер

Информация

pg_user_mappings \deu

Внешние таблицы

Назначение

доступ к данным на внешнем сервере

Создание

CREATE FOREIGN TABLE

IMPORT FOREIGN SCHEMA

участие в наследовании

включение ограничений: NOT NULL, CHECK

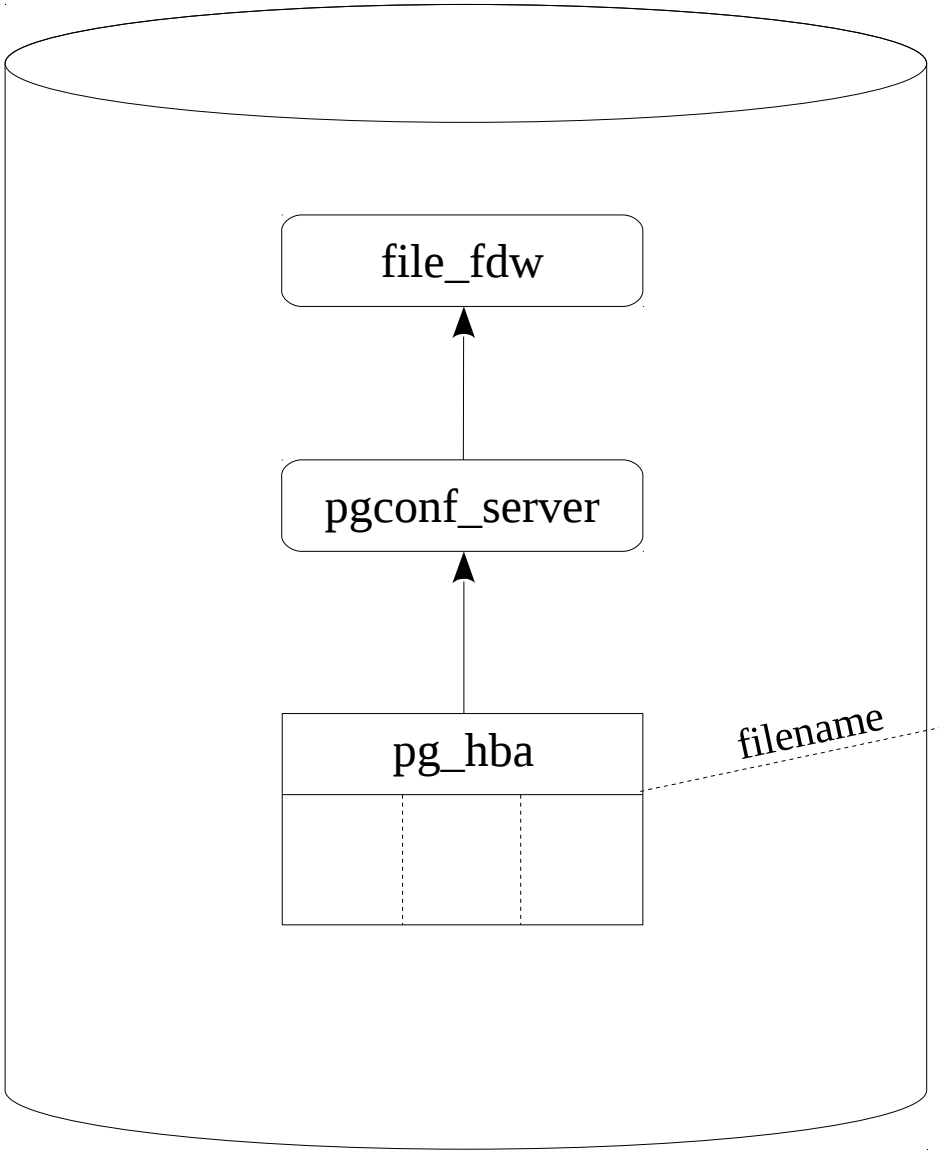
- декларативные, могут использоваться планировщиком

владелец сервера, роль с привилегией USAGE на сервер

Информация

pg_foreign_table pg_class where relkind = 'f' \det

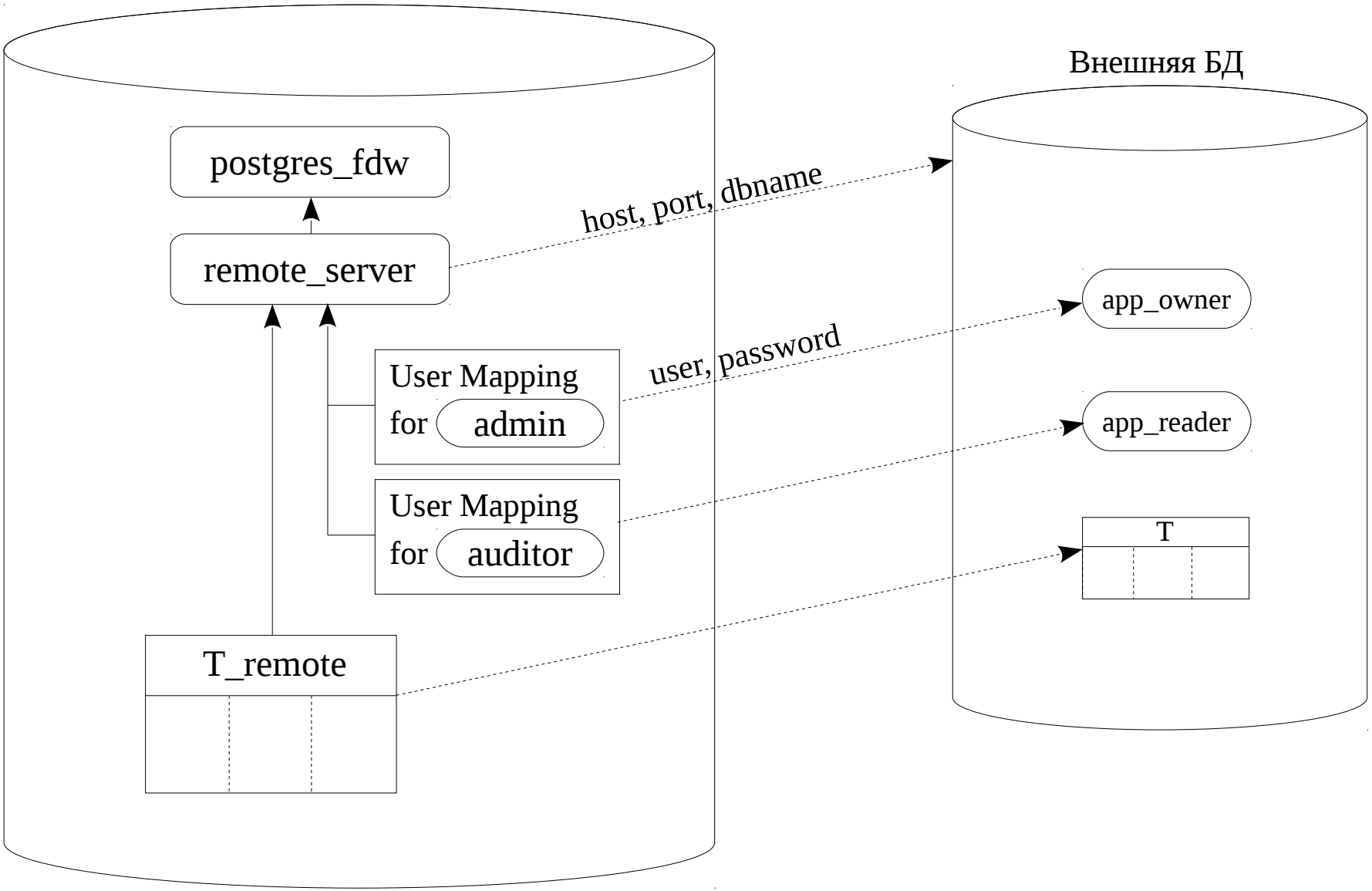
file_fdw




```
$ pwd
/usr/local/pgsql/data

$ ls *.conf
pg_hba.conf
pg_ident.conf
postgresql.auto.conf
postgresql.conf
```

postgres_fw


Внешняя БД
postgres_fdw
remote_server
host, port, dbname
User Mapping
for
admin
user, password
User Mapping
for
auditor
T_remote
app_owner
app_reader
T

Доступные FDW

Поставляются вместе с PostgreSQL (модули contrib)

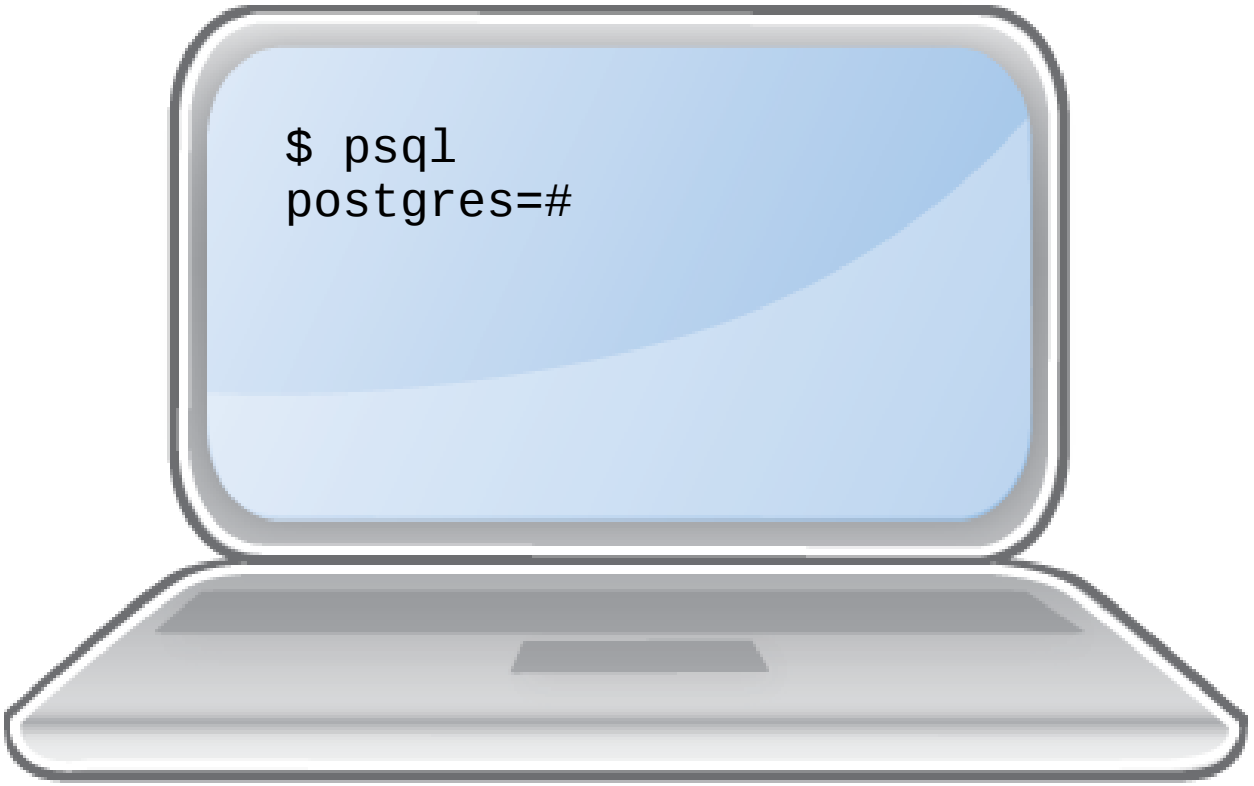
- file_fdw
- postgres_fdw

Внешние источники

- страница на wiki.postgresql.org
- PostgreSQL Extension Network

… написать свою

Демонстрация



Итоги

Работа с внешними данными регламентируется стандартом ISO/IEC 9075-9 (SQL/MED)

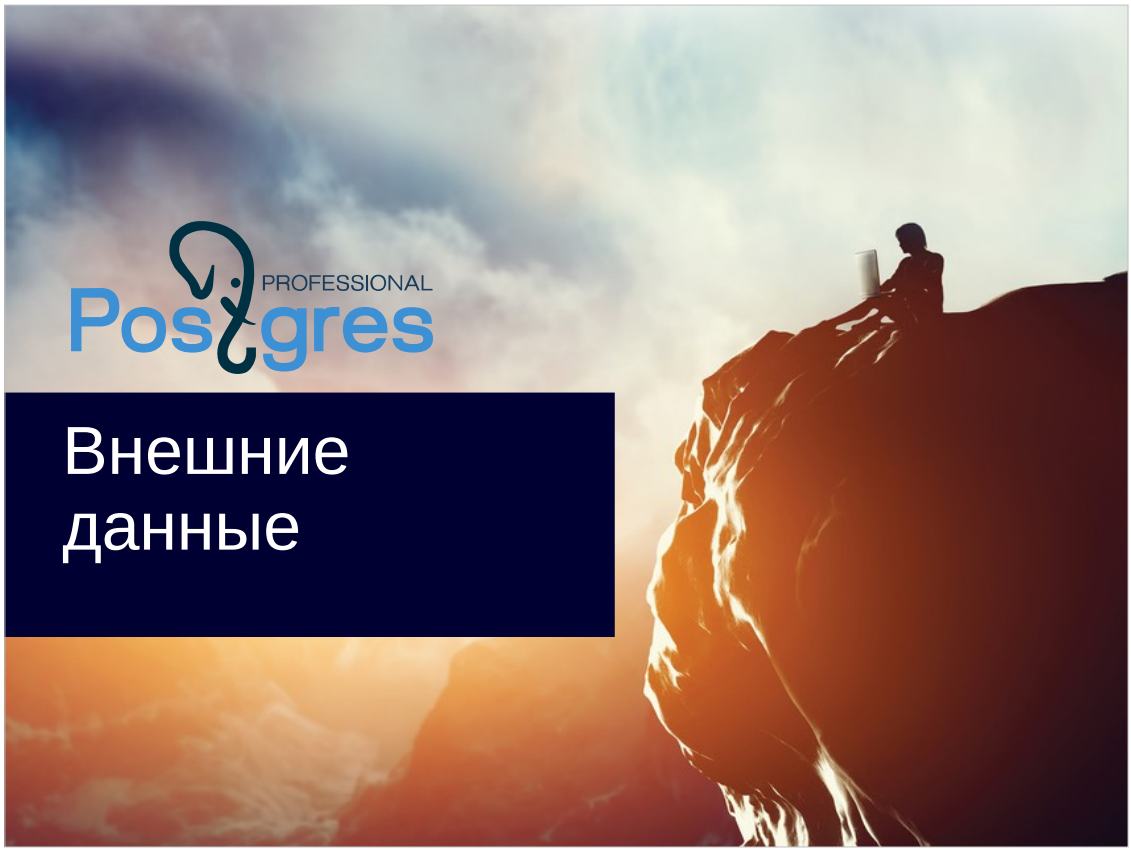
В общем случае, настройка доступа к внешним данным заключается в определении объектов БД:

- обертки сторонних данных
- внешние сервера
- сопоставления пользователей
- внешние таблицы

Вместе с PostgreSQL поставляются два модуля: file_fdw и postgres_fdw

Практика

1. file_fdw
   1.1. Сделать в базе данных *db25* внешнюю таблицу *pg_hba* для просмотра конфигурационного файла pg_hba.conf.
   1.2. Проверить содержимое таблицы *pg_hba*
2. postgres_fdw
   2.1. Создать еще одну БД: *db25_remote*.
   2.2. В *db25_remote* создать в схеме *public* таблицу *t* произвольной структуры.
   2.3. В *db25* создать внешнюю таблицу *t_remote* для таблицы *t* из *db25_remote*.
   2.4. В *db25* открываем транзакцию с уровнем изоляции READ COMMITTED и проверяем содержимое внешней таблицы *t_remote*.
   2.5. Открываем еще один сеанс в *db25_remote* и вносим изменения в таблицу *t*. Фиксируем изменения.
   2.6. Не закрывая транзакцию в *db25*, еще раз проверяем содержимое внешней таблицы *t_remote*.
   2.7. Видны ли сделанные изменения? Почему?



Авторы: Егор Рогов, Павел Лузанов

Обратная связь
Отзывы, замечания и предложения направляйте по адресу:
edu@postgrespro.ru

Темы

Назначение

Настройка доступа к внешним данным

file_fdw

postgres_fdw

Доступные FDW

Назначение

Доступ к внешним данным

- источник данных вне текущей БД
- возможно как чтение, так и запись

Прозрачно для приложения

- внешние данные представлены в виде таблиц БД
- доступ через GRANT/REVOKE
- команды DML, триггеры

Стандарт ISO/IEC 9075-9 (SQL/MED)

В стандарте ISO/IEC 9075-9 (SQL/MED) определяется, каким образом базы данных SQL должны работать с внешними источниками. При этом доступ к внешним данным возможен как на чтение, так и на запись.

Внешние данные представлены в PostgreSQL как таблицы, которые называются *внешними* или *сторонними*. С ними можно работать при помощи обычных команд DML: insert, update, delete, select. На внешние таблицы можно создавать триггеры. А для разграничения прав используются команды GRANT/REVOKE.

Основное отличие внешних таблиц от обычных в том, что данные физически не хранятся в БД PostgreSQL, а загружаются/отправляются из внешнего источника при выполнении запросов.

PostgreSQL не полностью поддерживает данный стандарт, но это направление сейчас очень активно развивается.

Более подробная информации о работе с внешними данными: http://www.postgresql.org/docs/9.5/static/ddl-foreign-data.html

Настройка доступа

Для получения доступа к внешним данным нужно создать несколько объектов:

1. Обертка сторонних данных
2. Внешний сервер
3. Сопоставление пользователей
4. Внешняя таблица

Прежде чем начать работать с внешними данными, необходимо настроить к ним доступ.

В общем случае настройка состоит из следующих этапов:

1. Создание обертки сторонних данных
2. Создание внешнего сервера
3. Определение сопоставления локальных пользователей и удаленных учетных записей (если это необходимо)
4. Создание внешних таблиц

Обертка сторонних данных

Назначение

API для работы с внешним источником

Создание

CREATE FOREIGN DATA WRAPPER

CREATE EXTENSION

только суперпользователь

Информация

pg_foreign_data_wrapper \dew

Обертка сторонних данных (foreign data wrapper или fdw) — это библиотека функций, которые реализуют доступ к определенному типу источника данных. Например, postgres_fdw реализует доступ к базам данных PostgreSQL, a file_fdw — доступ к файлам операционной системы.

Для создания обертки сторонних данных используется команда SQL CREATE FOREIGN DATA WRAPPER.

В настоящий момент все доступные fdw распространяются в виде расширений. Поэтому при установке расширения автоматически устанавливается соответствующая обертка.

Создать обертку сторонних данных может только суперпользователь.

Внизу этого и следующих слайдов приведены имена таблиц системного каталога и команды psql, которыми можно найти информацию о рассматриваемых объектах.

Внешний сервер

Назначение

содержит информацию о подключении к внешнем источнику

Создание

CREATE SERVER

на один FDW может быть несколько серверов

например, для каждой БД, к которой нужно подключиться

указываются специфичные для сервера опции подключения

например, для postgres_fw: OPTIONS (dbname 'postgres' host 'remotehost' port '5432')

привилегия USAGE на обертку сторонних данных

Информация

pg_foreign_server \des

Внешний сервер, как правило, содержит информацию о подключении к источнику данных. Для баз данных это обычно узел, имя базы данных, порт. Но не имя пользователя и пароль. Эта информация появится на этапе создания сопоставления пользователей.

Для создания внешнего сервера используется SQL-команды CREATE SERVER.

Если требуется подключение к нескольким внешним БД, то для каждой из них нужно создать отдельный сервер.

После ключевого слова OPTIONS указываются специфичные для внешней БД параметры подключения. Например, в postgres_fdw это информация об узле, базе данных, номере порта.

Для создания внешнего сервера требуется привилегия USAGE на ранее созданную обертку сторонних данных (fdw). Как и везде в PostgreSQL, владелец объекта (fdw) и суперпользователь также могут создавать внешние сервера.

Сопоставление ролей

Назначение

настройка соответствия между локальными ролями
и учетными записями на внешнем сервере

Создание

CREATE USER MAPPING

указываются дополнительные опции подключения к внешнему серверу, такие, как имя пользователя, пароль

например, для postgres_fdw: OPTIONS (user 'postgres' password 'qwerty')

владелец сервера, роль с привилегией USAGE на сервер

Информация

pg_user_mappings \deu

Внешний источник может потребовать аутентификацию при подключении. В таких случаях необходимо сопоставить, какие локальные роли могут подключаться к каким учетным записям на внешнем сервере.

Для создания сопоставления пользователей используется SQL-команда CREATE USER MAPPING.

После ключевого слова OPTIONS указываются дополнительные параметры для аутентификации на внешнем сервере, такие, как имя пользователя, пароль.

Для создания сопоставления пользователей требуется привилегия USAGE на внешний сервер.

Внешние таблицы

Назначение

доступ к данным на внешнем сервере

Создание

CREATE FOREIGN TABLE

IMPORT FOREIGN SCHEMA

участие в наследовании

включение ограничений: NOT NULL, CHECK

декларативные, могут использоваться планировщиком

владелец сервера, роль с привилегией USAGE на сервер

Информация

pg_foreign_table pg_class where relkind = 'f' \det

После создания внешнего сервера и сопоставлений пользователей (при необходимости) можно приступить к созданию внешних таблиц.

Для этого есть два способа:

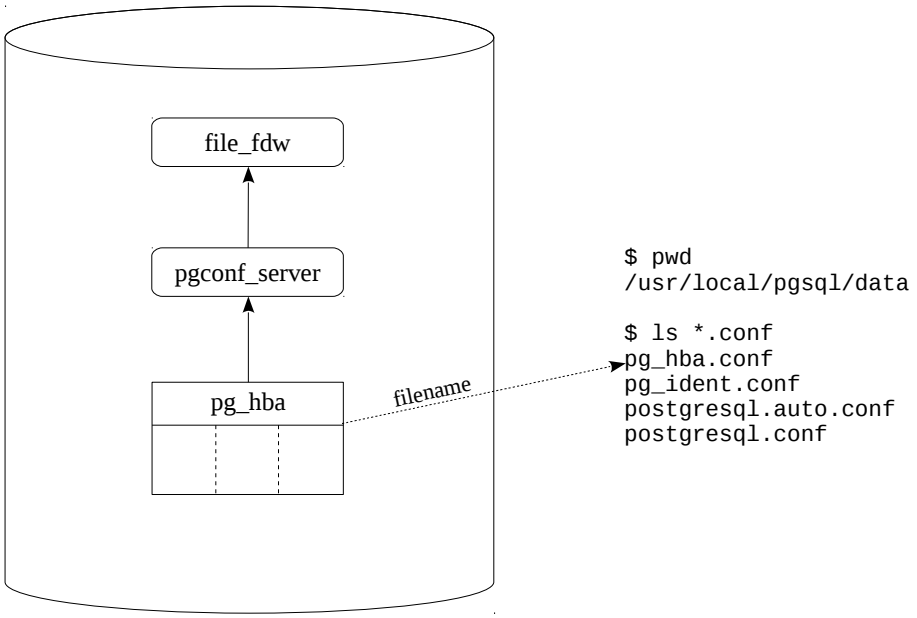
1. Создание SQL-командой CREATE FOREIGN TABLE.

2. Импорт таблиц при помощи SQL-команды IMPORT FOREIGN SCHEMA.

Отметим, что внешние таблицы могут участвовать в наследовании. Также для внешних таблиц можно добавлять ограничения целостности NOT NULL и CHECK. Эти ограничения не будут применяться при выполнении команд, ведь в любом случае PostgreSQL не сможет обеспечить соблюдение ограничений на внешнем источнике. Но планировщик может использовать информацию об ограничениях при построении плана выполнения запроса. Вместе с участием в наследовании это дает возможность использовать внешнюю таблицу в качестве секции в секционированной таблице.

Для создания внешних таблиц требуется привилегия USAGE на внешний сервер.

file_fdw



file_fdw предназначен для доступа к файловой системе сервера PostgreSQL.

Доступ возможен только на чтения и выполняется посредством SQL-команды COPY … FROM.

Рассмотрим пример создания внешней таблицы для доступа к файлу pg_hba.conf. Для этого можно выполнить следующие действия:

1. Установить расширение *file_fdw*, которое создаст обертку сторонних данных *file_fdw*.

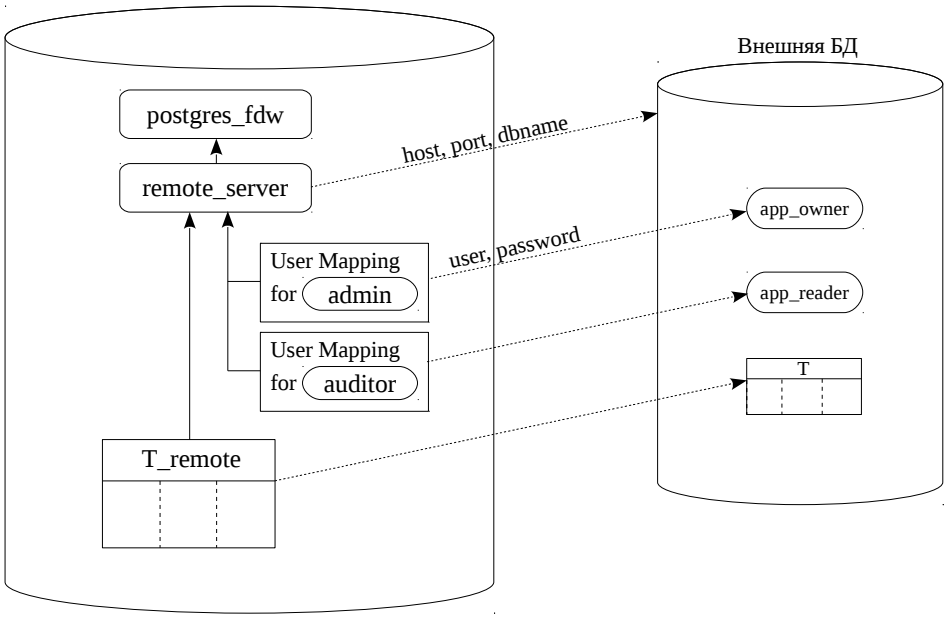
2. Создать сервер *pgconf_server.*

3. Создать внешнюю таблицу *pg_hba*. В простом случае таблица может состоять из одного столбца типа *text*. При этом каждая строка файла будет представлена отдельной записью в таблице. В опциях нужно указать полное имя файла *pg_hba.conf*.

Обратим внимание, что сопоставление пользователей в file_fdw не требуется.

Более подробная информации о file_fdw:
http://www.postgresql.org/docs/9.5/static/file-fdw.html

postgres_fw



postgres_fdw предназначен для доступа к базам данных PostgreSQL.

Посмотрим, как работать с postgres_fdw на следующем примере. Мы хотим настроить доступ к таблице T, расположенной во внешней БД PostgreSQL. В этой БД есть две роли: app_owner — владелец таблицы и app_reader — имеет доступ только на чтение. Мы хотим сделать внешнюю таблицу и предоставить к ней доступ двум локальным ролям: admin — полный доступ, auditor — только чтение.

Для реализации такого сценария выполним следующие действия:

1. Установим расширение *postgres_fdw*, которое создаст обертку сторонних данных *postgres_fdw*.
2. Создадим сервер *remote_server.* В опциях укажем имя внешней БД, узел и порт.
3. Создадим два сопоставления пользователей. Первое для роли *admin*, в опциях укажем роль *app_owner* и пароль (если есть) внешней БД. Второе — для роли *auditor*, которую свяжем с *app_reader*.
4. Создадим внешнюю таблицу *t_remote*.
5. Для роли *admin* можно выдать привилении на *t_remote,* разрешающие как чтение, так и запись. А для роли *auditor* выдать права только на чтение.

Более подробная информации о postgres_fdw:
http://www.postgresql.org/docs/9.5/static/postgres-fdw.html

Доступные FDW

Поставляются вместе с PostgreSQL (модули contrib)

- file_fdw
- postgres_fdw

Внешние источники

- страница на wiki.postgresql.org
- PostgreSQL Extension Network

… написать свою

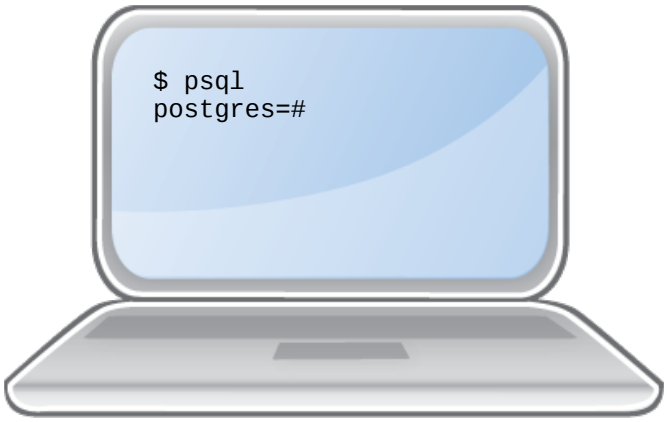
Рассмотренные обертки сторонних данных, file_fdw и postgres_fdw, поставляются вместе с PostgreSQL в виде contrib-модулей.

Помимо этого существует множество других fdw, которые предоставляют доступ ко многим популярным базам данных и не только. Информацию о них можно получить:

- на страничке wiki: https://wiki.postgresql.org/wiki/Fdw
- на сайте PGXN: http://pgxn.org/tag/fdw/

Если нужную обертку не удалось найти, то ее можно написать самостоятельно. О том, как это сделать, говорится в документации: http://www.postgresql.org/docs/9.5/static/fdwhandler.html

Демонстрация



Итоги

Работа с внешними данными регламентируется стандартом ISO/IEC 9075-9 (SQL/MED)

В общем случае, настройка доступа к внешним данным заключается в определении объектов БД:

- обертки сторонних данных
- внешние сервера
- сопоставления пользователей
- внешние таблицы

Вместе с PostgreSQL поставляются два модуля: file_fdw и postgres_fdw

Практика

1. file_fdw
   1.1. Сделать в базе данных *db25* внешнюю таблицу *pg_hba* для просмотра конфигурационного файла pg_hba.conf.
   1.2. Проверить содержимое таблицы *pg_hba*
2. postgres_fdw
   2.1. Создать еще одну БД: *db25_remote*.
   2.2. В *db25_remote* создать в схеме *public* таблицу *t* произвольной структуры.
   2.3. В *db25* создать внешнюю таблицу *t_remote* для таблицы *t* из *db25_remote*.
   2.4. В *db25* открываем транзакцию с уровнем изоляции READ COMMITTED и проверяем содержимое внешней таблицы *t_remote*.
   2.5. Открываем еще один сеанс в *db25_remote* и вносим изменения в таблицу *t*. Фиксируем изменения.
   2.6. Не закрывая транзакцию в *db25*, еще раз проверяем содержимое внешней таблицы *t_remote*.
   2.7. Видны ли сделанные изменения? Почему?

1. FILE_DFW

   Все действия под пользователем *postgres* в базе данных *db25*:
   - Создать расширение file_*fwd*
   - Создать внешний сервер *pg_conf* для *file_fdw*
   - Создать внешнюю таблицу *pg_hba* с одним столбцом типа *text*
   - options (filename '/usr/local/pgsql/data/pg_hba.conf')
2. POSTGRES_FDW

   Для создания в *db25* внешней таблицы *t_remote* для таблицы *t* из *db25_remote* (пункт 2.3):
   - Создать расширение *postgres_fwd*
   - Создать внешний сервер *remote_server*
     `options (dbname 'db25_remote')`
   - Создать соответствие для пользователя *postgres*
     `options (user 'postgres')`
   - Создать внешнюю таблицу *t_remote*
     `options (schema_name 'public', table_name 't')`