PostgreSQL

О PostgreSQL

- PostgreSQL - это мощная система объектно-реляционных баз данных с открытым исходным кодом, которая использует и расширяет язык SQL в сочетании со многими функциями, которые безопасно хранят и масштабируют самые сложные рабочие нагрузки данных.

Зачем использовать PostgreSQL

- Помимо того , что PostgreSQL является бесплатным и открытым исходным кодом, он обладает широкими возможностями расширения. Например, вы можете определять свои собственные типы данных, создавать собственные функции, даже писать код на разных языках программирования без перекомпиляции базы данных.

Зачем использовать PostgreSQL

- Начиная с версии 12, выпущенной в октябре 2019 года, PostgreSQL соответствует как минимум 160 из 179 обязательных функций для соответствия SQL: 2016 Core. На момент написания этой презентации ни одна реляционная база данных не соответствовала этому стандарту.

Функции, доступные в PostgreSQL

Типы данных

- Примитивы: целое, числовое, строковое, логическое
- Структурированные: дата / время, массив, диапазон, UUID
- Документ: JSON / JSONB, XML, Key-value (Hstore)
- Геометрия: точка, линия, круг, многоугольник
- Настройки: составные, пользовательские типы

Функции, доступные в PostgreSQL

Целостность данных

- УНИКАЛЬНЫЙ, НЕ НУЛЬ
- Основные ключи
- Иностранные ключи
- Ограничения исключения
- Явные Замки, Консультативные Замки

Функции, доступные в PostgreSQL

Параллельность, Производительность

- Индексирование: B-дерево, Многоколонка, Выражения, Частичное
- Расширенная индексация: GiST, SP-Gist, KNN Gist, GIN, BRIN, индексы покрытия, фильтры Блума
- Сложный планировщик запросов / оптимизатор, сканирование только по индексу, многоколоночная статистика
- Транзакции, вложенные транзакции (через точки сохранения)
- Мульти-версия управления параллелизмом (MVCC)
- Распараллеливание запросов на чтение и построение индексов B-дерева
- Разделение таблицы
- Все уровни изоляции транзакций, определенные в стандарте SQL, включая Serializable
- JIT-компиляция выражений Just-in-time

Функции, доступные в PostgreSQL

Надежность, аварийное восстановление

- Запись с опережением записи (WAL)
- Репликация: асинхронная, синхронная, логическая
- Восстановление на момент времени (PITR), активные резервы
- Табличные

Функции, доступные в PostgreSQL

Безопасность

- Аутентификация: GSSAPI, SSPI, LDAP, SCRAM-SHA-256, сертификат и многое другое
- Надежная система контроля доступа
- Безопасность на уровне столбцов и строк
- Многофакторная аутентификация с сертификатами и дополнительным методом

Функции, доступные в PostgreSQL

Растяжимость

- Хранимые функции и процедуры
- Процедурные языки: PL / PGSQL, Perl, Python (и многие другие)
- Выражения пути SQL / JSON
- Оболочки сторонних данных: подключайтесь к другим базам данных или потокам со стандартным интерфейсом SQL
- Настраиваемый интерфейс хранения для таблиц
- Множество расширений, обеспечивающих дополнительную функциональность, включая PostGIS

Функции, доступные в PostgreSQL

Интернационализация, поиск текста

- Поддержка международных наборов символов, например, через сопоставления ICU
- Без учета регистра и без учета ударения
- Полнотекстовый поиск

Основы архитектуры

- 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.)

Настройки

Работа с PostgreSQL может быть произведена через командную строку (терминал) с использованием утилиты psql – инструмент командной строки PostgreSQL. Попробуйте ввести

```
psql postgres (для выхода из интерфейса используйте \q)
```

Этой командой вы запустите утилиту psql. Хотя есть много сторонних инструментов для администрирования PostgreSQL, нет необходимости их устанавливать, т. к. psql удобен и отлично работает.

Настройки

Если вам нужна помощь, введите **\help** (или **-h**) в psql-терминале. Появится список всех доступных параметров справки. Вы можете ввести **\help [имя команды]**, если вам нужна помощь по конкретной команде. Например, если ввести **\help UPDATE** в консоли psql, вы увидите синтаксис команды update

```
Description: update rows of a table
[ WITH [ RECURSIVE ] with_query [, ...] ]
UPDATE [ ONLY ] table_name [ * ] [ [ AS ] alias ]
    SET { column_name = { expression | DEFAULT } |
          ( column_name [, ...] ) = ( { expression | DEFAULT } [, ...] ) |
          ( column_name [, ...] ) = ( sub-SELECT )
        } [, ...]
    [ FROM from_list ]
    [ WHERE condition | WHERE CURRENT OF cursor_name ]
    [ RETURNING * | output_expression [ [ AS ] output_name ] [, ...] ]
```

Настройки

Если у вас возникает много вопросов – не стоит отчаиваться. Поиск в интернете предоставит массу примеров, ну и официальную документацию psql никто не отменял. Первым делом необходимо проверить наличие существующих пользователей и баз данных. Выполните следующую команду, чтобы вывести список всех баз данных:

```
\list или \l
```

```
postgres=# \l
                                  List of databases
   Name    |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
-----------+----------+----------+-------------+-------------+-----------------------
 postgres  | postgres | UTF8     | ru_RU.UTF-8 | ru_RU.UTF-8 |
 template0 | postgres | UTF8     | ru_RU.UTF-8 | ru_RU.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
 template1 | postgres | UTF8     | ru_RU.UTF-8 | ru_RU.UTF-8 | =c/postgres          +
           |          |          |             |             | postgres=CTc/postgres
(3 rows)

postgres=#
```

Работа с PostgreSQL

Настройки

Чтобы вывести список всех пользователей, выполните команду **\du**. Атрибуты пользователя postgres говорят нам, что он суперпользователь.

```
postgres=# \du
                                   List of roles
 Role name |                         Attributes                         | Member of
-----------+------------------------------------------------------------+-----------
 postgres  | Superuser, Create role, Create DB, Replication, Bypass RLS | {}

postgres=#
```

Создание базы данных

Для создания базы данных используется команда **create database**. В приведенном ниже примере создается база данных с именем **proglib_db**:

```
postgres=# create database proglib_db;
CREATE DATABASE
postgres=#
```

Если вы забыли точку с запятой в конце запроса, знак «=» в приглашении postgres заменяется на «-», как показано на рисунке ниже. Это зачастую указывает на то, что необходимо завершить (дописать)

```
postgres=# create database proglib_db
postgres-# ;
ERROR:  database "proglib_db" already exists
postgres=#
```

На картинке нам сообщают об ошибке из-за того, что в нашем случае база уже создана. Вы поймете, что к чему, когда начнете писать более длинные запросы.

Версии PostgreSQL

Всего версий 14. Самая первая — 8.1, самая новая — 12. В этой таблице указано, какие функции по процедурным языкам были добавлены в какую версию.

	12	11	10	9,6	9,5	9,4	9,3	9,2	9,1	9,0	8,4	8,3	8,2	8,1
ДЕЛО в pl / pgsql	да	да	да	да	да	да	да	да	да	да	да	нет	нет	нет
Оператор CONTINUE для PL / pgSQL	да	да	да	да	да	да	да	да	да	да	да	да	да	да
СОЗДАТЬ ТРАНСФОРМ	да	да	да	да	да	нет	нет	нет	нет	нет	нет	нет	нет	нет
DO заявление для pl / perl	да	да	да	да	да	да	да	да	да	да	нет	нет	нет	нет
DO заявление для pl / pgsql	да	да	да	да	да	да	да	да	да	да	нет	нет	нет	нет
ИСКЛЮЧЕНИЕ поддержка в PL / pgSQL	да	да	да	да	да	да	да	да	да	да	да	да	да	да
ВЫПОЛНИТЬ ИСПОЛЬЗОВАНИЕ в PL / pgSQL	да	да	да	да	да	да	да	да	да	да	да	нет	нет	нет
FOREACH IN ARRAY в pl / pgsql	да	да	да	да	да	да	да	да	да	нет	нет	нет	нет	нет
Параметры IN / OUT / INOUT для pl / pgsql и PL / SQL	да	да	да	да	да	да	да	да	да	да	да	да	да	да
Именованные параметры	да	да	да	да	да	да	да	да	да	да	да	да	да	да
Создание языка не суперпользователя	да	да	да	да	да	да	да	да	да	да	да	да	нет	нет
Количество аргументов функции увеличено до 100	да	да	да	да	да	да	да	да	да	да	да	да	да	да
pl / pgsql установлен по умолчанию	да	да	да	да	да	да	да	да	да	да	нет	нет	нет	нет
Полиморфные функции	да	да	да	да	да	да	да	да	да	да	да	да	да	да
Поддержка Python 3 для pl / python	да	да	да	да	да	да	да	да	да	да	нет	нет	нет	нет
Параметры квалифицированной функции	да	да	да	да	да	да	да	да	да	да	да	да	нет	нет
ВОЗВРАТИТЕ ЗАПРОС в pl / pgsql	да	да	да	да	да	да	да	да	да	да	да	да	нет	нет
Спецификация ROWS и COST для функций	да	да	да	да	да	да	да	да	да	да	да	да	нет	нет
Поддержка прокручиваемого и обновляемого курсора для pl / pgsql	да	да	да	да	да	да	да	да	да	да	да	да	нет	нет
SQLERRM / SQLSTATE для pl / pgsql	да	да	да	да	да	да	да	да	да	да	да	да	да	да
Поддержка объектов Unicode в PL / python	да	да	да	да	да	да	да	да	да	да	нет	нет	нет	нет
Пользовательские исключения	да	да	да	да	да	да	да	да	да	да	да	нет	нет	нет
Функция валидатора для pl / perl	да	да	да	да	да	да	да	да	да	да	да	да	да	да