Базовый инструментарий

Установка и управление сервером

Авторы: Егор Рогов, Павел Лузанов, Илья Баштанов

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

Темы занятия

Основные понятия

Установка из исходных кодов

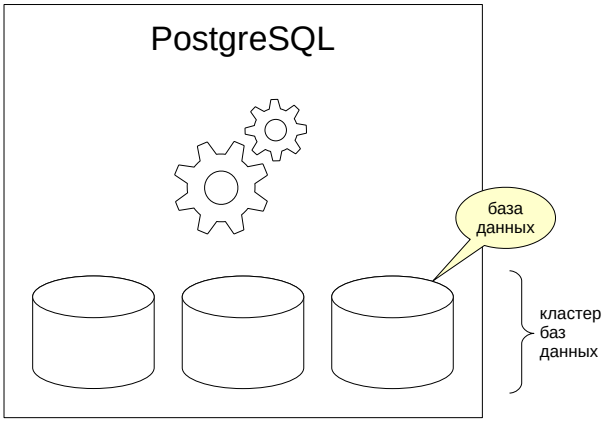
Управление сервером

Пакетная установка

Управление сервером в Ubuntu

Облачные решения

Кластер БД и сервер



Прежде чем говорить об установке, рассмотрим некоторые основные понятия.

PostgreSQL — программа, которая относится к классу *систем управления базами данных*.

Когда эта программа выполняется, мы называем ее *сервером* PostgreSQL или *экземпляром сервера*. Пока сервер представляется для нас «черным ящиком», но постепенно мы познакомимся с тем, как он устроен.

Данные, которыми управляет PostgreSQL, хранятся в *базах данных*. Один экземпляр PostgreSQL одновременно работает с несколькими базами данных. Этот набор баз данных называется *кластером баз данных*. Подробнее мы будем говорить о базах данных в теме «Организация данных. Базы данных и схемы».

Итак: кластер баз данных — это данные в файлах; сервер или экземпляр сервера — программа, управляющая кластером баз данных.

Исходные коды

Установка из исходных кодов

- стабильная версия сервера
- можно использовать нестандартные параметры или собрать на специфичной архитектуре

Установка из репозитория git

- текущая версия сервера
- необходимо в первую очередь для разработчиков ядра; требует более широкого набора инструментов

Полезно иметь представление о том, как происходит установка из исходных кодов, чтобы в случае необходимости можно было собрать PostgreSQL с нестандартными параметрами или на специфичной архитектуре.

По адресу https://www.postgresql.org/ftp/source/v13.6/ доступен исходный код версии 13.6 в двух вариантах (gzip и bzip2).
В виртуальной машине необходимый файл уже находится в домашнем каталоге пользователя postgres.

Исходные коды можно взять и непосредственно из git-репозитория проекта: git://git.postgresql.org/git/postgresql.git (проект имеет зеркала, в том числе на github).

В этом случае можно собрать не только стабильную версию, но и версию на любой конкретный коммит, в том числе наиболее актуальную.

При установке из репозитория git требуется более широкий набор инструментов. Например, лексический и синтаксический анализатор сделан с помощью Flex и Bison, и в git хранятся именно исходные коды для этих инструментов. При сборке из них генерируются файлы на Си, которые затем компилируются. При этом в архиве с исходными кодами находится уже сгенерированные файлы Си.

Необходимое ПО

Обязательно

- tar, gzip/bzip2, GNU make, компилятор Си (C89)

Используются, но можно отключить

- библиотеки GNU Readline, zlib

Дополнительно

- языки программирования Perl, Python, Tcl
  для использования PL/Perl, PL/Python, PL/Tcl
- Kerberos, OpenSSL, OpenLDAP, PAM
  для аутентификации и шифрования
- библиотека ICU для кроссплатформенной поддержки UNICODE

Отдельные инструменты при сборке из репозитория git

Для сборки требуется ряд программ и утилит.

Библиотека readline дает возможность редактировать командную строку, пользоваться историей команд и автодополнением. В серверной инсталляции может быть и не нужна, если не предполагается запускать psql в консоли.

Библиотека zlib используется для сжатия архивов pg_dump.

Распаковка архива

Архив с исходными кодами находится в домашнем каталоге пользователя student. Распаковываем архив и переходим в каталог с исходным кодом:

```
student$ tar xzf /home/student/postgresql-13.6.tar.gz
```

```
student$ ls -ld /home/student/postgresql-13.6
```

```
drwxrwxr-x 6 student student 4096 фев  8 00:29 /home/student/postgresql-13.6
```

```
student$ cd /home/student/postgresql-13.6
```

Создание конфигурации

Если требуется повторно выполнить конфигурацию, например с другими параметрами, то предварительно нужно очистить результаты предыдущего запуска:

```
student$ make distclean
```

В команде configure можно указать различные параметры конфигурации. Например:

- --prefix — каталог установки, по умолчанию /usr/local/pgsql;
- --enable-debug — для включения отладочной информации.

Полный список параметров есть в документации.

Также принимаются во внимание переменные окружения. Например, CC и CFLAGS настраивают компилятор C.

Для промышленного использования сервер лучше устанавливать в каталог, доступный на запись только суперпользователю. Но для эксперимента установим сервер в домашний каталог пользователя student. Кроме того, будем использовать порт 5555 вместо порта 5432 по умолчанию:

```
student$ ./configure --prefix=/home/student/pgsql13 --with-pgport=5555
```

Вывод этой и следующих команд занимает очень много места, мы его не будем показывать.

Сборка и установка PostgreSQL

Возможные варианты:

- make — сборка только сервера;
- make world — сборка сервера, всех расширений и документации.

Собираем сервер. В зависимости от характеристик компьютера команда make может работать достаточно долго (минуты и даже десятки минут).

```
student$ make
```

Теперь устанавливаем сервер.

```
student$ make install
```

Для сборки расширений нужно повторить последние две команды, перейдя в подкаталог contrib.

```
student$ cd /home/student/postgresql-13.6/contrib
```

```
student$ make
```

```
student$ make install
```

Если бы собирали сервер с расширениями и документацией (make world), то можно было бы установить все вместе командой make install-world.

Создание кластера

Postgres PROFESSIONAL

Утилита

initdb

Особенности

новый кластер баз данных не может принадлежать суперпользователю ОС

PGDATA — удобная переменная для каталога кластера

файлы конфигурации создаются в каталоге PGDATA

⚠ желательно включить расчет контрольных сумм для страниц с данными

После установки сервера необходимо создать кластер баз данных. Для этого предназначена утилита initdb.

В целях безопасности, каталог, в котором инициализируется кластер, не может принадлежать суперпользователю ОС. Владельцем кластера обычно делают пользователя postgres.

Владелец кластера может определить переменную окружения PGDATA, указывающую на каталог кластера. Эту переменную используют некоторые утилиты сервера, когда им нужно узнать расположение кластера. К таким утилитам относится initdb, а также основная утилита управления сервером pg_ctl, о которой мы будем говорить дальше.

В процессе инициализации кластера утилита initdb создает конфигурационные файлы в этом же каталоге PGDATA. Подробнее о файлах конфигурации в одной из следующих тем.

Утилита initdb имеет множество ключей, влияющих на ее работу. Один из важных -k или --data-checksums включает подсчет контрольных сумм на страницах данных. Проверка контрольной суммы будет выполняться при обращении к любой странице данных в кластере. Это несколько снижает производительность, но позволяет оперативно обнаружить повреждения в данных.

https://postgrespro.ru/docs/postgresql/13/app-initdb

Создание кластера

Теперь необходимо создать каталог для данных. Для промышленного использования владельцем каталога обычно назначают отдельного пользователя ОС. В нашем случае, пусть это будет каталог пользователя student внутри /home/student/pgsql13

```
student$ mkdir /home/student/pgsql13/data
```

Этот каталог часто называют PGDATA по имени переменной окружения, которую удобно использовать при работе с утилитами сервера.

```
student$ export PGDATA=/home/student/pgsql13/data
```

Все утилиты сервера установлены в подкаталоге bin, который стоит добавить в переменную PATH:

```
student$ export PATH=/home/student/pgsql13/bin:$PATH
```

Для инициализации кластера базы данных используется утилита initdb.

- В ключе -U можно указать имя суперпользователя СУБД. Если его не задать, то используется имя пользователя ОС, в нашем случае student. Обычно таким пользователем является postgres, так что зададим это имя явно.
- Ключ -k включает подсчет контрольных суммы страниц, что позволяет своевременно обнаруживать повреждение данных.
- Если переменная PGDATA не установлена, то следует в ключе -D указать каталог для данных. В нашем случае этот ключ можно было бы не указывать.

```
student$ initdb -U postgres -k -D /home/student/pgsql13/data
```

Управление сервером

Postgres PROFESSIONAL

Утилита

- pg_ctl

Основные задачи

- запуск, останов и получение статуса сервера
- обновление параметров конфигурации
- переключение на реплику

К основным операциям управления сервером относятся запуск и останов сервера, получение текущего статуса сервера, обновление конфигурации и некоторые другие.

Для выполнения этих действий предназначена утилита pg_ctl, идущая в составе PostgreSQL.

Запускать pg_ctl следует от имени владельца кластера баз данных.

Более подробная информация об управлении сервером для администраторов баз данных:

https://postgrespro.ru/docs/postgresql/13/app-pg-ctl

https://postgrespro.ru/docs/postgresql/13/runtime

Управление сервером

Теперь все готово к запуску сервера.

- В ключе -l укажем файл журнала сообщений сервера.
- Ключ -D опускаем, поскольку задана переменная PGDATA.

```
student$ pg_ctl start -l /home/student/logfile
```

Для проверки работы можно вызвать утилиту psql, которая подключится к серверу и вернет текущее время:

```
student$ psql -U postgres -p 5555 -c 'SELECT now();'
```

```
              now
-------------------------------
 2022-05-06 14:40:00.433906+03
(1 row)
```

Для останова сервера используется команда:

```
student$ pg_ctl stop -m fast
```

В ключе -m можно указать один из трех режимов останова:

- fast — принудительно завершает сеансы и записывает на диск изменения из оперативной памяти;
- smart — ожидает завершения всех сеансов и записывает на диск изменения из оперативной памяти;
- immediate — принудительно завершает сеансы, при запуске потребуется восстановление.

По умолчанию используется режим fast.

Установка пакета

Готовые пакеты — предпочтительный способ

Linux (ubuntu, Debian, Red Hat, CentOS и другие)

- входит в дистрибутив ОС
- репозиторий (yum, apt) или пакеты RPM, DEB

FreeBSD, OpenBSD

- пакеты из Ports and Packages Collection

Mac OS X

Windows

Предпочтительным вариантом является использование готовых пакетов, так как в этом случае получается понятная, поддерживаемая и легко обновляемая установка.

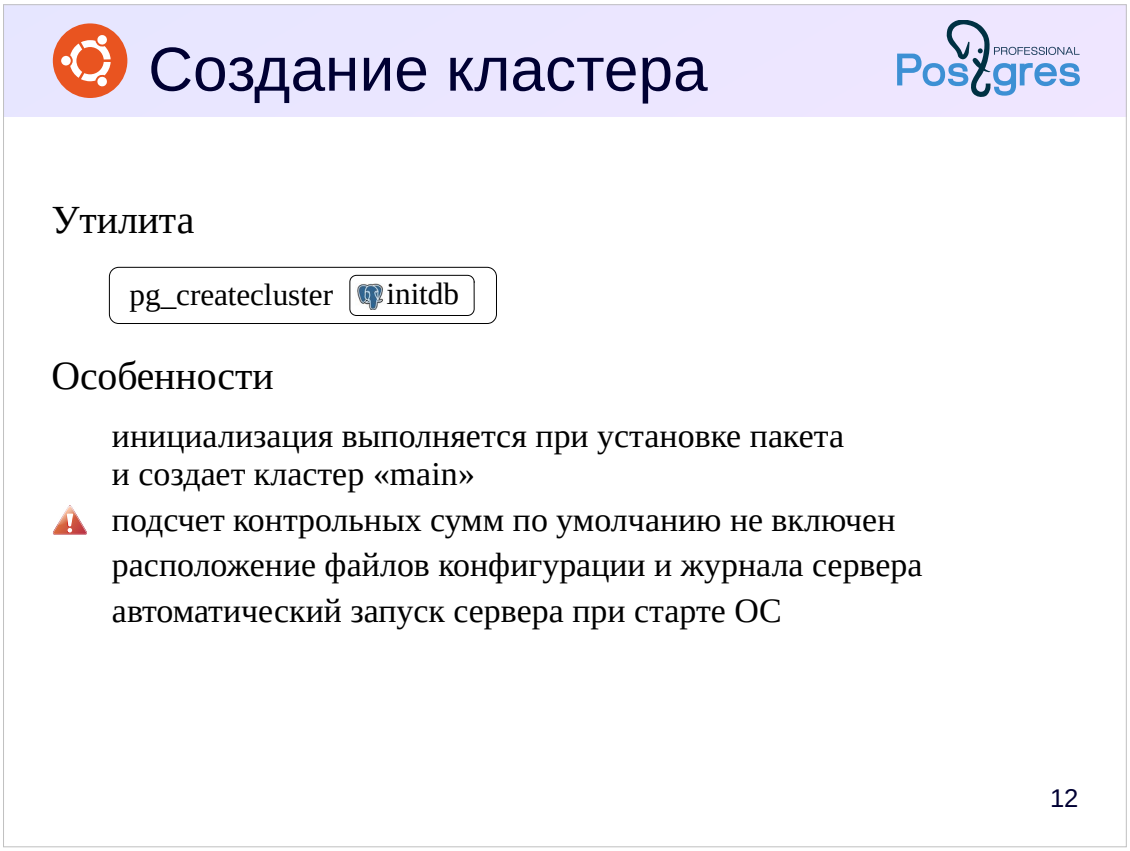
Пакеты существуют для большинства широко распространенных систем. В каждой из них могут быть свои особенности, с которыми стоит ознакомиться перед установкой.

В пакетные репозитории некоторых систем уже входит PostgreSQL, но, обычно, не самой последней версии. Актуальная версия может быть взята из репозитория PostgreSQL Global Development Group (PGDG):

https://www.postgresql.org/download/

(Версии Postgres Pro: https://postgrespro.ru/products/download)

Мы рассмотрим особенности пакета PostgreSQL для Ubuntu.



Для инициализации кластера в Ubuntu создана обертка pg_createcluster над утилитой initdb.

Справку по использованию pg_createcluster можно получить командой:

```
$ man pg_createcluster
```

Утилита pg_createcluster автоматически запускается при установке пакета и создает кластер баз данных с именем «main».

Стоит обратить внимание, что инициализация кластера проходит с отключенным подсчетом контрольных сумм в страницах данных.

Исполняемые файлы, файлы конфигурации и журнал сервера размещены в соответствии с правилами, принятыми в Ubuntu.

Кроме того, настраивается автоматический запуск и останов сервера PostgreSQL при запуске и останове операционной системы.

Для удаления кластера используется утилита pg_dropcluster.

Утилиты pg_createcluster и pg_dropcluster специфичны для Ubuntu. В других системах нужно явно инициализировать кластер с помощью initdb и задействовать подходящие средства операционной системы.

Создание кластера в Ubuntu

В виртуальной машине курса PostgreSQL установлен из пакета командой:

```
student$ sudo apt install -y postgresql-13
```

Каталог установки PostgreSQL:

```
student$ sudo ls -l /usr/lib/postgresql/13
```

```
total 8
drwxr-xr-x 2 root root 4096 апр 29 13:14 bin
drwxr-xr-x 4 root root 4096 апр 29 13:14 lib
```

Владелец ПО сервера — пользователь root.

Утилита pg_config показывает, с какими параметрами был собран сервер:

```
student$ sudo /usr/lib/postgresql/13/bin/pg_config --configure
```

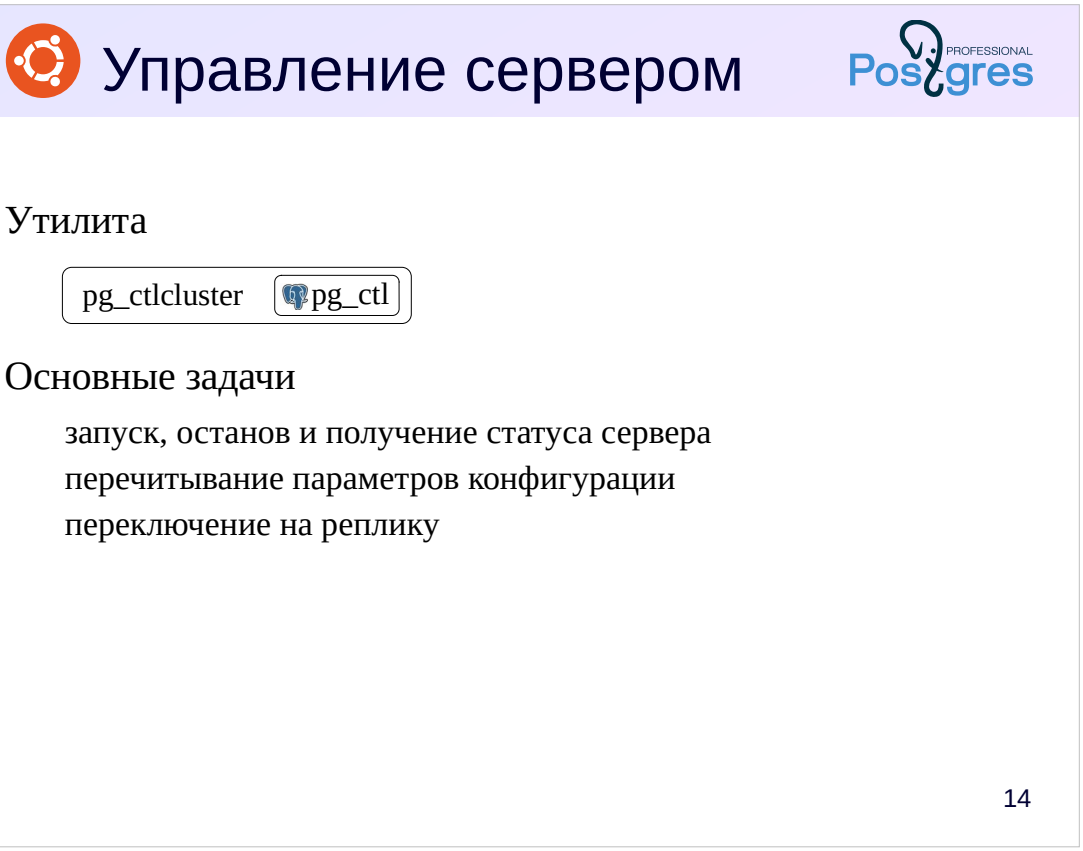
```
 '--build=x86_64-linux-gnu' '--prefix=/usr' '--includedir=${prefix}/include' '--mandir=${prefix}/share/man' '--infodir=${prefix}/share/info' '--sysconfdir=/etc' '--localstatedir=/var'
```

Кластер баз данных с именем main уже инициализирован и находится в каталоге /var/lib/postgresql/13/main.

Владельцем каталога является пользователь postgres. Вот содержимое каталога:

```
student$ sudo ls -l /var/lib/postgresql/13/main
```

```
total 84
drwx------ 6 postgres postgres 4096 мая  6 14:39 base
drwx------ 2 postgres postgres 4096 мая  6 14:39 global
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_commit_ts
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_dynshmem
drwx------ 4 postgres postgres 4096 мая  6 14:39 pg_logical
drwx------ 4 postgres postgres 4096 мая  6 14:39 pg_multixact
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_notify
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_replslot
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_serial
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_snapshots
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_stat
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_stat_tmp
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_subtrans
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_tblspc
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_twophase
-rw------- 1 postgres postgres    3 мая  6 14:39 PG_VERSION
drwx------ 3 postgres postgres 4096 мая  6 14:39 pg_wal
drwx------ 2 postgres postgres 4096 мая  6 14:39 pg_xact
-rw------- 1 postgres postgres   88 мая  6 14:39 postgresql.auto.conf
-rw------- 1 postgres postgres  130 мая  6 14:39 postmaster.opts
-rw------- 1 postgres postgres  108 мая  6 14:39 postmaster.pid
```



В пакетном дистрибутиве для Ubuntu доступ к утилите pg_ctl осуществляется не напрямую, а через специальную обертку pg_ctlcluster. Справку по использованию pg_ctlcluster можно получить командой:

```
$ man pg_ctlcluster
```

В других системах утилита pg_ctl может использоваться непосредственно.

Управление сервером в Ubuntu

При установке из пакета в настройки запуска ОС добавляется запуск PostgreSQL. Поэтому после загрузки операционной системы отдельно стартовать PostgreSQL не нужно.

Можно явным образом управлять сервером с помощью следующих команд, которые выдаются либо от имени пользователя ОС postgres, либо через sudo:

Остановить сервер:

```
student$ sudo pg_ctlcluster 13 main stop
```

Запустить сервер:

```
student$ sudo pg_ctlcluster 13 main start
```

Перезапустить:

```
student$ sudo pg_ctlcluster 13 main restart
```

Получить текущий статус сервера:

```
student$ sudo pg_ctlcluster 13 main status
```

```
pg_ctl: server is running (PID: 3203)
/usr/lib/postgresql/13/bin/postgres "-D" "/var/lib/postgresql/13/main" "-c" "config_file=/etc/postgresql/13/main/postgresql.conf"
```

Перечитать файлы конфигурации:

```
student$ sudo pg_ctlcluster 13 main reload
```

Утилита pg_ctlcluster позволяет управлять несколькими серверами разных версий. В качестве параметров утилите передаются:

- 13 — номер версии сервера;
- main — название сервера.

Журнал сообщений сервера при пакетной установке находится здесь:

```
student$ ls -l /var/log/postgresql/postgresql-13-main.log
```

```
-rw-r----- 1 postgres adm 2747 мая  6 14:40 /var/log/postgresql/postgresql-13-main.log
```

Заглянем в конец журнала:

```
student$ tail -n 10 /var/log/postgresql/postgresql-13-main.log
```

```
2022-05-06 14:40:03.754 MSK [3154] LOG:  aborting any active transactions
2022-05-06 14:40:03.767 MSK [3154] LOG:  background worker "logical replication launcher" (PID 3161) exited with exit code 1
2022-05-06 14:40:03.768 MSK [3156] LOG:  shutting down
2022-05-06 14:40:03.888 MSK [3154] LOG:  database system is shut down
2022-05-06 14:40:04.074 MSK [3203] LOG:  starting PostgreSQL 13.6 (Ubuntu 13.6-1.pgdg22.04+1+b1) on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 11.2.0-16ubuntu1) 11.2.0, 64-bit
2022-05-06 14:40:04.075 MSK [3203] LOG:  listening on IPv4 address "127.0.0.1", port 5432
2022-05-06 14:40:04.086 MSK [3203] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
2022-05-06 14:40:04.116 MSK [3204] LOG:  database system was shut down at 2022-05-06 14:40:03 MSK
2022-05-06 14:40:04.163 MSK [3203] LOG:  database system is ready to accept connections
2022-05-06 14:40:06.503 MSK [3203] LOG:  received SIGHUP, reloading configuration files
```

Облачные решения

PostgreSQL в виртуальной среде

- накладные расходы на виртуализацию
- администратор имеет полный доступ к экземпляру

PostgreSQL as a Service

- предлагается многими облачными провайдерами
- основное администрирование берет на себя провайдер
- доступ только к ограниченным инструментам для управления, резервного копирования, мониторинга

Средства виртуализации могут использоваться и для баз данных, в том числе PostgreSQL. Надо понимать, что в обмен за удобства этой технологии приходится расплачиваться накладными расходами. Для высоконагруженных систем любые дополнительные промежуточные слои между СУБД и железом нежелательны.

Кроме того, многие ведущие зарубежные и российские облачные провайдеры предлагают использовать PostgreSQL как сервис (Database as a Service, managed database).

В этом случае провайдер берет на себя большинство задач администрирования. Полного доступа к экземпляру в этом случае нет, а возможности управления ограничиваются предоставленными инструментами для таких задач, как мониторинг производительности или резервное копирование и восстановление.

В рамках курса мы не будем рассматривать особенности отдельных облачных решений. Для этого стоит изучить документацию, предоставляемую провайдерами услуг.

Итоги

Два варианта установки — пакеты или исходный код

Доступны облачные решения

Сервер выполняется под выделенным пользователем ОС

Перед использованием сервера необходимо инициализировать кластер баз данных

Специальные команды для управления сервером

Практика

Включение расчета контрольных сумм в кластере.

1. Остановите сервер.
2. Проверьте, рассчитываются ли контрольные суммы в кластере.
3. Включите расчет контрольных сумм.
4. Запустите сервер.

2, 3. Используйте утилиту pg_checksums. Для ее запуска укажите полный путь: `/usr/lib/postgresql/13/bin/pg_checksums`

https://postgrespro.ru/docs/postgresql/13/app-pgchecksums

Практика+

1. Установите PostgreSQL из исходных кодов так, как это показано в демонстрации.

   Создайте кластер баз данных, запустите сервер.

   Убедитесь, что сервер работает.

   Остановите сервер.

1. Останов сервера

```
student$ sudo pg_ctlcluster 13 main stop
```

2. Проверка

Чтобы узнать, включен ли расчет контрольных сумм страниц, запустим утилиту pg_checksums с ключом --check:

```
student$ sudo /usr/lib/postgresql/13/bin/pg_checksums --check -D /var/lib/postgresql/13/main
```

```
pg_checksums: error: data checksums are not enabled in cluster
```

3. Включение расчета контрольных сумм

Выполняем pg_checksums с ключом --enable:

```
student$ sudo /usr/lib/postgresql/13/bin/pg_checksums --enable -D /var/lib/postgresql/13/main
```

```
Checksum operation completed
Files scanned:  1494
Blocks scanned: 4941
pg_checksums: syncing data directory
pg_checksums: updating control file
Checksums enabled in cluster
```

Расчет контрольных сумм включен.

4. Запуск сервера

```
student$ sudo pg_ctlcluster 13 main start
```

Практика+

Установка PostgreSQL из исходных кодов.

1. Соберите PostgreSQL без расширений и установите его в домашнем каталоге student на порту 5555.
2. Создайте кластер баз данных, запустите сервер.
3. Убедитесь, что сервер работает.
4. Остановите сервер.

1. Установку PostgreSQL выполняем под пользователем ОС student. Файл postgresql-13.6.tar.gz с исходными кодами находится в домашнем каталоге.

Установите сервер в подкаталог pgsql13 в домашнем каталоге student. Для этого создайте каталог pgsql13 и укажите его в ключе --prefix утилиты configure. А в ключе --with-pgport используйте 5555.

2. Инициализируйте кластер в каталоге /home/student/pgsql13/data.

Добавьте ключ -k для включения подсчета контрольных сумм и укажите ключ -U postgres для указания имени суперпользователя СУБД.

3. Для старта сервера используйте утилиту pg_ctl. В ключе -l укажите имя файла журнала, иначе сообщения сервера будут направляться в окно терминала.

Для проверки работы сервера используйте команду:

psql -U student -d postgres -c ‘SELECT now()’

1. Сборка PostgreSQL из исходных кодов

Все необходимые команды были показаны в демонстрации, но они не выполнялись. Повторите эти действия своими руками в виртуальной машине курса.

Имейте в виду, что компиляция исходных кодов может занимать продолжительное время (от нескольких минут до десятков минут в зависимости от мощности компьютера).