СУБД. PostgreSQL

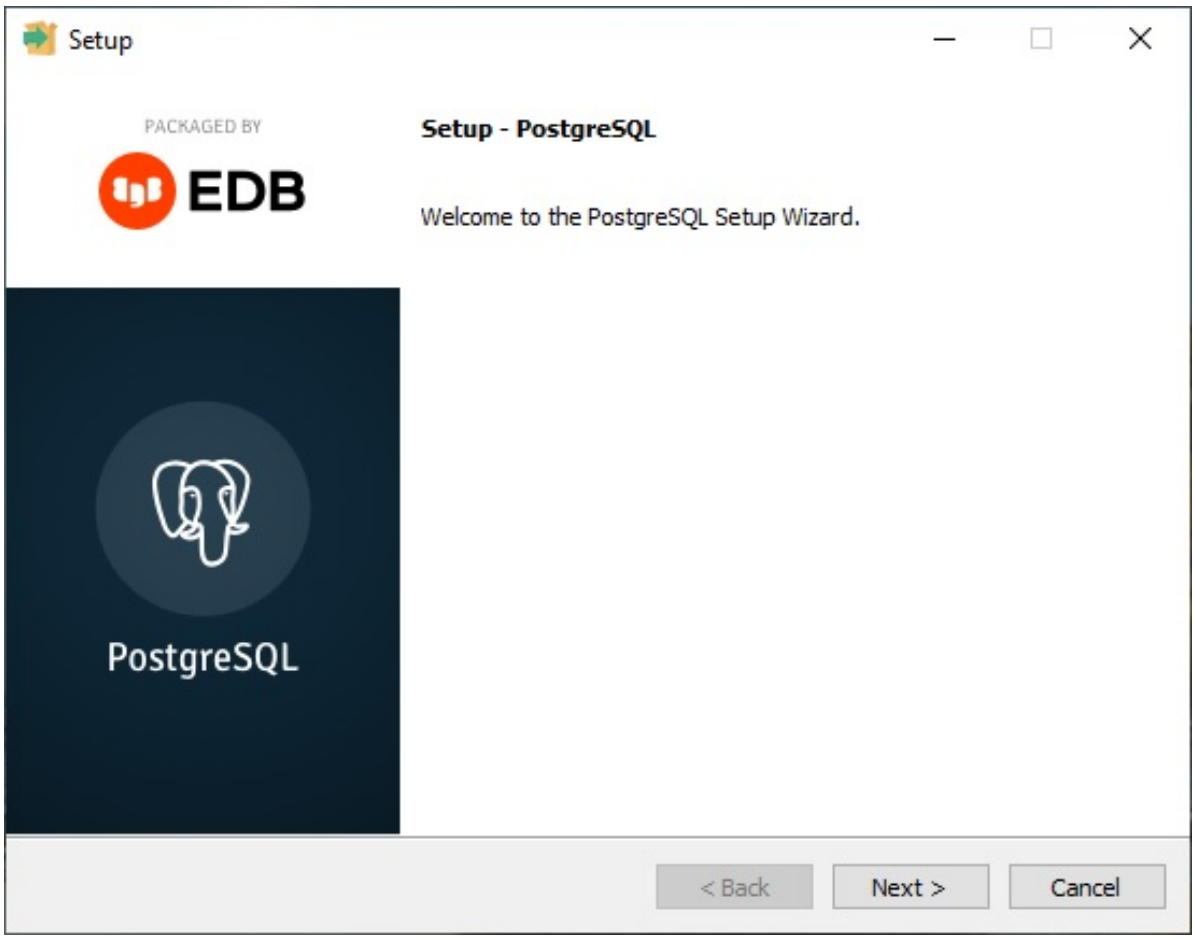
Установка PostgreSQL

https://www.postgresql.org
В учебном курсе «Основы SQL» для демонстрации работы SQL используется PostgreSQL. Сейчас это самая популярная из бесплатных систем управления базами данных. Все SQL запросы в курсе проверены на работоспособность именно в PostgreSQL. Однако большая часть запросов использует синтаксис стандарта ANSI SQL, поэтому они будут работать и в других системах, включая MySQL, Microsoft SQL Server и Oracle. Вы можете использовать любую систему управления базами данных, которая вам нравится, но я рекомендую PostgreSQL.

1. Загрузите PostgreSQL для вашей операционной системы на странице Downloads официального сайта. Я устанавливал на Windows, если вы используете другую операционную систему, то выбирайте соответствующие ссылки для загрузки. Примеры в курсе проверены на **PostgreSQL 13**, поэтому рекомендую устанавливать именно эту версию. Однако на предыдущих версиях, начиная с PosgreSQL 10, также все должно работать.

Инсталлятор для Windows и Mac OS загружается с сайта компании EDB, которая предоставляет платную поддержку для PostgreSQL. Однако PostgreSQL, которую вы установите с помощью этого инсталлятора от EDB, будет полностью бесплатной.

2. Запустите скачанный инсталлятор PostgreSQL.



Основы архитектуры

Прежде чем продолжить, вы должны разобраться в основах архитектуры системы PostgreSQL. Составив картину взаимодействия частей PostgreSQL, вы сможете лучше понять материал этой главы.

Говоря техническим языком, PostgreSQL реализован в архитектуре клиент-сервер. Рабочий сеанс PostgreSQL включает следующие взаимодействующие процессы (программы):

- Главный серверный процесс, управляющий файлами баз данных, принимающий подключения клиентских приложений и выполняющий различные запросы клиентов к базам данных. Эта программа сервера БД называется `postgres`.
- Клиентское приложение пользователя, желающее выполнять операции в базе данных. Клиентские приложения могут быть очень

разнообразными: это может быть текстовая утилита, графическое приложение, веб-сервер, использующий базу данных для отображения веб-страниц, или специализированный инструмент для обслуживания БД. Некоторые клиентские приложения поставляются в составе дистрибутива PostgreSQL, однако большинство создают сторонние разработчики.

Как и в других типичных клиент-серверных приложениях, клиент и сервер могут располагаться на разных компьютерах. В этом случае они взаимодействуют по сети TCP/IP. Важно не забывать это и понимать, что файлы, доступные на клиентском компьютере, могут быть недоступны (или доступны только под другим именем) на компьютере-сервере.

Сервер PostgreSQL может обслуживать одновременно несколько подключений клиентов. Для этого он запускает («порождает») отдельный процесс для каждого подключения. Можно сказать, что клиент и серверный процесс общаются, не затрагивая главный процесс `postgres`. Таким образом, главный серверный процесс всегда работает и ожидает подключения клиентов, принимая которые, он организует взаимодействие клиента и отдельного серверного процесса. (Конечно всё это происходит незаметно для пользователя, а эта схема рассматривается здесь только для понимания.)

Создание базы данных

Первое, как можно проверить, есть ли у вас доступ к серверу баз данных, — это попытаться создать базу данных. Работающий сервер PostgreSQL может управлять множеством баз данных, что позволяет создавать отдельные базы данных для разных проектов и пользователей.

Возможно, ваш администратор уже создал базу данных для вас. В этом случае вы можете пропустить этот этап и перейти к следующему разделу.

Для создания базы данных, в этом примере названной `mydb`, выполните следующую команду:

```
$ createdb mydb
```

Если вы не увидите никаких сообщений, значит операция была выполнена успешно и продолжение этого раздела можно пропустить.

Если вы видите сообщение типа:

```
createdb: command not found
```

значит PostgreSQL не был установлен правильно. Либо он не установлен вообще, либо в путь поиска команд оболочки не включён его каталог. Попробуйте вызвать ту же команду, указав абсолютный путь:

```
$ /usr/local/pgsql/bin/createdb mydb
```

У вас этот путь может быть другим. Свяжитесь с вашим администратором или проверьте, как были выполнены инструкции по установке, чтобы исправить ситуацию.

Ещё один возможный ответ:

```
createdb: ошибка: не удалось подключиться к серверу через сокет
"/tmp/.s.PGSQL.5432":
No such file or directory
        Он действительно работает локально и принимает
        соединения через этот сокет?
```

Это означает, что сервер не работает или `createdb` не может к нему подключиться. И в этом случае пересмотрите инструкции по установке или обратитесь к администратору.

Также вы можете получить сообщение:

```
createdb: ошибка: не удалось подключиться к серверу через
сокет"/tmp/.s.PGSQL.5432":ВАЖНО:  роль "joe" не существует
```

где фигурирует ваше имя пользователя. Это говорит о том, что администратор не создал учётную запись PostgreSQL для вас. (Учётные записи PostgreSQL отличаются от учётных записей пользователей операционной системы.) Если вы сами являетесь администратором, прочитайте Главу 22, где написано, как создавать учётные записи. Для создания нового пользователя вы должны стать пользователем операционной системы, под именем которого был установлен PostgreSQL (обычно это `postgres`). Также возможно, что вам назначено имя пользователя PostgreSQL, не совпадающее с вашим именем в ОС; в этом случае вам нужно явно указать ваше имя пользователя PostgreSQL, используя ключ `-U` или установив переменную окружения `PGUSER`.

Если у вас есть учётная запись пользователя, но нет прав на создание базы данных, вы увидите сообщение:

```
createdb: ошибка: создать базу данных не удалось:
  ОШИБКА:  нет прав на создание базы данных
```

Создавать базы данных разрешено не всем пользователям. Если PostgreSQL отказывается создавать базы данных для вас, значит вам необходимо соответствующее разрешение. В этом случае обратитесь к вашему администратору. Если вы устанавливали PostgreSQL сами, то для целей этого введения вы должны войти в систему с именем пользователя, запускающего сервер БД. [1]

Вы также можете создавать базы данных с другими именами. PostgreSQL позволяет создавать сколько угодно баз данных. Имена баз данных должны начинаться с буквы и быть не длиннее 63 символов. В качестве имени базы данных удобно использовать ваше текущее имя пользователя. Многие утилиты предполагают такое имя по умолчанию, так что вы сможете упростить ввод команд. Чтобы создать базу данных с таким именем, просто введите:

```
$ createdb
```

Если вы больше не хотите использовать вашу базу данных, вы можете удалить её. Например, если вы владелец (создатель) базы данных `mydb`, вы можете уничтожить её, выполнив следующую команду:

```
$ dropdb mydb
```

(Эта команда не считает именем БД по умолчанию имя текущего пользователя, вы должны явно указать его.) В результате будут физически удалены все файлы, связанные с базой данных, и так как отменить это действие нельзя, не выполняйте его, не подумав о последствиях.

Узнать о командах `createdb` и `dropdb` больше можно в справке createdb и dropdb.

---

[1] Объяснить это поведение можно так: Учётные записи пользователей PostgreSQL отличаются от учётных записей операционной системы. При подключении к базе данных вы можете указать, с каким именем пользователя PostgreSQL нужно подключаться. По умолчанию же используется имя, с которым вы зарегистрированы в операционной системе. При этом получается, что в PostgreSQL всегда есть учётная запись с именем, совпадающим с именем системного пользователя, запускающего сервер, и к тому же этот пользователь всегда имеет права на создание баз данных. И

чтобы подключиться с именем этого пользователя PostgreSQL, необязательно входить с этим именем в систему, достаточно везде передавать его с параметром `-U`.

Подключение к базе данных

Создав базу данных, вы можете обратиться к ней:

- Запустив терминальную программу PostgreSQL под названием *psql*, в которой можно интерактивно вводить, редактировать и выполнять команды SQL.
- Используя существующие графические инструменты, например, pgAdmin или офисный пакет с поддержкой ODBC или JDBC, позволяющий создавать и управлять базой данных. Эти возможности здесь не рассматриваются.
- Написав собственное приложение, используя один из множества доступных языковых интерфейсов. Подробнее это рассматривается в Части IV.

Чтобы работать с примерами этого введения, начните с `psql`. Подключиться с его помощью к базе данных `mydb` можно, введя команду:

```
$ psql mydb
```

Если имя базы данных не указать, она будет выбрана по имени пользователя. Об этом уже рассказывалось в предыдущем разделе, посвящённом команде `createdb`.

В `psql` вы увидите следующее сообщение:

```
psql (16.1)
Type "help" for help.

mydb=>
```

Последняя строка может выглядеть и так:

```
mydb=#
```

Что показывает, что вы являетесь суперпользователем, и так скорее всего будет, если вы устанавливали экземпляр PostgreSQL сами. В этом случае на вас не будут распространяться никакие ограничения доступа, но для целей данного введения это не важно.

Если вы столкнулись с проблемами при запуске `psql`, вернитесь к предыдущему разделу. Команды `createdb` и `psql` подключаются к серверу одинаково, так что если первая работает, должна работать и вторая.

Последняя строка в выводе `psql` — это приглашение, которое показывает, что `psql` ждёт ваших команд и вы можете вводить SQL-запросы в рабочей среде `psql`. Попробуйте эти команды:

```
mydb=> SELECT version();
                                                 version
-------------------------------------------------------------------------
----------------------------
 PostgreSQL 16.1 on x86_64-pc-linux-gnu, compiled by gcc (Debian
4.9.2-10) 4.9.2, 64-bit
(1 row)

mydb=> SELECT current_date;
    date
------------
 2016-01-07
(1 row)

mydb=> SELECT 2 + 2;
 ?column?
----------
        4
(1 row)
```

В программе `psql` есть множество внутренних команд, которые не являются SQL-операторами. Они начинаются с обратной косой черты, «`\`». Например, вы можете получить справку по различным SQL-командам PostgreSQL, введя:

```
mydb=> \h
```

Чтобы выйти из `psql`, введите:

```
mydb=> \q
```

и `psql` завершит свою работу, а вы вернётесь в командную оболочку операционной системы. (Чтобы узнать о внутренних командах, введите `\?` в приглашении командной строки `psql`.) Все возможности `psql` документированы в справке psql. В этом руководстве мы не будем использовать эти возможности явно, но вы можете изучить их и применять при удобном случае.