

Лабораторная работа №6

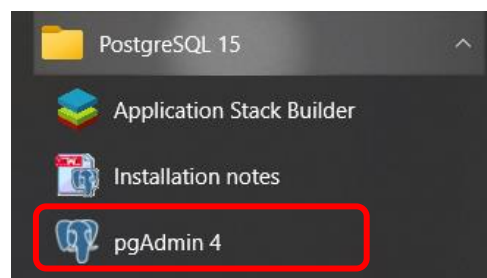
Введение в PostgreSQL. Управление базой данных

PostgreSQL является одной из наиболее популярных объектно-реляционных СУБД, существенный вклад в разработку которой внесли российские разработчики. Компания «Постгрес Професионал» развивает версию СУБД Postgres Pro, входящую в Единый реестр российского ПО. На сайте компании¹ можно скачать дистрибутивы и ознакомиться с процессом установки (*также см. инструкцию в электронном курсе Moodle*).

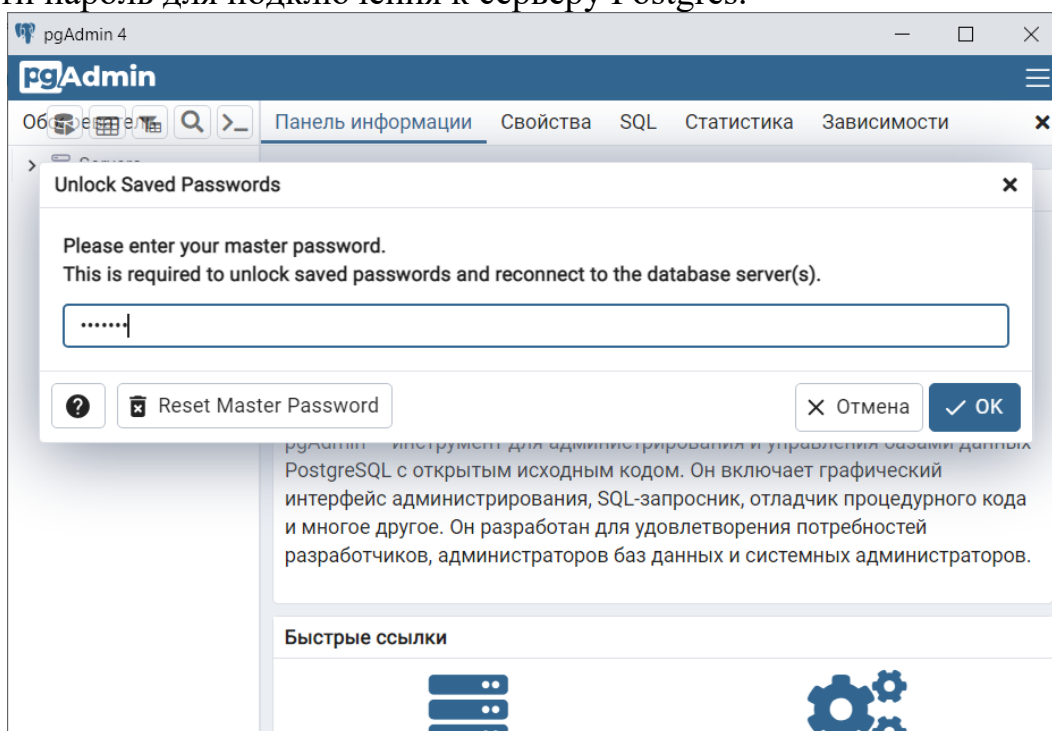
1. Запуск PostgreSQL

1.1. Графический клиент pgAdmin

Для упрощения администрирования на сервере PostgreSQL в базовый комплект установки входит такой инструмент как **pgAdmin**. Он представляет графический клиент для работы с сервером, через который мы в удобном виде можем создавать, удалять, изменять базы данных и управлять ими. Так, на Windows после установки можно найти значок pgAdmin в меню Пуск и запустить его:



При запуске pgAdmin необходимо произвести соединение с сервером, в частности нужно ввести пароль для подключения к серверу Postgres.



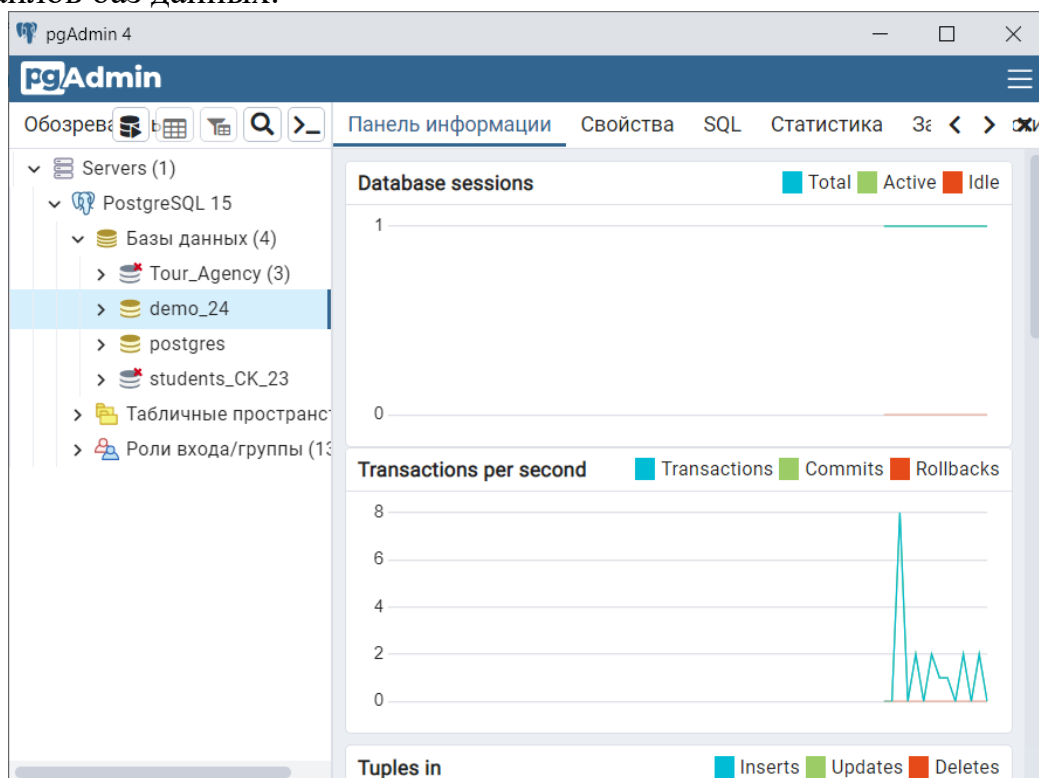
Здесь необходимо ввести пароль для суперпользователя postgres, который был задан при установке PostgreSQL. При установке СУБД в кабинетах 16-го корпуса был задан пароль: **student** (в некоторых кабинетах **Student**).

После успешного соединения в **обозревателе объектов pgAdmin** отображается содержимое сервера. Основные узлы:

- узел **Базы данных (Databases)** отображает все имеющиеся БД. По умолчанию здесь всегда есть база данных – postgres;

¹ <https://postgrespro.ru/windows>

- узел **Роли входа/группы (Login/Group Roles)** предназначен для управления пользователями и их ролями;
- узел **Табличные пространства (Tablespaces)** позволяет управлять местом хранения файлов баз данных.



Задание 1. Запустите графический клиент pgAdmin 4 и произведите соединение с сервером.

Изучите список объектов в **Обозревателе** pgAdmin. Ответьте на вопросы:

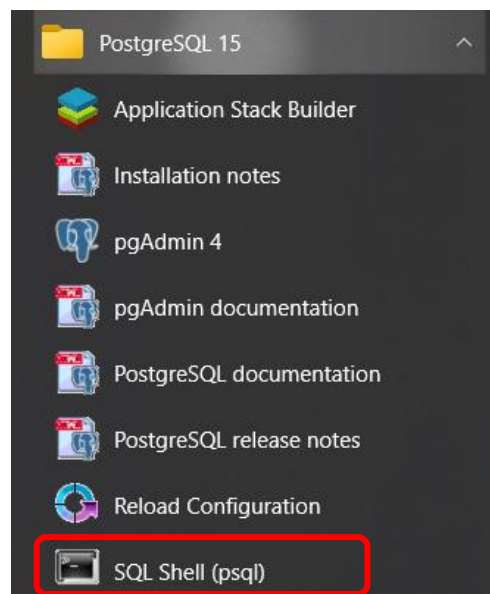
- Какие БД есть на вашем сервере?
- Какие имена для входа зарегистрированы на вашем сервере?
- Какие табличные пространства созданы для вашего сервера?

В отчет вставьте скриншот(ы) обозревателя объектов с раскрытыми основными узлами.

1.2. Консольный клиент psql

Консольный клиент представляет еще один способ взаимодействия с сервером PostgreSQL. Данная программа также, как и pgAdmin, позволяет выполнять команды языка SQL.

После запуска psql программа предложит ввести название сервера, базы данных, порта и пользователя. Эти пункты можно прощелкать, т.к. для них будут использоваться значения по умолчанию (для сервера - localhost, для базы данных - postgres, для порта - 5432, в качестве пользователя - суперпользователь postgres). Далее надо будет ввести пароль для пользователя (по умолчанию пользователя postgres): *student* (в некоторых кабинетах *Student*).



```

Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Пароль пользователя postgres:
psql (15.1)
ПРЕДУПРЕЖДЕНИЕ: Кодовая страница консоли (866) отличается от основной
                    страницы Windows (1251).
                    8-битовые (русские) символы могут отображаться некорректно.
                    Подробнее об этом смотрите документацию psql, раздел
                    "Notes for Windows users".
Введите "help", чтобы получить справку.

postgres=# |

```

Здесь надо ввести пароль
символы пароля НЕ
отображаются

И после удачного подключения можно будет отправлять серверу команды через psql. По умолчанию консоль в Windows поддерживает кодировку CP866, а БД могут работать с другой кодировкой, например, 1251. Если наблюдаются ошибки с кодировкой, то это можно исправить с помощью команды: `psql \! chcp 1251`

```

Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]: postgres
Пароль пользователя postgres:
psql (15.1)
ПРЕДУПРЕЖДЕНИЕ: Кодовая страница консоли (866) отличается от основной
                    страницы Windows (1251).
                    8-битовые (русские) символы могут отображаться некорректно.
                    Подробнее об этом смотрите документацию psql, раздел
                    "Notes for Windows users".
Введите "help", чтобы получить справку.

postgres=#
postgres=# \! chcp 1251
Текущая кодовая страница: 1251
postgres=# SET client_encoding='win1251';
SET

```

Для создания БД применяется команда `create database`, после которой указывается название БД, причем команда завершается точкой с запятой. Например,

```
create database test2;
```

Для осуществления взаимодействия с БД к ней нужно подключиться. Для этого применяется команда `\c` (сокращение от `connect`), после которой указывается имя базы данных. Например,

```
\c test2
```

Далее можно работать с БД. Например,

```

postgres=# create database test2;
CREATE DATABASE
postgres=# \c test2
Вы подключены к базе данных "test2" как пользователь "postgres".
test2=# create table users (Id serial primary key, Name character varying(30),
      Age integer);
CREATE TABLE
test2=# |

```

Создание базы данных test2

Подключение к базе данных test2

Создание таблицы users

```

CREATE TABLE
test2=# insert into users (Name, Age) values ('Tom', 33);
INSERT 0 1
test2=# select * from users;
 id | name | age
---+---+---
  1 | Tom  |  33
(1 ёёЁёёр)
test2=# |

```

Добавление данных

Получение данных

1.3. Развертывание учебной БД (добавление сторонней БД из резервной копии)

Описание учебной БД: https://edu.postgrespro.ru/qpt/qpt_01_demo.pdf

Рассмотрим, как развернуть в вашем кластере PostgreSQL учебную БД «Авиаперевозки», подготовленную компанией Postgres Professional. На сайте компании есть раздел, посвященный этой БД: <https://postgrespro.ru/education/demodb>.

Она предоставляется в трех версиях, отличающихся объемом данных: самая компактная версия содержит данные за один месяц, версия среднего размера – за три месяца, а самая полная версия – за год. Все данные были сгенерированы с помощью специальных алгоритмов, обеспечивающих их «правдоподобность». Мы начнем работу с компактной версией БД «Авиаперевозки».

1 шаг к развертыванию БД: нужно скачать ее заархивированную резервную копию по ссылке https://edu.postgrespro.ru/demo_small.zip. Затем необходимо извлечь файл из архива. Извлеченный файл называется demo_small.sql.

2 шаг: создаем БД с именем demo в нашем кластере PostgreSQL.

```
\i demo_small.sql
```

Если файл не будет найден, можно сменить текущую директорию(папку)

```
\cd 'абсолютный_путь_до_директории_с_demo_small.sql'
```

или сразу использовать команду:

```
\i 'полный_путь_к_файлу_sql'
```

После нажатия клавиши Enter начнется считывание файла, которое займет некоторое время и после завершения в БД должна быть добавлена на ваш сервер.

Например,

```
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Пароль пользователя postgres:
psql (15.1)
ПРЕДУПРЕЖДЕНИЕ: Кодовая страница консоли (866) отличается от основной
страницы Windows (1251).
8-битовые (русские) символы могут отображаться некорректно.
Подробнее об этом смотрите документацию psql, раздел
"Notes for Windows users".
Введите "help", чтобы получить справку.

postgres=# \! chcp 1251
Текущая кодовая страница: 1251
postgres=# \i 'c:/БД/demo_small.sql'
SET
SET
SET
SET
```

Примеры команд для работы с БД.

Пример 1. Подключение (переход) к нужной БД: `\c demo`

Пример 2. Вывод объектов в текущей схеме (по умолчанию текущей схемой будет public): `\d`

Пример 3. Для вывода схем: `\dn`

Пример 4. Вывод всех объектов из всех схем: `\d *.*`

Пример 5. Вывод из конкретной схемы: `\d bookings.*`

Пример 6. Смена текущей схемы: `SET search_path = bookings;`

Пример 7. Теперь `\d` выведет все объекты bookings

Пример 8. Для вывода структур таблиц: `\d+ bookings.*`

Запросы. Объекты pg_ являются системными. psql умеет выводить результат запросов в разных форматах:

1) **Формат с выравниванием значений** (используется по умолчанию). Пример 9:

```
SELECT schemaname, tablename, tableowner FROM pg_tables LIMIT 5;
```

Ширина столбцов выровнена по значениям. Также выводится строка заголовков и итоговая строка.

Команды psql для переключения режима выравнивания:

\a – переключатель режима: с выравниванием/без выравнивания.

\t – переключатель отображения строки заголовка и итоговой строки.

2) **Формат без выравнивания**. Пример 10:

```
\t \a
\pset fieldsep ' '
SELECT schemaname, tablename, tableowner FROM pg_tables LIMIT 5;
\t \a
```

Отключено выравнивание, вывод заголовка и итоговой строки, а в качестве разделителя столбцов сделан пробел.

3) **Расширенный формат** удобен, когда нужно вывести много столбцов для одной или нескольких записей. Пример 11:

```
\x
SELECT * FROM pg_tables WHERE tablename = 'pg_class';
\x
```

Расширенный режим можно установить только для одного запроса, если в конце вместо ";" указать \gx. Пример 12:

```
SELECT * FROM pg_tables WHERE tablename = 'pg_proc' \gx
```

Все возможности форматирования результатов запросов доступны через команду:

```
\pset
```

Задание 2.

1. Запустите консольный клиент psql и произведите соединение с сервером.
2. Настройте корректную работу с кодировкой.
3. Разверните учебную БД **demo** из резервной копии ([см. алгоритм](#)).
4. Выполните команды из примеров 1-8.
5. Ознакомьтесь с форматами вывода результатов, выполнив команды из примеров 9-12. Результаты выполнения всех пунктов задания подтвердите скриншотами.

ВАЖНО: далее работаем в графическом клиенте pgAdmin, в том числе создаем и выполняем SQL-запросы, если иное не указано.

Для однозначного понимания синтаксиса языка SQL здесь и далее будем придерживаться основных соглашений, приведенных в табл.

Соглашение	Назначение
ВЕРХНИЙ РЕГИСТР	Ключевые слова Transact-SQL
[]	Необязательные элементы синтаксиса. Скобки вводить не следует.
{ }	Обязательные элементы, из которых надо выбрать один. Скобки вводить не нужно
	Разделяет элементы внутри [] или { } – обозначает, что может быть выбран только один из элементов, разделенных вертикальной чертой
[,...n]	Предшествующий элемент можно повторить n раз. Отдельные вхождения элемента разделяются запятыми
[...n]	Предшествующий элемент можно повторить n раз. Отдельные вхождения элемента разделяются пробелами

2. Создание базы данных

2.1. Кластер баз данных

При установке PostgreSQL был создан кластер БД, состоящий из трех одинаковых баз.

- БД `template0` – не меняется, используется при создании резервной копии и при создании БД в кодировке, отличной от используемой в кластере.
- БД `template1` – используется для создания новых БД кластера копированием этой БД. В эту БД можно добавить объекты и расширения, которые при копировании попадут во все БД кластера.
- БД `postgres` – необязательная БД, к которой по умолчанию подключается пользователь `postgres`.

Объекты в СУБД можно:

- `CREATE` – создавать;
- `ALTER` – изменять;
- `DROP` – удалять.

2.2. Создание и настройка базы данных

Создание базы данных – это процесс указания имени файла и при необходимости определения других параметров БД.

В pgAdmin предусмотрены средства для создания БД как с помощью кода на языке SQL, так и с помощью графического интерфейса пользователя.

Для создания БД используется команда `CREATE DATABASE`. По умолчанию новая БД создается путем копирования стандартной системной БД `template1`. Для создания БД можно использовать другой шаблон, например, `TEMPLATE template0`, можно создать чистую БД, содержащую только стандартные объекты, предопределенные установленной версией PostgreSQL. Этот прием используется, когда нежелательно копировать в новую БД дополнительные объекты, содержащиеся в шаблоне `template1`.

Оператор **`CREATE DATABASE`**, имеющий следующий формат:

`CREATE DATABASE имя`

```
[[ WITH ] [ OWNER [=] имя_пользователя ]  
[ TEMPLATE [=] шаблон ]  
[ ENCODING [=] кодировка ]  
[ LC_COLLATE [=] категория_сортировки ]  
[ LC_CTYPE [=] категория_типов_символов ]  
[ TABLESPACE [=] табл_пространство ]  
[ ALLOW_CONNECTIONS [=] разр_подключения ]  
[ CONNECTION LIMIT [=] предел_подключений ]  
[ IS_TEMPLATE [=] это_шаблон ]]
```

Параметры:

имя – имя создаваемой БД;

имя_пользователя – имя пользователя (роли), назначаемого владельцем новой БД, либо `DEFAULT`, чтобы владельцем стал пользователь по умолчанию (пользователь, выполняющий команду). Чтобы создать БД и сделать ее владельцем другую роль, необходимо быть непосредственным или опосредованным членом этой роли, либо суперпользователем;

шаблон – имя шаблона, из которого будет создаваться новая БД, либо `DEFAULT`, чтобы выбрать шаблон по умолчанию (`template1`);

кодировка – кодировка символов в новой БД. Укажите строковую константу (например, SQL_ASCII) или целочисленный номер кодировки, либо DEFAULT, чтобы выбрать кодировку по умолчанию (кодировку шаблона);

табл_пространство – имя табличного пространства, связываемого с новой БД, или DEFAULT для использования табличного пространства шаблона. Это табличное пространство будет использоваться по умолчанию для объектов, создаваемых в этой базе. За подробностями обратитесь к CREATE TABLESPACE;

категория_сортировки – порядок сортировки (LC_COLLATE), который будет использоваться в новой БД. Этот параметр определяет порядок сортировки строк, например, в запросах с ORDER BY, а также порядок индексов по текстовым столбцам. По умолчанию используется порядок сортировки, установленный в шаблоне;

категория_типов_символов – классификация символов, которая будет применяться в новой БД. Этот параметр определяет принадлежность символов категориям, например, строчные, заглавные, цифры и т. п. По умолчанию используется классификация символов, установленная в шаблоне;

разр_подключения – если false, никто не сможет подключаться к этой БД. По умолчанию имеет значение true, то есть подключения принимаются (если не ограничиваются другими механизмами, например, GRANT/REVOKE CONNECT);

предел_подключений – максимальное количество одновременных подключений (CONNECTION LIMIT) к этой БД. Значение -1 (по умолчанию) снимает ограничение;

это_шаблон – если true, БД сможет клонировать любой пользователь с правами CREATEDB; в противном случае (по умолчанию), клонировать базу смогут только суперпользователи и ее владелец.

Все описанные параметры, кроме имени БД, являются необязательными.

Пример 13:

```
CREATE DATABASE Test_1;
```

Пример 14:

```
CREATE DATABASE "uchDB"  
WITH OWNER = postgres  
ENCODING = 'UTF8'  
TABLESPACE = pg_default  
LC_COLLATE = 'Russian_Russia.1251'  
LC_CTYPE = 'Russian_Russia.1251'  
CONNECTION LIMIT = -1;
```

Задание 3. Создайте БД из примеров 13 и 14. Убедитесь, что БД созданы. Сравните свойства созданных БД. Сделайте выводы.

Примечание: для открытия **Запросника** необходимо в обозревателе объектов pgAdmin выделить любую БД и выполнить команду Инструменты / **Запросник**.

2.3. Табличное пространство

Табличные пространства позволяют администраторам БД организовать логику размещения файлов объектов БД в файловой системе. При установке СУБД выбирается папка для создания БД, в которой будут физически храниться объекты БД. Это место расположения можно изменить. Для этого используются табличные пространства. Они отражают физическую структуру БД.

При инициализации кластера создается:

- **pg_default** – табличное пространство по умолчанию, в него будут помещаться все создаваемые объекты, если не выбрано другого пространства;

- **pg_global** – содержит объекты системного каталога, общие для всего кластера.

Для просмотра табличных пространств кластера можно использовать команду:

```
SELECT spcname FROM pg_tablespace;
```

Результат выполнения запроса:

	spcname name
1	pg_default
2	pg_global

Создать новое табличное пространство и поменять для БД **Test_1** актуальное табличное пространство на него можно следующими командами:

```
CREATE TABLESPACE ts LOCATION 'D:/sql';
```

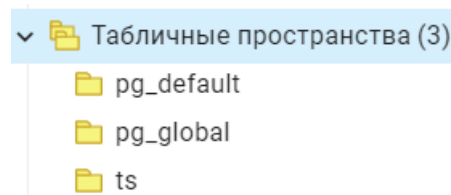
```
ALTER DATABASE Test_1 SET TABLESPACE ts;
```

Табличное пространство отображается и в **Обозревателе объектов** в pgAdmin:

Этого же результата можно достичь, удалив БД и установив табличное пространство при создании.

```
DROP DATABASE Test_1;
```

```
CREATE DATABASE Test_1 TABLESPACE ts;
```



Задание 4.

1. Проверьте какие табличные пространства созданы на вашем сервере.

2. Создайте свое табличное пространство **ts_test** в каталоге 'D:/sql' (обязательно проверьте содержимое папки).

Результаты выполнения задания подтвердите SQL-скриптами и скриншотами.

2.4. Изменение и удаление базы данных

Удаление базы данных осуществляется с помощью оператора **DROP DATABASE**:

```
DROP DATABASE [ IF EXISTS ] имя_БД [ [ WITH ] ( параметр [, ...] ) ]
```

допускается параметр **FORCE**

Параметры:

IF EXISTS - не считать ошибкой, если БД не существует. В этом случае будет выдано замечание.

Имя_БД – Имя базы данных, подлежащей удалению.

FORCE – Попытаться принудительно завершить все существующие подключения к целевой БД. Подключения не завершаются, если в целевой базе имеются подготовленные транзакции, активные слоты логической репликации или подписки.

Команда **DROP DATABASE** удаляет БД. Она удаляет из системного каталога записи, относящиеся к базе, а также удаляет с диска каталог, содержащий данные. Выполнить её может только владелец БД. Кроме того, нельзя удалить БД, к которой вы подключены в данный момент. (Чтобы выполнить эту команду, подключитесь к postgres или любой другой базе данных.) Также команда не будет выполнена, когда к целевой базе подключены какие-то ещё пользователи, если вы не добавите указание **FORCE**.

Изменить атрибуты БД можно с помощью команды **ALTER DATABASE**:

--меняет параметры на уровне БД (только владелец БД или суперпользователь)

```
ALTER DATABASE имя [ [ WITH ] параметр [ ... ] ]
```

Здесь **параметр**:

ALLOW_CONNECTIONS **разр_подключения**
CONNECTION LIMIT **предел_подключений**
IS_TEMPLATE **это_шаблон**

--меняет имя БД. Переименовать БД может только владелец БД
--или суперпользователь. Переименовать текущую БД нельзя.
--Нужно сначала подключиться к другой БД

ALTER DATABASE **имя** RENAME TO **новое_имя**

--меняет владельца базы данных

ALTER DATABASE **имя** OWNER TO { **новый_владелец** | CURRENT_ROLE |
CURRENT_USER | SESSION_USER }

-- меняет табличное пространство по умолчанию для БД. команда физически
-- переносит все таблицы или индексы из прежнего основного табличного
-- пространства БД в новое, которое должно быть пустым для этой БД
-- и к ней никто не должен быть подключён.

ALTER DATABASE **имя** SET TABLESPACE **новое_табл_пространство**

Задание 5.

1. Для ранее созданных БД test_1 и uchDB поменяйте табличное пространство на ts_test. Обязательно проверьте содержимое каталога 'D:/sql'.

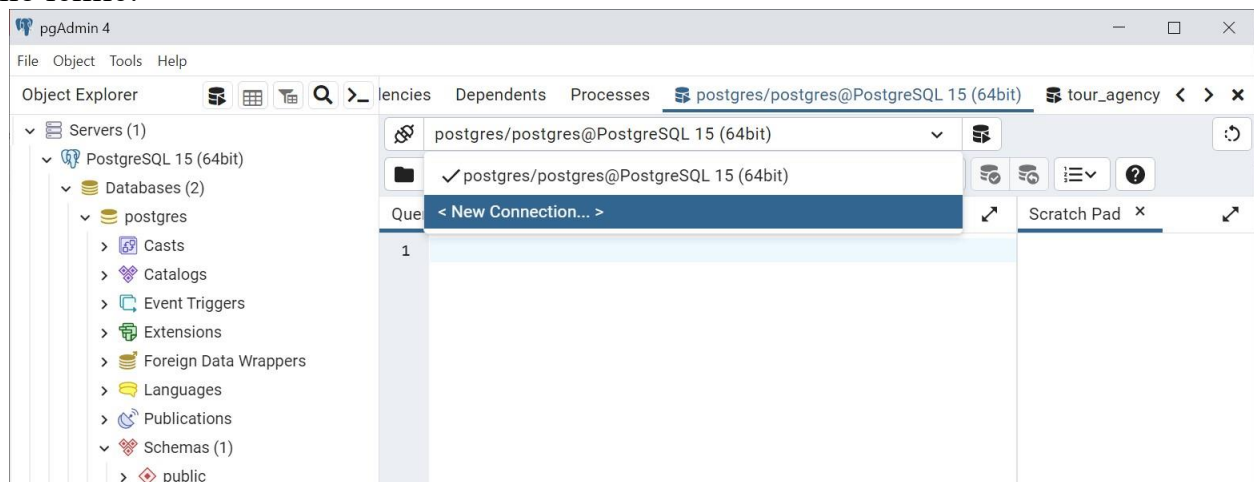
2. Переименуйте БД test_1 в test_FIO (где FIO – это ваши инициалы).

Результаты выполнения задания подтвердите SQL-скриптами и скриншотами.

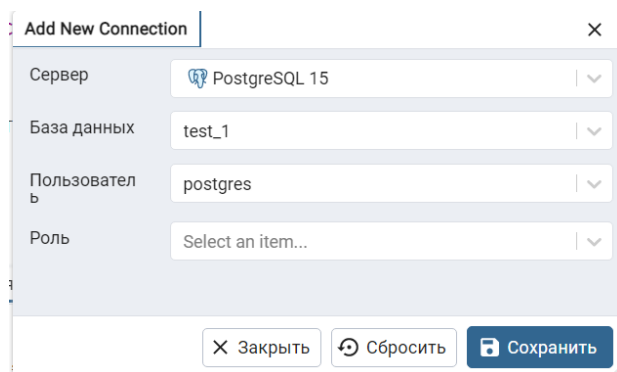
2.5. Схема базы данных

Созданные БД отображаются в обозревателе объектов pgAdmin:

Чтобы объекты создавались в новой БД, к ней нужно подключиться, создав новое подключение:



Выбрать новое соединение, указать к какой БД необходимо подключиться и каким пользователем:



База данных PostgreSQL может содержать одну или несколько именованных схем, которые в свою очередь содержат таблицы, типы данных, представления, функции и другие объекты. Схема БД – это пространство имен, в котором имена объектов могут совпадать с именами других объектов, существующих в других схемах. Для обращения к объекту нужно либо дополнить его имя именем схемы в виде префикса. Пользователь может обращаться к объектам в любой схеме текущей БД, если ему назначены соответствующие права. Кроме перечисленных преимуществ, введение схем позволяет:

- использовать одну БД могут несколько пользователей независимо друг от друга;
- администраторам объединить объекты БД в логические группы.

Создание схемы:

CREATE SCHEMA имя_схемы

[AUTHORIZATION указание_роли] [элемент_схемы [...]]

Здесь «указание_роли» определяется следующим выражением:

[GROUP] имя_пользователя

| CURRENT_USER

| SESSION_USER

имя_схемы – имя создаваемой схемы. Если опущено, то именем схемы будет имя_пользователя. Это имя не может начинаться с pg_, так как такие имена зарезервированы для системных схем;

имя_пользователя – имя пользователя (роли), назначаемого владельцем новой схемы. Если опущено, по умолчанию владельцем будет пользователь, выполняющий команды.

элемент_схемы – команда CREATE SCHEMA может дополнительно включать подкоманды, создающие объекты в новой схеме, т. е. элементы схемы. Эти подкоманды, по сути, воспринимаются как отдельные команды, выполняемые после создания схемы, за исключением того, что с предложением AUTHORIZATION все создаваемые объекты будут принадлежать указанному в нем пользователю. Создать объекты других типов можно отдельными командами после создания схемы.

Перед тем как создавать таблицы в БД, полезно создать схемы. Если этого не сделать, то все создаваемые объекты попадут в схему public. По умолчанию все пользователи имеют право создавать объекты в схеме public (PostgreSQL 15 или ранее), что не совсем хорошо с точки зрения безопасности.

Схемы можно создавать, изменять и удалять.

-- создание схемы с именем Test

CREATE SCHEMA Test;

-- переименование схемы в new_Test

ALTER SCHEMA Test RENAME TO new_Test;

-- удаление схемы new_Test

DROP SCHEMA new_Test;

В кластере баз данных, помимо схемы `public` и схем, созданных пользователем, существуют еще специальные схемы:

`pg_catalog` – системный каталог;

`information_schema` – вариант системного каталога;

`pg_temp` – для временных таблиц.

Список схем можно посмотреть с помощью запроса.

```
SELECT * FROM information_schema.schemata;
```

Результат выполнения представлен на рисунке. Столбец `catalog_name` показывает, в какой БД созданы схемы, `schema_name` – имена схем, `schema_owner` – кто владелец этих схем.

	catalog_name name	schema_name name	schema_owner name	
1	tour_agency	public	pg_database_owner	
2	tour_agency	agency	postgres	
3	tour_agency	information_schema	postgres	
4	tour_agency	pg_catalog	postgres	
5	tour_agency	pg_toast	postgres	

Задание 6.

1. Получите список схем учебной БД `demo` и БД `test_FIO`.
2. В БД `test_FIO` создайте схему `test_1`, в БД `uchDB` – схему `test_2`.
3. В БД `test_FIO` переименуйте схему `test_1` в `schema_test`.
4. В БД `uchDB` удалите схему `test_2`.
5. Удалите БД `uchDB`.

Результаты выполнения задания подтвердите SQL-скриптами и скриншотами.

Задание 7. С помощью скрипта на языке SQL:

1. Создайте табличное пространство `ts_faculty`, которое должно располагаться в каталоге `'D:/SQL/faculty'`.
2. Создайте БД `db_faculty_FIO`². БД должна располагаться в созданном вами табличном пространстве `ts_faculty`.
3. Создайте в БД «ФАКУЛЬТЕТ» схему `Faculty`.

Результаты выполнения задания подтвердите SQL-скриптами и скриншотами.

2.6. Резервное копирование данных

Необходимо уделять особое внимание сохранности информации, с которой работает пользователь. СУБД предлагает **резервное копирование** информации.

Ранее в задании 2 мы уже рассмотрели один из вариантов восстановления базы данных из резервной копии в PostgreSQL.

Операцию резервного копирования и восстановления БД можно совершить также в **Обозревателе** в `pgAdmin`.

Создание резервной копии БД:

1. Вызвать контекстное меню на имени нужной БД, в котором выбрать пункт **Резервная копия...**

2. Откроется диалоговое окно **Backup (База данных: имя_БД)**, которое позволяет задать почти все необходимые параметры резервного копирования.

² В качестве FIO укажите свою фамилию с инициалами или только инициалы

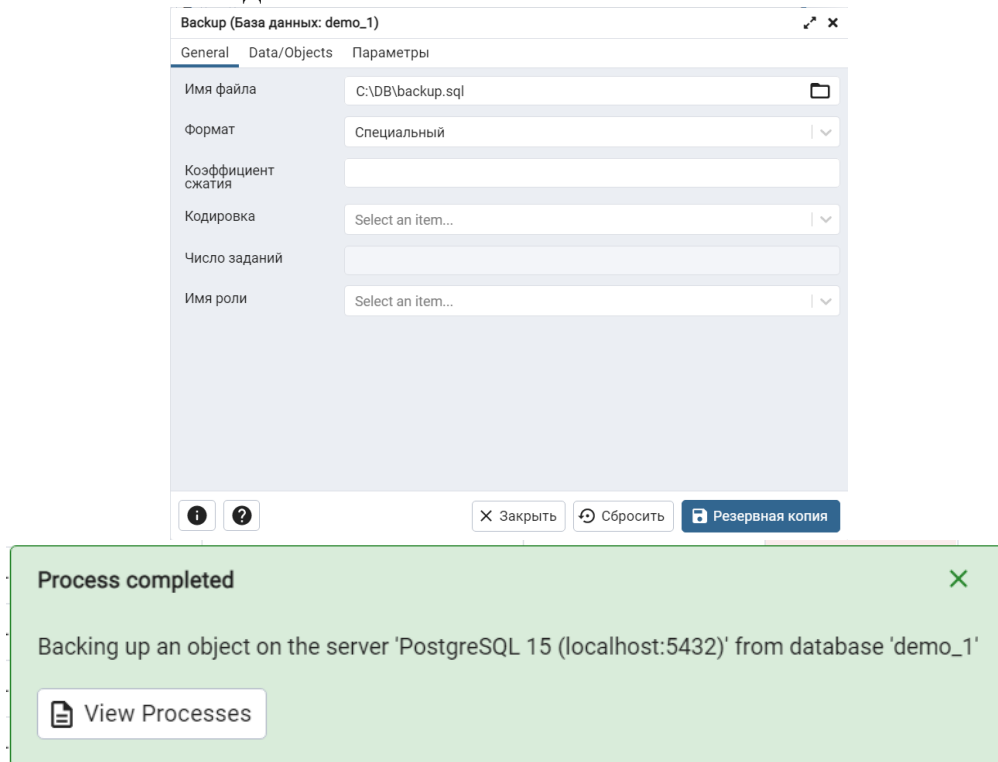
3. На вкладке **General** этого окна (см. рис.) задаются:

- Имя файла (с указанием местоположения);
- Формат и другие.

4. После задания необходимых параметров нажимаем кнопку **Резервная копия**.

5. Ждем появления уведомления об окончании процедуры резервного копирования.

Можно отслеживать на вкладке Processes.



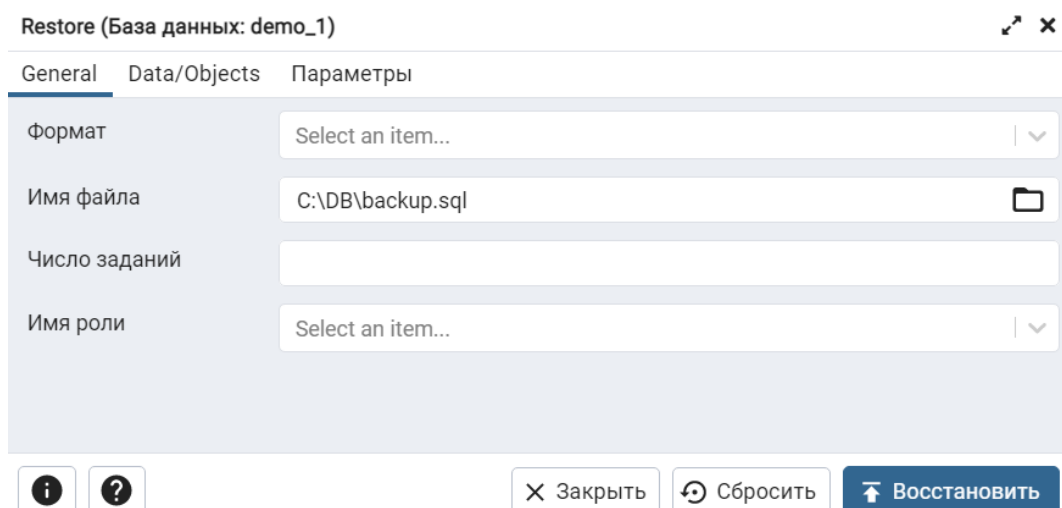
Восстановление БД из резервной копии:

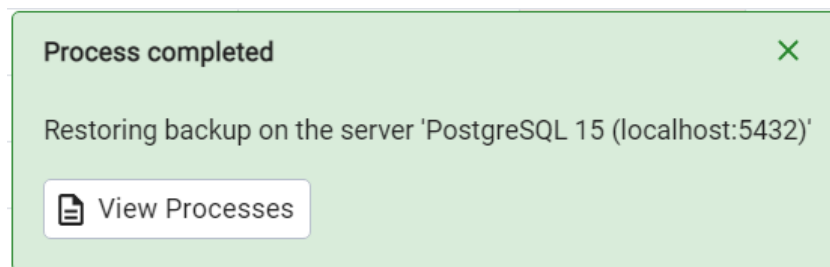
1. Вызвать контекстное меню на имени БД, в которую должна быть восстановлена БД из резервной копии (например, заранее созданную пустую БД), выбрать пункт **Восстановить...**

2. Откроется диалоговое окно **Restore (База данных: имя_БД)**, в котором нужно задать параметры восстановления. На вкладке **General** этого окна (см. рис.) указать файл, из которого будет выполняться восстановление БД.

3. После задания необходимых параметров нажимаем кнопку **Восстановить**.

4. Ждем появления уведомления об окончании процедуры восстановления БД. Можно отслеживать на вкладке Processes.





Задание 8. Создайте резервную копию учебной базы данных **demo** с помощью команд *Обозревателя объектов* в **pgAdmin**. Резервную копию БД разместите в своей папке. Результат выполнения задания подтвердите скриншотами.

Задание 9

1. В БД **test_FIO** с помощью команд обозревателя объектов восстановите БД из резервной копии, созданной вами в задании 8.

2. Дождитесь окончания процедуры восстановления, проверьте результат. Например, просмотрите содержимое нескольких таблиц восстановленной БД.

Результаты выполнения задания подтвердите скриншотами.

3. Удалите базу данных **test_FIO** и табличное пространство **ts_test**, после выполнения п. 1-3 и фиксации результатов