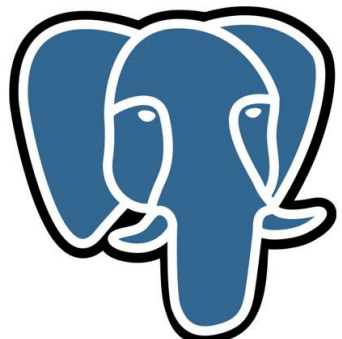
РАБОТА В СУБД POSTGRESQL



PostgreSQL

СУБД обеспечивает работу приложений с информационной моделью вне зависимости от того, какая прикладная программа или же какой пользователь работает с данными, единым образом защищает данные от рассогласованности, оптимизирует выполнение операций над данными, оптимизирует обращение к данным, и др.

СУБД (система управления базой данных) – компьютерная программа для управления данными и доступа к ним

ФУНКЦИИ

- Поддержка логической модели данных (определение данных, оперирование данными)
- Восстановление данных (транзакции, журнализация, контрольные точки)
- Управление одновременным доступом к данным в БД
- Безопасность данных (права доступа и прочее)
- Оптимизация выполнения операций и др.

С У Б Д

Oracle

MS SQL Server

SyBase

MySQL

MS Access

PostgreSQL

Firebird

...

Состав СУБД

СУБД состоит из

- ядра, находящегося в памяти
- сервера
- неограниченного количества программ клиентов, выполняющих поставленные пользователями задачи

О PostgreSQL

PostgreSQL — наиболее полнофункциональная, свободно распространяемая СУБД с открытым кодом

Современный PostgreSQL ведет происхождение от проекта **POSTGRES**, который разрабатывался под руководство
Майкла Стоунбрейкер
Калифорний
университет
Беркли

Ассоциация вычислительной техники (англ. Association for Computing Machinery, ACM) наградила премией Тьюринга Майкла Стоунбрейкера (Michael Stonebraker), создателя

Работа над проектом началась в 1985 году, и до 1988 года был опубликован ряд научных статей, описывающих модель данных, язык запросов POSTQUEL (в то время SQL еще не был общепризнанным стандартом) и устройство хранилища данных

Ingres (1973)
Postgres (1985)
Illustra (1997, Informix, IBM)
Mariposa / Cohera (2001, Peoplesoft, Oracle)
Aurora / Streambase (2003)
C-Store / Vertica (2005)
Morpheus / Goby (2006)
H-Store / VoltDB (2007)
Sci-DB (2008)

Из истории

Первая версия СУБД была выпущена в 1989 году

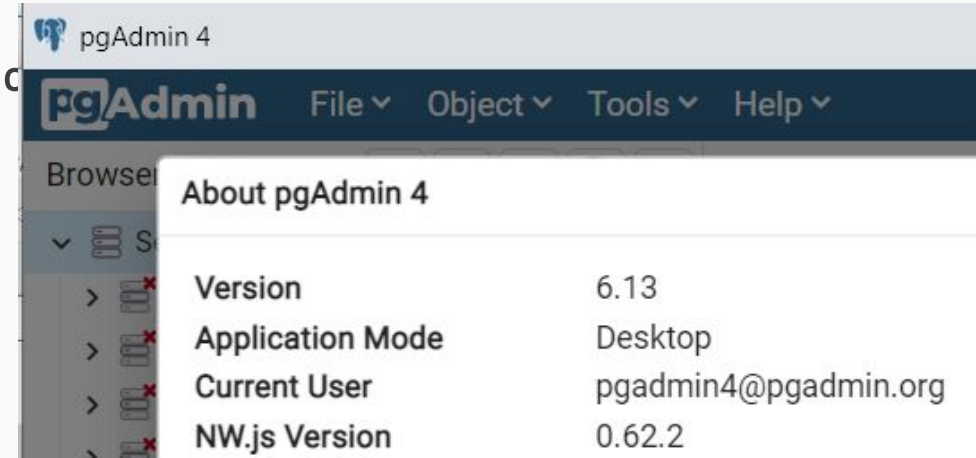
База данных совершенствовалась на протяжении нескольких лет, а в 1993 году, когда вышла версия 4.2, проект был закрыт

Но, несмотря на официальное прекращение, открытый код и BSD-лицензия позволили выпускникам Беркли Эндрю Ю и Джоли Чену в 1994 году взяться за его дальнейшее развитие. Они заменили язык запросов *POSTQUEL* на ставший к тому времени общепринятым *SQL*, а

не выдержит испытание временем, и было выбрано новое имя — **PostgreSQL**, которое отражает связь и с оригинальным проектом POSTGRES, и с переходом на SQL

Новая версия стартовала как 6.0, продолжая исходную нумерацию, управление проектом взяла на себя сначала небольшая группа инициативных пользователей и разработчиков, которая получила название **Глобальной группы разработки** PostgreSQL (PostgreSQL Global Development Group)

Вклад российских разработчиков в PostgreSQL весьма значителен - это, пожалуй, самый крупный глобальный проект с открытым исходным кодом с таким широким представительс



Компания

История

В 2015 году Олег Бартунов, астроном и научный сотрудник ГАИШ МГУ, вместе с **Федором Сигаевым** и **Александром Коротковым** основали компанию Postgres Professional для развития отечественной экспертизы в области разработки систем баз данных и создания российской СУБД

Олег Бартунов

Генеральный директор Postgres Professional

Олег имеет статус PostgreSQL Major Contributor. В 1982 г. он окончил физический факультет МГУ, работает в ГАИШ МГУ. Профессиональный астроном. Изначально использовал PostgreSQL для решения задач астрономии, с 1996 г. участвует в разработке СУБД и продвигает PostgreSQL в России. Создатель крупнейшего астрономического сайта astronet.ru. Занимался разработкой портала rambler.ru. Совместно с Фёдором Сигаевым разработал для PostgreSQL систему полнотекстового поиска, средства поддержки слабоструктурированных данных, индексные методы доступа, в том числе к пространственным данным, а также разнообразные расширения для СУБД.

Федор Сигаев

Технический директор Postgres Professional Разработка.

Федор имеет статусы PostgreSQL Major Contributor, PostgreSQL Committer и FreeBSD Contributor. В 1996 г. он окончил физический факультет МГУ. Работал в одной из первых веб-студий страны «Махаон». После этого работал в компаниях Rambler, Stack Group, Mail.ru. С 2000 г. — разработчик СУБД PostgreSQL. Совместно с Олегом Бартуновым разработал для PostgreSQL систему полнотекстового поиска, средства поддержки слабоструктурированных данных, индексные методы доступа, в том числе к пространственным данным, и другой функционал данной платформы.

Немного истории

Цикл работы над очередной версией PostgreSQL обычно занимает **около года**

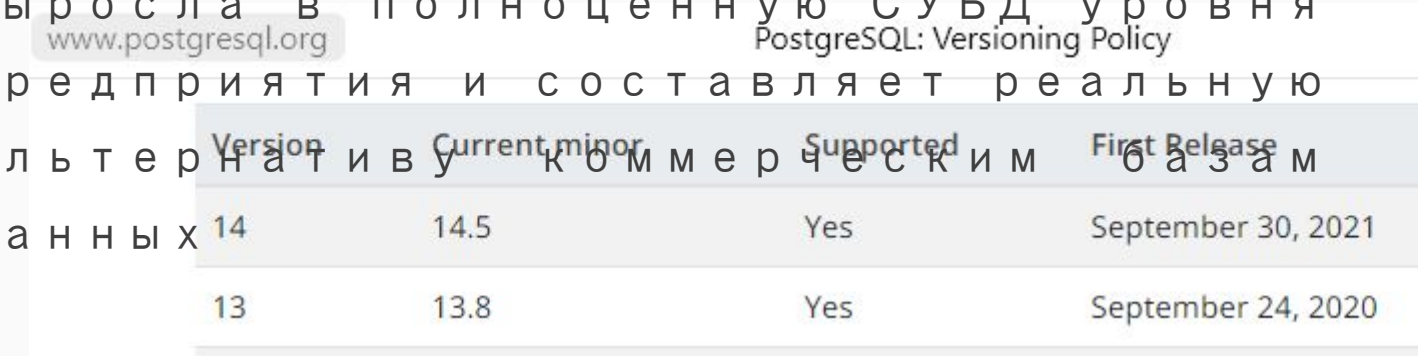
За это время от всех желающих принимаются на рассмотрение патчи с исправлениями, изменениями и новым функционалом

Раньше номер основной версии состоял из двух чисел, но, начиная с 2017 года, было решено оставить только одно.

Таким образом, за 9.6 последовала 10, а последней актуальной версией PostgreSQL является версия 14, вышедшая в

Глобальная группа разработки PostgreSQL выполняет поддержку основных версий системы в течение пяти лет с момента выпуска. Эта поддержка, как и координация разработки, осуществляется через списки рассылок

PostgreSQL является одной из самых популярных баз данных, за свою более чем 20-летнюю историю развития на прочном фундаменте, заложенном академической разработкой, PostgreSQL выросла в полноценную СУБД уровня предприятия и составляет реальную альтернативу коммерческим базам данных



Характеристики СУБД PostgreSQL

Надёжность и отказоустойчивость	Горячее резервирование, восстановление на заданный момент времени в прошлом, репликация данных
Безопасность	Подключение по защищённому каналу, аутентификация (по паролю, с помощью внешних сервисов), управление пользователями и доступом (роли, разграничение прав доступа)
Соответствие стандартам	Высокий уровень соответствия стандарту ANSI SQL и поддержка 170 из 177 обязательных возможностей
Поддержка транзакционности	Полная поддержка свойств ACID, изоляция транзакций (используется механизм многоверсионного управления одновременным доступом MVCC), который позволяет обходиться без блокировок строк

Характеристики СУБД PostgreSQL

Масштабируемость и производительность	Параллельное выполнение запросов чтения и соединения, ряда служебных программ. JIT компиляция запросов (перевод в МК во время выполнения запросов), возможности репликации для горизонтального масштабирования
Планировщик запросов	Основан на стоимости запроса, использует собираемую статистику (дисковые операции, время работы процессора), планировщик оптимизирует сложные запросы, применяя все методы доступа к данным и способы выполнения соединений
Возможности индексирования	Hash индекс, GiST (сбалансирование дерево поиска), SP-GiST (обобщённое несбалан.дерево), и др., также существует возможность объединения сразу несколько индексов для ускорения доступа
Кросс-платформенность	PostgreSQL работает на операционных системах семейства Unix (Linux, FreeBSD, macOS), а также на Windows

Характеристики СУБД PostgreSQL

Расширяемость	Пользователи могут самостоятельно, не меняя базовый код системы добавлять типы данных, функции, индексные и табличные методы доступа, подключения к внешним источникам, загружаемые расширения
Доступность	Неограниченное использование СУБД, модификация кода, включение в состав других продуктов, в том числе закрытых и коммерческих
Независимость	PostgreSQL не принадлежит ни одной компании и развивается международным сообществом, в том числе и российскими разработчиками

Возможности разработчика

Встроенный PL/pgSQL, C, Perl, Python, Java, JavaScript

Программные интерфейсы для обращения к СУБД из приложений (ODBC, JDBC)

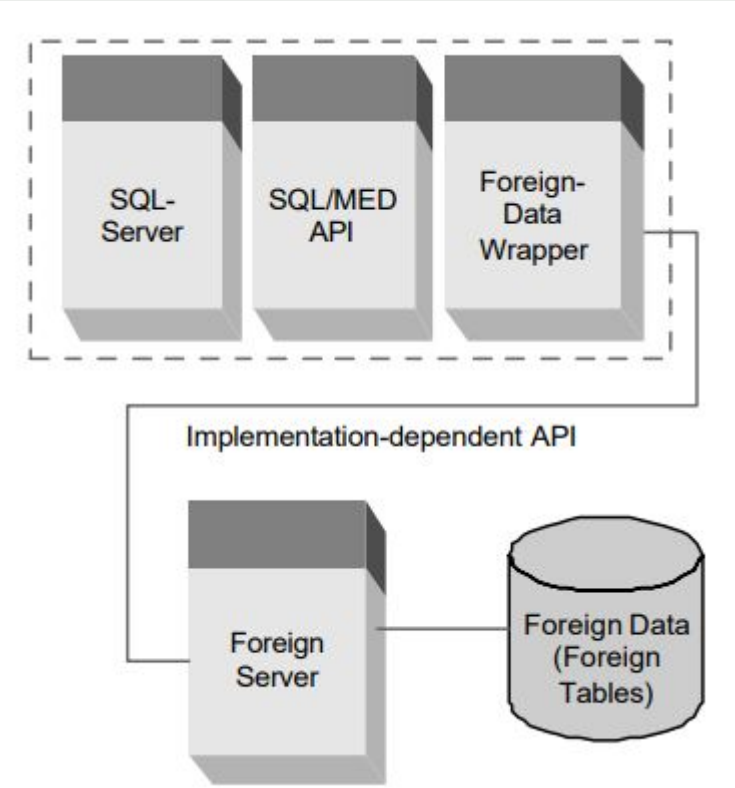
Объекты БД для реализации логики (последовательности, индексы, представления (+материализованные), подзапросы и with-запросы
Агрегатные и оконные функции

Хранимый код – процедуры, функции, триггеры

Слабоструктурированные данные в духе NoSQL, хранилище-ключ-значение xml, json и др.

Подключение источников данных, включая все основные СУБД в качестве внешних таблиц по стандарты SQL/MED

Стандарт SQL/MED



SQL/MED, или **Management of External Data** (управление внешними данными) — расширение стандарта SQL, закреплённое в ISO/IEC 9075-9:2003

SQL/MED реализует расширение SQL за счёт определения адаптеров внешних данных и ссылочных типов, позволяющим SQL управлять внешними данными

Под внешними данными понимаются данные, доступ к которым осуществляется SQL-ориентированными СУБД, но не управляется ими

Этот стандарт может использоваться при разработке федеративных баз данных

SQL стал результатом исследовательского проекта компании IBM, проект включал создание реляционной системы базы данных и языка **SEQUEL** (Structured English Query Language, английский язык структурированных запросов)

SQL (англ. Structured Query Language — язык структурированных запросов) — универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных, в его чистом (базовом) виде является информационно-логическим языком, а не языком программирования

Стандарт языка спецификацией **SQL/PSM** предусматривает возможность его процедурных расширений, с учётом которых язык уже вполне может рассматриваться в качестве языка программирования

История языка SQL

1. Начало 1970-х г. – IBM разрабатывает 1-ю версию языка, первая публикация 1974г. (SEQUEL) для СУБД System/R
2. 1979 г. – завершение исследовательского проекта, язык переименован в SQL
3. 1977 г. фирма Relational Software приступила к созданию промышленной реляционной СУБД на основе SQL, фирма переименовывается в Oracle
4. Середина 70-х – исследовательский проект по созданию экспериментальной СУБД Ingress
5. 1980-1982 гг. компания IBM создаёт СУБД реляционного типа SQL/DS в будущем DB2
6. Середина 80-х гг. – общепринят стандарт для реляционных СУБД

Стандартизация SQL

1982 году Американский национальный институт стандартов (American National Standards Institute — ANSI) создал комитет X3H2, взяв за основу диалект SQL, реализованный в СУБД DB2, комитет постарался его обобщить, учитывая возможности других реляционных СУБД

В 1986 году предложенный комитетом вариант SQL был официально утвержден как стандарт ANSI, в 1987 году был принят в качестве стандарта Международной организацией стандартов (International Standards Organization — ISO)

Стандарт ANSI/ISO приняло правительство США как федеральный стандарт в области обработки информации (Federal Information Processing Standard — FIPS)

В 1989 году стандарт был незначительно изменен и получил название SQL-89 (или SQL1)

В 1992 году ANSI принял стандарт SQL-92 (или SQL2), в котором были расширены способы ограничения целостности, введена поддержка различных языков программирования, предусмотрена обработка транзакций и многое другое

В 1992 году стандарт SQL-99 (или SQL3), SQL стал поддерживать модель данных, выходящую за рамки реляционной, ячейки таблиц могут быть многозначными, что позволяет представлять иерархическую и сетевую модели, язык расширен до возможности представления объектной модели данных и манипулирования ею, SQL стал состоять из отдельных частей (parts), составляющих его основу (foundation), которая дополняется независимо определенными модулями (packages)

Стандартизация SQL

2003	SQL:2003	Введены функции, связанные с XML (SQL/XML), **оконные функции**, стандартизированные **последовательности** и столбцы с **автоматически генерируемыми значениями** (включая столбцы идентификаторов)
2006	SQL:2006	ISO/IEC 9075-14:2006 определяет способы, которыми SQL может использоваться с **XML**, определяет **способы импорта и хранения XML-данных в базе данных SQL**, манипулирования ими в базе данных и публикации как XML, так и обычных SQL-данных в форме XML Позволяет приложениям **интегрировать запросы в свой код SQL** с помощью XQuery, языка запросов XML, опубликованного Консорциумом World Wide Web (W3C), для одновременного доступа к обычным SQL-данным и XML-документам
2008	SQL:2008	Упорядочивает порядок с помощью определений **внешних курсоров**, добавляет оператор **TRUNCATE**, предложение **FETCH**
2011	SQL:2011	Добавляет временные данные (ПЕРИОД ДЛЯ) Улучшения для **оконных функций** и

Возможности SQL

- удаление таблиц базы данных и других ее объектов (доменов, представлений, индексов, триггеров, хранимых процедур, функций и т.д.)
- ✔ указание физической организации данных
- ✔ поддержка ограничений целостности и непротиворечивости базы данных
- ✔ защита данных от несанкционированного доступа посредством определения пользователей (с именами и паролями) и ролей, прав доступа к данным и прав на изменение состояния базы данных
- ✔ манипулирование данными в таблицах базы, включая вставку, изменение и удаление значений
- ✔ поиск данных в нескольких таблицах и упорядочение полученных результатов
- ✔ организация резервного копирования и восстановления базы данных
- ✔ поддержка целостности транзакций
- ✔ поддержка пользовательских процедур и функций, расширяющих функциональные

Категории языка SQL

1. DML (Data Manipulation Language) Работа с данными
2. DDL (Data Definition Language) – работа с объектами БД
3. TCL (Transaction Control Language) работа с транзакциями
4. DCL (Data Control Language) доступ к данным

DDL (Data Definition Language)

Операторы DDL (Data Definition Language) - операторы определения объектов базы данных (CREATE TABLE, ALTER TABLE, DROP TABLE ...)

Эти операторы позволяют создавать, изменять и уничтожать базы данных.

DML (Data Manipulation Language)

Операторы DML (Data Manipulation Language) - операторы манипулирования данными (SELECT, INSERT, UPDATE, DELETE)] выбор, вставка, изменение, удаление данных

TCL (Transaction Control Language)

Операторы TCL (Transaction Control Language) - применяется для управления изменениями, защиты и управления данными.

Операторы этой группы Commit - завершение транзакции и сохранение изменений, Rollback - откат транзакции, отмена изменений, Transaction - Установка параметров доступа к данным текущей транзакции

DCL (Data Control Language или Access Control Language)

Операторы DCL (Data Control Language или Access Control Language) применяются для осуществления административных функций: Grant - присвоение привилегий, Revoke - отмена привилегий

Клиент для PostgreSQL

Клиенты СУБД

1. Консольный клиент psql

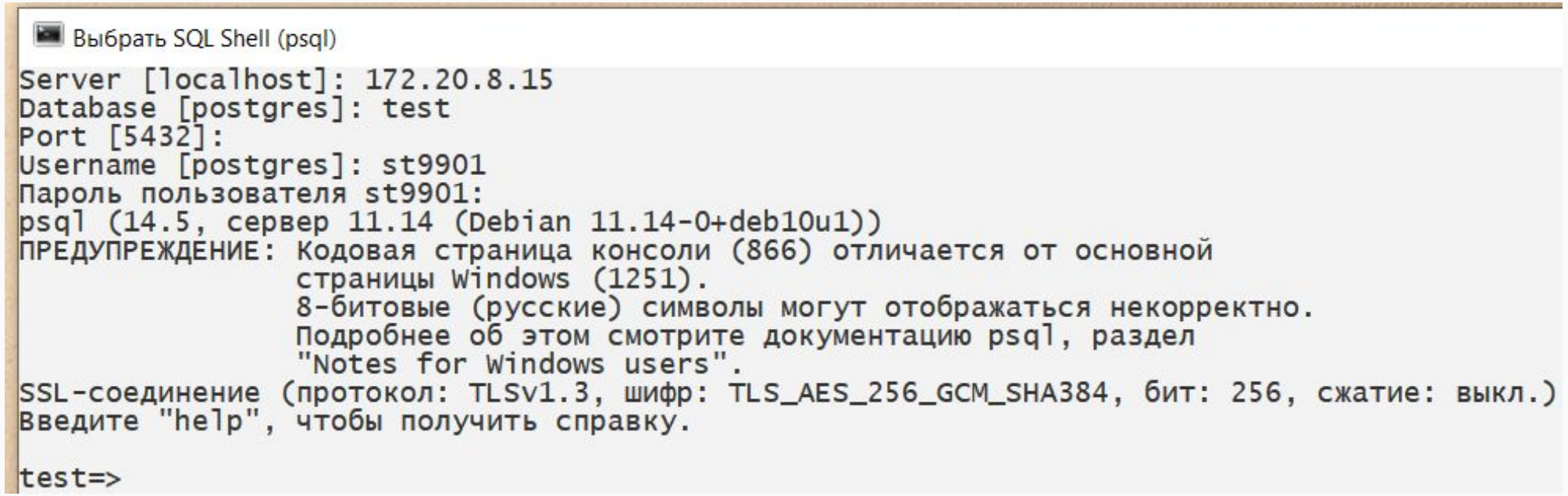
2. Графические клиенты

✔ pgAdmin

✔ Dbeaver

- macOS GUI Clients
  - 1.1 Open Source and Completely Free Software
  - 1.2 Proprietary
    - 1.2.1 SQLPro for Postgres
    - 1.2.2 TablePlus
    - 1.2.3 Postico
    - 1.2.4 PSequel (free)
    - 1.2.5 SEQUEL for PostgreSQL
- Windows GUI Clients
  - 2.1 Open Source / Free Software
    - 2.1.1 HeidiSQL
  - 2.2 Proprietary
    - 2.2.1 dbForge Studio for PostgreSQL
    - 2.2.2 Database Tour Pro
    - 2.2.3 SQLGate
    - 2.2.4 Datazenit
    - 2.2.5 EMS SQL Manager for PostgreSQL Freeware
    - 2.2.6 PostgreSQL Maestro
    - 2.2.7 Nucleon Database Master
    - 2.2.8 SQLPro
    - 2.2.9 SQL Data Analysis
- Cross-platform GUI Clients
  - 3.1 Open Source
    - 3.1.1 Beekeeper Studio
    - 3.1.2 pgAdmin 4
    - 3.1.3 Postbird
    - 3.1.4 Sqlectron
    - 3.1.5 SQuirrel
    - 3.1.6 pgAdmin 3
    - 3.1.7 Tora
    - 3.1.8 SQL Workbench/J

8 лучших GUI клиентов PostgreSQL в 2022 г. (proglib.io)



ПОДКЛЮЧЕНИЕ К БАЗЕ ДАННЫХ TEST

```
test=> create database db;
CREATE DATABASE
test=> \c db
psql (14.5, сервер 11.14 (Debian 11.14-0+deb10u1))
SSL-соединение (протокол: TLSv1.3, шифр: TLS_AES_256_GCM_SHA384, бит: 256, сжатие: выкл.)
Вы подключены к базе данных "db" как пользователь "st9901".
```

СОЗДАНИЕ БАЗЫ ДАННЫХ И ПОДКЛЮЧЕНИЕ К НЕЙ

Подключение к другой БД

Не забудьте про **точку с запятой** в конце команды — пока PostgreSQL не увидит этот символ, он будет считать, что вы продолжаете ввод (так что команду можно разбить на несколько строк)

/c имя_БД

```
test=> \c st9901_01
psql (14.5, сервер 11.14 (Debian 11.14-0+deb10u1))
SSL-соединение (протокол: TLSv1.3, шифр: TLS_AES_256_GCM_SHA384, бит: 256, сжатие: выкл.)
Вы подключены к базе данных "st9901_01" как пользователь "st9901".
st9901_01=>
```

```
test=> \?
General
  \copyright             show PostgreSQL usage and distribution terms
  \crosstabview [COLUMNS] execute query and display results in crosstab
  \errverbose            show most recent error message at maximum verbosity
  \g [(OPTIONS)] [FILE]  execute query (and send results to file or |pipe);
                         \g with no arguments is equivalent to a semicolon
  \gdesc                 describe result of query, without executing it
  \gexec                 execute query, then execute each value in its result
  \gset [PREFIX]         execute query and store results in psql variables
  \gx [(OPTIONS)] [FILE] as \g, but forces expanded output mode
  \q                     quit psql
  \watch [SEC]           execute query every SEC seconds

Help
  \? [commands]          show help on backslash commands
  \? options             show help on psql command-line options
  \? variables           show help on special variables
  \h [NAME]              help on syntax of SQL commands, * for all commands

Query Buffer
  \e [FILE] [LINE]       edit the query buffer (or file) with external editor
  \ef [FUNCNAME [LINE]]  edit function definition with external editor
  \ev [VIEWNAME [LINE]]  edit view definition with external editor
  \p                     show the contents of the query buffer
  \r                     reset (clear) the query buffer
  \w FILE                write query buffer to file
```

СПЕЦИАЛЬНЫЕ КОМАНДЫ, КОТОРЫЕ ПОНИМАЕТ ТОЛЬКО PSQL

Команды с \ не похожи на SQL — она начинается с обратной косой черты – это специальные команды, которые понимает только psql

Справочная информация довольна объемна, показывается с помощью настроенной в операционной системе команды-пейджера (more или less)

Полезные команды psql

\h Справка по SQL: список доступных команд или синтаксис конкретной команды

\x Переключает традиционный табличный вывод (столбцы и строки) на расширенный (каждый столбец на отдельной строке) и обратно

\l Список баз данных

\du Список пользователей

\dt Список таблиц

\di Список индексов

\dv Список представлений

\df Список функций

\dn Список схем

\dx Список установленных расширений

\dp Список привилегий

\d имя Подробная информация по конкретному объекту базы данных. \d+ имя Еще более подробная информация по конкретному объекту

\timing on Показывать время выполнения операторов

\q Завершение сеанса работы

ТАБЛИЦЫ

В реляционных СУБД данные представляются в виде таблиц
Заголовок таблицы определяет столбцы;
собственно данные располагаются в строках
Данные не упорядочены (нельзя полагаться на то, что строки хранятся в порядке их добавления в таблицу)

Типы данных

Для каждого столбца устанавливается тип данных и существует возможность создания новых типов данных

Postgres Pro Standard : Документация: 14: Глава 8. Типы данных : Компания Postgres

Для каждого столбца устанавливается тип данных и существует возможность создания новых типов данных

Помимо обычных значений, определяемых типом данных, поле может иметь неопределенное значение **NULL**

числовой	символьный	дата / время	логический
Значение поля может быть только числом	В этих полях хранятся символьные последовательности (слова, тексты и пр.)	Эти поля предназначены для хранения календарных дат и данных о времени суток Дата: «день / месяц / год» Время: «часы : минуты»	да / нет true / false « 1 » / « 0 »

Числовые типы данных

serial: представляет автоинкрементирующееся числовое значение, которое занимает 4 байта и может хранить числа от 1 до 2147483647. Значение данного типа образуется путем автоинкремента значения предыдущей строки. Поэтому, как правило, данный тип используется для определения идентификаторов строки.

smallserial: представляет автоинкрементирующееся числовое значение, которое занимает 2 байта и может хранить числа от 1 до 32767. Аналог типа `serial` для небольших чисел.

bigserial: представляет автоинкрементирующееся числовое значение, которое занимает 8 байт и может хранить числа от 1 до 9223372036854775807. Аналог типа `serial` для больших чисел.

smallint: хранит числа от -32768 до +32767. Занимает 2 байта. Имеет псевдоним **int2**.

integer: хранит числа от -2147483648 до +2147483647. Занимает 4 байта. Имеет псевдонимы **int** и **int4**.

bigint: хранит числа от -9223372036854775808 до +9223372036854775807. Занимает 8 байт. Имеет псевдоним **int8**.

Числовые типы данных

```
Id SERIAL,
TotalWeight NUMERIC(9,2),
Age INTEGER,
Surplus REAL
```

- **numeric**: хранит числа с фиксированной точностью, которые могут иметь до 131072 знаков в целой части и до 16383 знаков после запятой.
- **decimal**: хранит числа с фиксированной точностью, которые могут иметь до 131072 знаков в целой части и до 16383 знаков в дробной части. То же самое, что и numeric.
- **real**: хранит числа с плавающей точкой из диапазона от 1E-37 до 1E+37. Занимает 4 байта. Имеет псевдоним float4.
- **double precision**: хранит числа с плавающей точкой из диапазона от 1E-307 до 1E+308. Занимает 8 байт. Имеет псевдоним float8.

Типы для работы с валютой (денежными единицами)

Для работы с денежными единицами определен тип **money**, который может принимать значения в диапазоне от -92233720368547758.08 до +92233720368547758.07 и занимает 8 байт

Символьные типы

- **character(n)**: представляет строку из фиксированного количества символов. С помощью параметра задается количество символов в строке. Имеет псевдоним **char(n)**.
- **character varying(n)**: представляет строку из переменной длины. С помощью параметра задается задается максимальное количество символов в строке. Имеет псевдоним **varchar(n)**.
- **text**: представляет текст произвольной длины.

Бинарные данные

Для хранения бинарных данных определен тип **bytea**. Он хранит данные в виде бинарных строк, которые представляют последовательность октетов или байт

Типы для работы с датами и временем

Распространенные форматы дат:

yyyy-mm-dd – 1999-01-08

Month dd, yyyy – January 8, 1999

mm/dd/yyyy – 1/8/1999

Распространенные форматы времени:

hh:mi – 13:21

hh:mi am/pm – 1:21 pm

- **timestamp**: хранит дату и время. Занимает 8 байт. Для дат самое нижнее значение - 4713 г до н.э., самое верхнее значение - 294276 г н.э.
- **timestamp with time zone**: то же самое, что и `timestamp`, только добавляет данные о часовом поясе.

- **date**: представляет дату от 4713 г. до н.э. до 5874897 г н. э. Занимает 4 байта.
- **time**: хранит время с точностью до 1 микросекунды без указания часового пояса. Принимает значения от 00:00:00 до 24:00:00. Занимает 8 байт.

- **time with time zone**: хранит время с точностью до 1 микросекунды с указанием часового пояса. Принимает значения от 00:00:00+1459 до 24:00:00-1459. Занимает 12 байт.

- **interval**: представляет временной интервал. Занимает 16 байт.

Логический тип

Тип **boolean** может хранить одно из двух значений: true или false.

Вместо true можно указывать следующие значения: TRUE, 't', 'true', 'y', 'yes', 'on', '1'.

Вместо false можно указывать следующие значения: FALSE, 'f', 'false', 'n', 'no', 'off', '0'.

Типы для представления интернет-адресов

cidr: интернет-адрес в формате IPv4 и IPv6. Например, 192.168.0.1. Занимает от 7 до 19 байт.

inet: интернет-адрес в формате cidr/y, где cidr это адрес в формате IPv4 или IPv6, а /y - количество бит в адресе (если этот параметр не указан, то используется 34 для IPv4, 128 для IPv6). Например, 192.168.0.1/24 или 2001:4f8:3:ba:2e0:81ff:fe22:d1f1/128. Занимает от 7 до 19 байт.

macaddr: хранит MAC-адрес. Занимает 6 байт.

macaddr8: хранит MAC-адрес в формате EUI-64. Занимает 8 байт.

Геометрические типы

- **point**: представляет точку на плоскости в формате `(x,y)`. Занимает 16 байт.
- **line**: представляет линию неопределенной длины в формате `{A,B,C}`. Занимает 32 байта.
- **lseg**: представляет отрезок в формате `((x1,y1),(x2,y2))`. Занимает 32 байта.
- **box**: представляет прямоугольник в формате `((x1,y1),(x2,y2))`. Занимает 32 байта.
- **path**: представляет набор содиненных точек. В формате `((x1,y1),...)` путь является закрытым (первая и последняя точка соединяются линией) и фактически представляет многоугольник. В формате `[(x1,y1),...]` путь является открытым Занимает 16+16n байт.
- **polygon**: представляет многоугольник в формате `((x1,y1),...)`. Занимает 40+16n байт.
- **circle**: представляет окружность в формате `<(x,y),r>`. Занимает 24 байта.

Другие типы данных

json: хранит данные json в текстовом виде

jsonb: хранит данные json в бинарном формате

uuid: хранит универсальный уникальный идентификатор (UUID), например, a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11. Занимает **32** байта

xml: хранит данные в формате XML

Создание таблиц

```
CREATE TABLE <table-name >
(<column name > <data type>[(<size>)],
<column name > <data type> [(<size>)] ...);
```

Точный синтаксис команды CREATE TABLE можно посмотреть в документации, а можно прямо в psql

```
st9901_01=> create table courses(
st9901_01(> nomer char(4) primary key,
st9901_01(> title text,
st9901_01(> hours integer);
CREATE TABLE
```

Ввод данных

```
INSERT INTO <имя_таблицы> [ (<имя_столбца>,<имя_столбца>,...) ]
                          VALUES (<значение>,<значение>,..)
```

Если вам требуется массовая загрузка данных из внешнего источника можно использовать предназначенную для этого команду COPY

```
st9901_01=> insert into courses (nomer, title, hours)
st9901_01-> values ('9901', 'Базы данных', 30),
st9901_01-> ('9903', 'Операционные системы', 60);
INSERT 0 2
```

ДЗ

1. Написать инструкции для создания таблиц, изменения структуры таблицы (из практического задания №1)
2. Написать выборку с CASE
3. Написать инструкцию для создания таблицы на основе существующих данных

Полезные ссылки

П.Лузанов, Е.Рогов, И.Лёвшин PostGres. Первое знакомство introbook_v8.pdf (postgrespro.ru)

Клиенты PostgreSQL PostgreSQL Clients – PostgreSQL wiki

Postgres Pro Standard : Документация: 14: Глава 8. Типы данных : Компания Postgres Professional

PostgreSQL | Создание и удаление базы данных (metanit.com)