ALT Linux Wiki

PostgreSQL

Особенности работы с PostgreSQL (http://www.postgresql.org) в Альт.

Содержание

Установка и начальный запуск

> **Примечание:** Информация об установке отечественной Postgres Pro Standard приведена на отдельной странице

Вы можете выбрать сервер PostgreSQL определённой версии как обычный, так и предназначенный для работы 1С:Предприятие (содержит в имени 1C):

```
postgresql15-server
postgresql14-server
postgresql14-1C-server
postgresql13-server
postgresql12-server
postgresql11-server
postgresql10-server
postgresql9.6-server
```

```
apt-get update
apt-get install postgresql14-server
```

> **Внимание!** Перед запуском службы необходимо создать системные базы данных:
>
> ```
> /etc/init.d/postgresql initdb
> ```
>
> (через systemctl postgresql initdb не работает)

Запуск службы:

```
service postgresql start
```

или

```
systemctl start postgresql
```

Включение службы по умолчанию:

```
chkconfig postgresql on
```

или

```
systemctl enable postgresql
```

Доступ по сети

По умолчанию доступ по сети выключен. Для того, чтобы включить, выполните:

```
echo "listen_addresses = 'localhost'" >> /var/lib/pgsql/data/postgresql.conf
```

> **Примечание:** Обратите внимание, что доступ открыт только на **localhost**. Если хотите открыть на внешнем интерфейсе, указывайте реальный адрес IP или имя узла.

Управление доступом

Для управления доступом, правьте файл **`/var/lib/pgsql/data/pg_hba.conf`**:

```
echo "host ПОЛЬЗОВАТЕЛЬ БАЗА 127.0.0.1/32 md5" >> /var/lib/pgsql/data/pg_hba.conf
```

не забудьте после всего этого перезапустить службу:

```
service postgresql restart
```

Работа под администратором

Для заведения пользователей и создания баз данных, переключитесь в учётную запись `postgres`:

```
psql -U postgres
```

```
# psql -U postgres
psql (9.4.5)
Введите "help", чтобы получить справку.

postgres=#
```

> **Примечание:** Выход по Ctrl+D или командой `quit`

Создание пользователя

```
createuser -U postgres --no-superuser --no-createdb --no-createrole --encrypted --pwprompt
ПОЛЬЗОВАТЕЛЬ
```

Создание базы данных

```
createdb -U postgres -O [ПОЛЬЗОВАТЕЛЬ] [БАЗА]
```

Просмотр доступных баз данных

```
# psql -U postgres -c "\l+"
Список баз данных
    Имя     |  Владелец  | Кодировка | LC_COLLATE  |  LC_CTYPE   |     Права доступа     |
Размер  | Табл. пространство |                  Описание
------------+------------+-----------+-------------+-------------+-----------------------+------
---+--------------------+-------------------------------------------
 postfactor | postfactor | UTF8      | ru_RU.UTF-8 | ru_RU.UTF-8 |                       | 12 MB
| pg_default         |
 postgres   | postgres   | UTF8      | ru_RU.UTF-8 | ru_RU.UTF-8 |                       | 6724
kB | pg_default         | default administrative connection database
 sogo       | sogo       | UTF8      | ru_RU.UTF-8 | ru_RU.UTF-8 | =Tc/sogo             +| 7572
kB | pg_default         |
            |            |           |             |             | sogo=CTc/sogo         |
|                    |
 template0  | postgres   | UTF8      | ru_RU.UTF-8 | ru_RU.UTF-8 | =c/postgres          +| 6601
kB | pg_default         | unmodifiable empty database
            |            |           |             |             | postgres=CTc/postgres |
|                    |
 template1  | postgres   | UTF8      | ru_RU.UTF-8 | ru_RU.UTF-8 | =c/postgres          +| 6724
kB | pg_default         | default template for new databases
            |            |           |             |             | postgres=CTc/postgres |
|                    |
(5 строк)
```

Имена баз в первом столбце.

Резервная копия всех баз данных

```
pg_dumpall -U postgres -f /tmp/posgresql
```

Резервная копия будет в файле `/tmp/posgresql`.

Восстановление из резервной копии

Документация: https://www.postgresql.org/docs/9.0/static/migration.html

```
mv /var/lib/pgsql/data{,.old}
/etc/init.d/postgresql initdb
service postgresql start
psql -U postgres -f /tmp/posgresql postgres
```

Continuous Archiving and Point-in-Time Recovery

Если в настройках сервера включено ведение журналов упреждающей записи (WAL), то появляется возможность резервного копирования на уровне файловой системы с копированием журналов. В этом случае при сбое мы можем восстановить базу от контрольной точки до произвольного момента времени, после контрольной точки, с помощью файлов журналов. Подробнее об этом можно почитать здесь (https://www.postgresql.org/docs/9.6/static/continuous-archiving.html)

Настройка журналов WAL

Для того, чтобы PostgreSQL начал писать файлы журналов необходимо изменить конфигурационный файл сервера `/var/lib/pgsql/data/postgresql.conf` следующим образом:

```
wal_level = archive # указывается уровень ведения журналов
archive_mode = on # разрешается ведение журналов
archive_command = '/var/lib/pgsql/bin/copy_wal.sh "%f" "%p"'
```

archive_command - это команда выполняемая сервером над журналом который надо архивировать. %p заменяется полным путем к файлу, %f заменяется именем файла. Простейшая команда это `'test ! -f /mnt/server/archivedir/%f && cp %p /mnt/server/archivedir/%f'` она проверяет не был ли файл уже архивирован, и если не был то копирует его в каталог /mnt/server/archivedir/
Создадим каталог для архивных журналов и установим нужные права:

```
# mkdir /var/lib/pgsql/wals
# chown postgres:postgres /var/lib/pgsql/wals
# chmod 700 /var/lib/pgsql/wals
```

Также создадим каталог для нашего скрипта архивирования журналов:

```
# mkdir /var/lib/pgsql/bin
# chown postgres:postgres /var/lib/pgsql/bin
# chmod 700 /var/lib/pgsql/bin
```

Рассмотрим файл `/var/lib/pgsql/bin/copy_wal.sh`:

```
#!/bin/bash
DPATH=/var/lib/pgsql/wals
DATE=`date +"%b %d %T"`
if [ -e /var/lib/pgsql/backup_in_progress ]; then # По наличию файла проверяет не идет ли процесс
резервного копирования журналов
```

```
резервного копирования журналов
 echo "${DATE} - идет процесс резервного копирования журналов" >> "${DPATH}/wal-c-log.log"
 exit 1
fi
if [ -e ${DPATH}/$1 ]; then # Проверяет скопирован ли журнал раннее
 echo "${DATE} - файл уже архивирован" >> "${DPATH}/wal-c-log.log"
 exit 1
fi
echo "${DATE} - /bin/gzip $2 ${DPATH}/$1" >> "${DPATH}/wal-c-log.log"
gzip < $2 > "${DPATH}/$1" # Архивирует файл журнала
```

Скрипт проверяет состояния резервного копирования журналов и архивирует журналы в каталог **`/var/lib/pgsql/wals`**, все свои действия он пишет в **`/var/lib/pgsql/wals/wal-c-log.log`**
Установим нужные права на скрипт:

```
# chown postgres:postgres /var/lib/pgsql/bin/copy_wal.sh
# chmod 700 /var/lib/pgsql/bin/copy_wal.sh
```

Теперь перезапустим сервер базы данных:

```
# systemctl restart postgresql
```

Со временем в каталоге **`/var/lib/pgsql/wals`** будут появляться файлы вида:

```
-rw------- 1 postgres postgres 3848185 июл 12 18:40 000000010000001900000048
-rw------- 1 postgres postgres 3830181 июл 12 18:33 000000010000001900000047
-rw------- 1 postgres postgres 4131659 июл 12 18:26 000000010000001900000046
-rw------- 1 postgres postgres 3977349 июл 12 18:26 000000010000001900000045
```

Если они появляются значит журналирование настроены правильно.

Онлайн резервное копирование

Используя Low Level API

Использование низкоуровневого API подразумевает следующие шаги:
1. Убедитесь что журналирование включено и работает
2. Подключитесь к базе от имени суперпользователя:

```
# psql -U postgres
```

Для начала резервного копирования выполните следующий запрос:

```
postgres=# select pg_start_backup('Full Backup - Testing');
```

Этот запрос может выполняться длительное время, так как создается контрольная точка и минимизируется влияние на текущие операции.
Чтобы начать резервное копирование как можно быстрее (используя все ресурсы для создания контрольной точки) выполните команду с параметром **`true`**:

```
postgres=# select pg_start_backup('Full Backup - Testing',true);
```

Пример вывода команды **pg_start_backup**:

```
pg_start_backup
-----------------
 1A/20000028
 (1 строка)
```

3. Теперь можно делать файловое резервное копирование каталога с базой данных (`/var/lib/pgsql/data`) любыми удобными инструментами (cp, rsync и т.д.)
4. После окончания резервирования подключитесь к базе снова и введите команду:

```
postgres=# select pg_stop_backup();
```

При этом сервер выйдет из режима резервного копирования и переключится на новый файл журнала (WAL). Переключение на новый файл журнала необходимо для того, чтобы все изменения произошедшие во время резервного копирования сразу были доступны для архивирования.

5. Теперь остается в дополнение к файловой резервной копии добавить файлы журналов, созданные во время резервирования. Для удобства в каталоге с журналами будет создан файл вида `000000010000001A00000020.00000028.backup`, в котором находится информация о резервной копии. Первая часть имени этого файла

```
000000010000001A00000020
```

обозначает имя первого журнала WAL необходимого для восстановления базы. Журналы с меньшим именем могут быть удалены.

С помощью команды pg_basebackup

Так же резервную копию можно создать использую команду **pg_basebackup**. Эта команда создает копию в виде файлов или архива tar.
Для ее работы необходимо дополнительно настроить сервер базы данных (`/var/lib/pgsql/data/postgresql.conf`):

```
max_wal_senders = 1
```

И разрешить подсоединение пользователю postgres для репликации раскомментировать нужную строчку в файле `/var/lib/pgsql/data/pg_hba.conf`:

```
local   replication     postgres
```

Перезапустите сервер баз данных:

```
# systemctl restart postgresql
```

Пример использования команды:

```
# pg_basebackup -D /var/lib/pgsql/backup -F t -z -U postgres -w -c fast -l "pg_basebackup test
backup"
```

Описание параметров команды:

- -D -- каталог для архивации (должен быть пустой)
- -F -- формат вывода (t - tar архив)
- -z -- сжимает выходной файл gzip
- -U -- пользователь для подключения к базе
- -w -- не запрашивать пароль
- -c -- режим создания контрольной точки (fast - быстрая)
- -l -- метка резервной копии

После выполнения команды в каталоге `/var/lib/pgsql/backup` появится архив `base.tar.gz`. А в каталоге с журналами будет создан файл с описанием резервной копии. В дополнение к резервной копии необходимо копировать архивы журналов.

Восстановление из резервной копии

Для восстановления из резервной копии, сделанной с помощью механизма Continuous Archiving, потребуется архив с базовой резервной копией и файлы журналов WAL.
1. Остановите сервер.
2. Если позволяет свободное место временно скопируйте каталог с базой данных. Если места недостаточно сохраните как минимум содержимое подкаталога `pg_xlog` каталога базы данных, так в нем хранятся не архивированные журналы WAL.
3. Восстановите все файлы из архива с резервной копией. Проверьте права и владельца файлов. Они должны принадлежать пользователю от имени которого запускается сервер PostgreSQL.
4. Удалите все файлы из восстановленного подкаталога `pg_xlog`, так как содержащиеся там файлы журналов устарели.
5. Скопируйте не архивированные файлы журналов сохраненные в п.2 в подкаталог `pg_xlog` восстановленной базы данных. Проверьте права и владельца.
6. В каталоге базы данных `/var/lib/pgsql/data` создайте файл `recovery.conf`. Файл должен иметь права 600 и владельца от имени которого запускается сервер. Запустите сервер PostgreSQL. Если восстановление по какой-то причине прервется просто запустите сервер еще раз. После окончания восстановления файл `recovery.conf` переименуется в файл `recovery.done`, чтобы исключить повторный запуск восстановления. Сервер готов к работе.
В файле `recovery.conf` должна быть прописана как минимум одна команда `restore_command`, которая указывает откуда берутся файлы журналов.
Предположим что резервную копию мы разархивировали в каталог `/var/lib/pgsql/data`
Архивные журналы находятся в каталоге `/var/lib/pgsql/wal`
Журналы архивировались командой **archive_command = 'gzip < %p > /var/lib/pgsql/wal/%f'**
Тогда файл `recovery.conf` должен содержать следующую строку:

```
restore_command = 'gunzip < /var/lib/pgsql/wal/%f > %p'
```

Также можно задать момент времени до которого нужно проводить восстановление:

```
recovery_target_time = '2017-05-25 19:07:01'
```

Если метку времени не задавать будет выполнено восстановление на сколько это возможно.

Резервное копирование с помощью pg_probackup

Программа для резервного копирования разрабатываемая PostgresPro Документация (https://postgrespro.ru/docs/postgrespro/11/app-pgprobackup) GitHub (https://postgrespro.ru/docs/postgrespro/11/app-pgproback

up)

Установка

Для установки используем подготовленные для ОС Альт собранных на p9 репозитории PostgresPro, перечень для других платформ можно найти на GitHub:

```
#echo rpm "http://repo.postgrespro.ru/pg_probackup/rpm/latest/altlinux-p9 x86_64 vanilla" >
/etc/apt/sources.list.d/pg_probackup.list
#apt-get update
#apt-get install pg_probackup-version
```

где version = 12,11,10,9.6,9.5

Переход на новую версию

https://www.postgresql.org/docs/10/upgrading.html
