PostGIS

Шпаргалка по основным командам PostgreSQL

Вся работа с PostgreSQL осуществляется под пользователем postgres.

```
sudo su postgres
```

Работать с PosgreSQL можно как в интерактивном режиме, так и из командной строки. Программа — psql.

```
psql
```

Основные команды PostgreSQL в интерактивном режиме:

- \connect db_name – подключиться к базе с именем db_name
- \du – список пользователей
- \dp (или \z) – список таблиц, представлений, последовательностей, прав доступа к ним
- \di – индексы
- \ds – последовательности
- \dt – список таблиц
- \dt+ — список всех таблиц с описанием
- \dt *s* — список всех таблиц, содержащих s в имени
- \dv – представления
- \dS – системные таблицы
- \d+ – описание таблицы
- \o – пересылка результатов запроса в файл
- \l – список баз данных
- \i – читать входящие данные из файла
- \e – открывает текущее содержимое буфера запроса в редакторе (если иное не указано в окружении переменной EDITOR, то будет использоваться по умолчанию vi)
- \d “table_name” – описание таблицы
- \i запуск команды из внешнего файла, например \i /my/directory/my.sql
- \pset – команда настройки параметров форматирования
- \echo – выводит сообщение
- \set – устанавливает значение переменной среды. Без параметров выводит список текущих переменных (\unset – удаляет).
- \? – справочник psql
- \help – справочник SQL
- \q (или Ctrl+D) – выход с программы

Работа с PostgreSQL из командной строки:

- -c (или –command) – запуск команды SQL без выхода в интерактивный режим
- -f file.sql — выполнение команд из файла file.sql
- -l (или –list) – выводит список доступных баз данных
- -U (или –username) – указываем имя пользователя (например postgres)

- -W (или –password) – приглашение на ввод пароля
- -d dbname — подключение к БД dbname
- -h – имя хоста (сервера)
- -s – пошаговый режим, то есть, нужно будет подтверждать все команды
- –S – однострочный режим, то есть, переход на новую строку будет выполнять запрос (избавляет от ; в конце конструкции SQL)
- -V – версия PostgreSQL без входа в интерактивный режим

Примеры:

Выполнение команды в базе dbname

```
psql -U postgres -d dbname -c «CREATE TABLE my(some_id serial PRIMARY KEY, some_text text);»
```

Вывод результата запроса в html-файл

```
psql -d dbname -H -c «SELECT * FROM my» -o my.html
```

Утилиты (программы) PosgreSQL:

- createdb и dropdb – создание и удаление базы данных (соответственно)
- createuser и dropuser – создание и пользователя (соответственно)
- pg_ctl – программа предназначенная для решения общих задач управления (запуск, останов, настройка параметров и т.д.)
- postmaster – многопользовательский серверный модуль PostgreSQL (настройка уровней отладки, портов, каталогов данных)
- initdb – создание новых кластеров PostgreSQL
- initlocation – программа для создания каталогов для вторичного хранения баз данных
- vacuumdb – физическое и аналитическое сопровождение БД
- pg_dump – архивация и восстановление данных
- pg_dumpall – резервное копирование всего кластера PostgreSQL
- pg_restore – восстановление БД из архивов (.tar, .tar.gz)

Примеры создания резервных копий:

Создание бекапа базы mydb, в сжатом виде

```
pg_dump -h localhost -p 5440 -U someuser -F c -b -v -f mydb.backup mydb
```

Создание бекапа базы mydb, в виде обычного текстового файла, включая команду для создания БД

```
pg_dump -h localhost -p 5432 -U someuser -C -F p -b -v -f mydb.backup mydb
```

Создание бекапа базы mydb, в сжатом виде, с таблицами которые содержат в имени payments

```
pg_dump -h localhost -p 5432 -U someuser -F c -b -v -t *payments* -f
payment_tables.backup mydb
```

Дамп данных только одной, конкретной таблицы. Если нужно создать резервную копию нескольких таблиц, то имена этих таблиц перечисляются с помощью ключа -t для каждой таблицы.

```
pg_dump -a -t table_name -f file_name database_name
```

Создание резервной копии с сжатием в gz

```
pg_dump -h localhost -O -F p -c -U postgres mydb | gzip -c > mydb.gz
```

Список наиболее часто используемых опций:

- -h host — хост, если не указан то используется localhost или значение из переменной окружения PGHOST.
- -p port — порт, если не указан то используется 5432 или значение из переменной окружения PGPORT.
- -u — пользователь, если не указан то используется текущий пользователь, также значение можно указать в переменной окружения PGUSER.
- -a, —data-only — дамп только данных, по-умолчанию сохраняются данные и схема.
- -b — включать в дамп большие объекты (blog’и).
- -s, —schema-only — дамп только схемы.
- -C, —create — добавляет команду для создания БД.
- -c — добавляет команды для удаления (drop) объектов (таблиц, видов и т.д.).
- -O — не добавлять команды для установки владельца объекта (таблиц, видов и т.д.).
- -F, —format {c|t|p} — выходной формат дампа, custom, tar, или plain text.
- -t, —table=TABLE — указываем определенную таблицу для дампа.
- -v, —verbose — вывод подробной информации.
- -D, —attribute-inserts — дамп используя команду INSERT с списком имен свойств.

Бекап всех баз данных используя команду pg_dumpall.

```
pg_dumpall > all.sql
```

Восстановление таблиц из резервных копий (бэкапов):

Восстановление бекапов, которые хранятся в обычном текстовом файле (plain text)

```
psql
```

Восстановление сжатых бекапов (tar)

```
pg_restore
```

Восстановление всего бекапа с остановкой на первой ошибке

```
psql -h localhost -U someuser –set ON_ERROR_STOP=on -f mydb.sql
```

Для восстановления из tar-арихива нам понадобиться сначала создать базу с помощью CREATE

```
DATABASE mydb; (если при создании бекапа не была указана опция -C) и восстановить
```

```
pg_restore –dbname=mydb –jobs=4 –verbose mydb.backup
```

Восстановление резервной копии БД, сжатой gz

```
gunzip mydb.gz
psql -U postgres -d mydb -f mydb
```

ERROR: must be owner of relation planet_osm_nodes

В окне терминала Ubuntu и из «renderaccount»:
Войдите в систему как встроенный пользователь Ubuntu «postgres»

```
sudo -u postgres -i
```

Подключиться к базе данных «gis»

```
postgres=# \c gis
```

Теперь вы подключены к базе данных «gis» как пользователь «postgres».
Список таблиц «gis»

```
gis=# \dt
```

Список отношений

```
Schema | Name | Type | Owner
--------+------------------+-------+---------------
public | planet_osm_nodes | table | postgres
public | planet_osm_rels | table | postgres
public | planet_osm_ways | table | postgres
public | spatial_ref_sys | table | renderaccount
(4 rows)
```

Изменить владельца таблиц

```
gis=# ALTER TABLE planet_osm_nodes OWNER TO renderaccount;
ALTER TABLE
gis=# ALTER TABLE planet_osm_rels OWNER TO renderaccount;
ALTER TABLE
gis=# ALTER TABLE planet_osm_ways OWNER TO renderaccount;
ALTER TABLE
```

Список таблиц «gis» для подтверждения изменений

```
gis=# \dt
```

Список отношений

```
Schema | Name | Type | Owner
--------+------------------+-------+---------------
public | planet_osm_nodes | table | renderaccount
public | planet_osm_rels | table | renderaccount
public | planet_osm_ways | table | renderaccount
public | spatial_ref_sys | table | renderaccount
(4 rows)
```

Оставьте «gis»

```
gis-# \q
```

Выйти «postgres»

```
exit
```

из системы

...снова в «renderaccount»

```
osm2pgsql -d gis --create --slim -G --hstore --tag-transform-script
~/src/openstreetmap-carto/openstreetmap-carto.lua -C 2500 --number-processes
1 -S ~/src/openstreetmap-carto/openstreetmap-carto.style ~/data/bulgaria-
latest.osm.pbf
```

...Proceed with the tutorial.