

Лабораторные работы по курсу Базы данных

Лабораторная работа 1 «Знакомство с PostgreSQL»

Москва, 2023

Оглавление

1.	Теоретическая часть	3
1.1.	Основные определения курса	3
1.2.	Общее устройство PostgreSQL	3
1.3.	Работа с <i>pgAdmin</i>	3
1.4.	Язык программирования SQL	5
1.5.	Создание новой роли.	6
1.6.	Создание базы данных.....	7
2.	Практическая часть	10
2.1.	Задание 1.....	10
2.2.	Задание 2.....	10
2.3.	Задание 3.....	10
	Список литературы.....	11
	Приложение.....	12

1. Теоретическая часть

1.1. Основные определения курса

Прежде чем приступить к изучению баз данных, необходимо дать все основные определения.

База данных – совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами моделирования данных.

Система управления базами данных (СУБД) – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

В данном курсе будет рассматриваться объектно-реляционная СУБД **PostgreSQL**. PostgreSQL является одной из самых популярных СУБД на настоящее время. Одной из основных её особенностей является открытый исходный код. [1]

1.2. Общее устройство PostgreSQL

СУБД PostgreSQL представляет собой клиент-серверную архитектуру. Рабочий сеанс PostgreSQL включает следующие взаимодействующие процессы:

- Главный серверный процесс, управляющий файлами баз данных, принимающий подключения клиентских приложений и выполняющий различные запросы клиентов к базам данных.
- Клиентское приложение пользователя, с помощью которого выполняются операции в базе данных.

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. [2]

Взаимодействие между клиентским приложением и сервером происходит посредством запросов. [3]

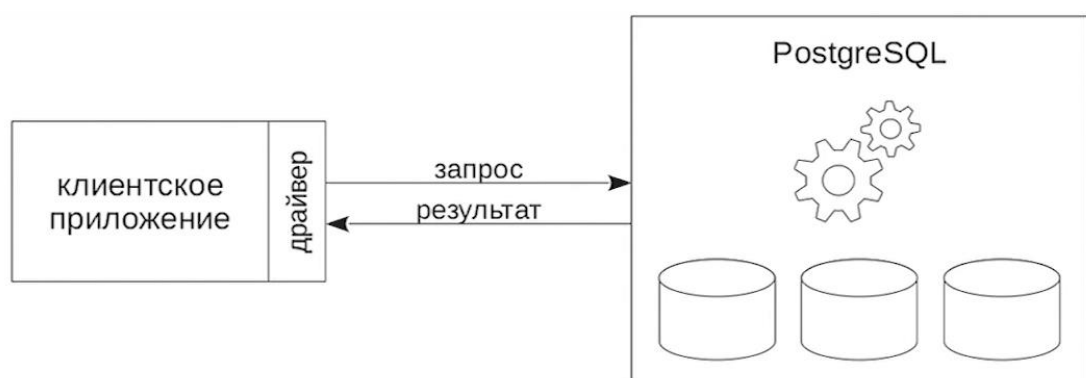
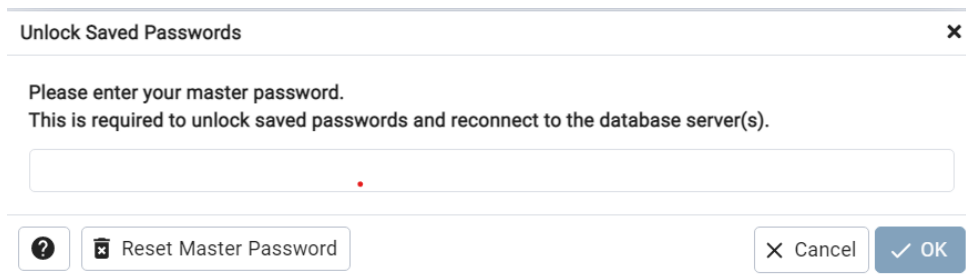


Рисунок 1 Взаимодействие клиента и сервера

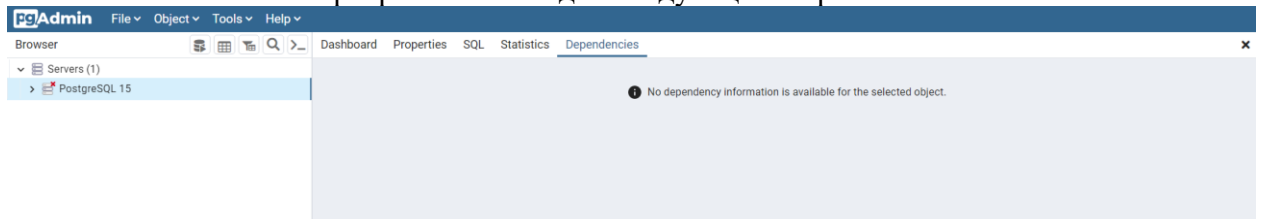
Работу с PostgreSQL возможно осуществить из терминала или с помощью графического приложения. В данном курсе будет рассматриваться инструмент *pgAdmin*.

1.3. Работа с *pgAdmin*

После запуска программы *pgAdmin* вам будет предложено ввести пароль, указанный при установке программы. Если программа была установлена администратором – спросите пароль у преподавателя.



Основное окно программы выглядит следующим образом:



Для подключения к серверу дважды щелкните на название сервера в окошке слева

▼ Servers (1)

> PostgreSQL 15

При удачном подсоединении появляются три новые вкладки

▼ Servers (1)

▼ PostgreSQL 15

> Databases (1)

> Login/Group Roles

> Tablespaces

Первая вкладка – Database содержит всю информацию о хранимых базах данных.

На текущий момент база данных всего одна – **postgres**

▼ Servers (1)

▼ PostgreSQL 15

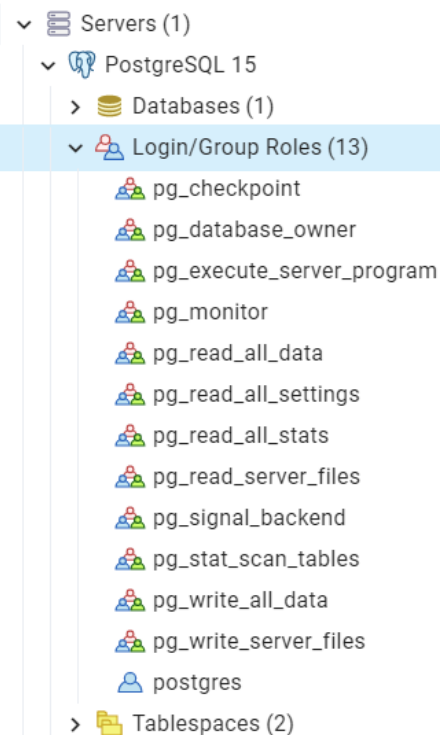
▼ Databases (1)

> postgres

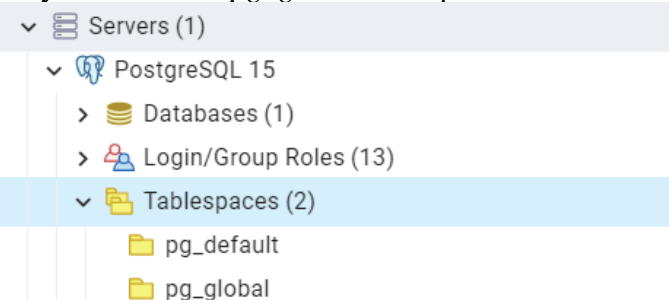
> Login/Group Roles (13)

> Tablespaces (2)

Вторая вкладка – Login/Group Roles. В ней содержатся все созданные роли и группы, в которые данные роли могут входить. Это предназначено для разделения прав пользователей базы данных, например, между администратором и программистом. По умолчанию создана одна роль – *postgres*.



Третья вкладка – Tablespaces. В ней располагаются *табличные пространства*, которые определяют физическое расположение данных. Например, табличные пространства возможно использовать, чтобы расположить архивные данные на медленных носителях, а данные, с которыми идет постоянная работа на быстрых. При инициализации создается два табличных пространства - *pg_default*, для хранения данных по умолчанию и *pg_global* для хранения общих объектов.



1.4. Язык программирования SQL

Работа с базами данных будет осуществляться с помощью языка программирования SQL. В отличие от знакомых вам императивных языков программирования C, C++, Python, Pascal и т.п. SQL является декларативным языком. [4]

Декларативное программирование (от *declare* - описание) — парадигма программирования, в которой задаётся спецификация решения задачи, то есть описывается ожидаемый результат, а не способ его получения.

Сравним между собой два подхода. Предположим, что нам необходимо найти в некоторой базе данных, содержащей информацию о студентах вуза, всех молодых людей по имени Александр. Напишем на псевдо-языке программирования решение данной задачи.

Декларативный подход	Императивный подход
Для всех строчек таблицы Студент Если (имя студента = Александр) То выведи информацию о нем на экран	Выбери всю информацию Из таблицы Студент Где имя студента = Александр


Как можно увидеть из таблицы, в первом случае мы задаем последовательность действий, которые приведут к желаемому результату. Во втором случае – описываем результат того, что хотим получить.

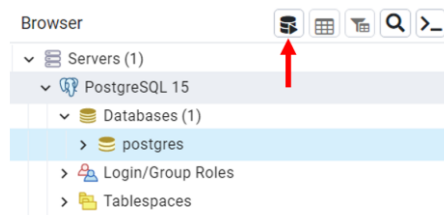
Язык SQL включает в себя операторы, инструкции, вычисляемые функции.

Операторы SQL делятся на:

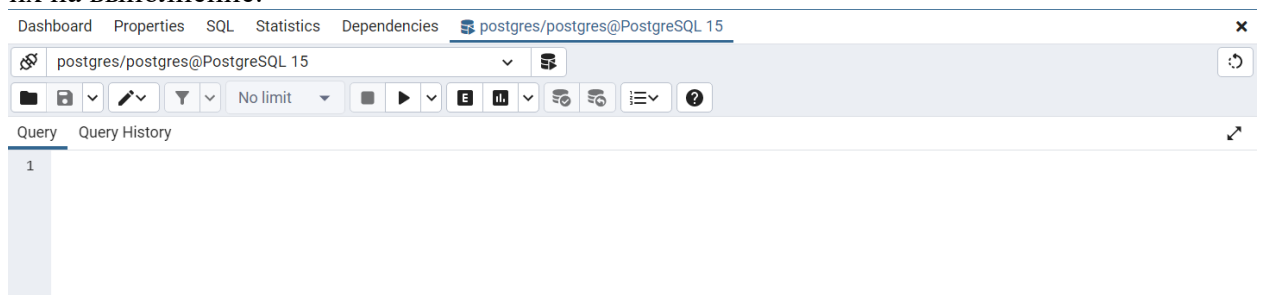
- операторы определения данных (Data Definition Language, DDL)
- операторы манипуляции данными (Data Manipulation Language, DML)
- операторы определения доступа к данным (Data Control Language, DCL)
- операторы управления транзакциями (Transaction Control Language, TCL)

Более подробно данные операторы будут рассмотрены в дальнейшем.

Для того, чтобы создать запрос на языке SQL в программе *pgAdmin* необходимо воспользоваться утилитой Query tool. Для этого перейдите во вкладку Databases – postgres и нажмите на символ .



Перед вами откроется командное окно, в которое возможно вводить запросы и запускать их на выполнение.



Обратим внимание на строку с подключением. Она записана в формате «база данных/роль@сервер». Для данного примера база данных называется *postgres*, пользователь – *postgres*, сервер – PostgreSQL 15

1.5. Создание новой роли.

При работе с базами данных важно разделять права доступа между различными пользователями. В первую очередь это необходимо в целях безопасности. Например, рядовой сотрудник не должен иметь возможности удалить или испортить базу данных.

В СУБД PostgreSQL разграничение доступа реализуется с помощью понятий роли и привилегий.

Каждому пользователю в СУБД назначается роль, обладающая определенными привилегиями. Например, определенной роли возможно выделить привилегию только на чтение данных из таблиц.

Для создания роли используется оператор **CREATE ROLE**

```
CREATE ROLE имя [ [ WITH ] параметр [ ... ] ]
```

Здесь *параметр*:

```
SUPERUSER | NOSUPERUSER
| CREATEDB | NOCREATEDB
| CREATEROLE | NOCREATEROLE
| INHERIT | NOINHERIT
| LOGIN | NOLOGIN
| REPLICATION | NOREPLICATION
| BYPASSRLS | NOBYPASSRLS
```

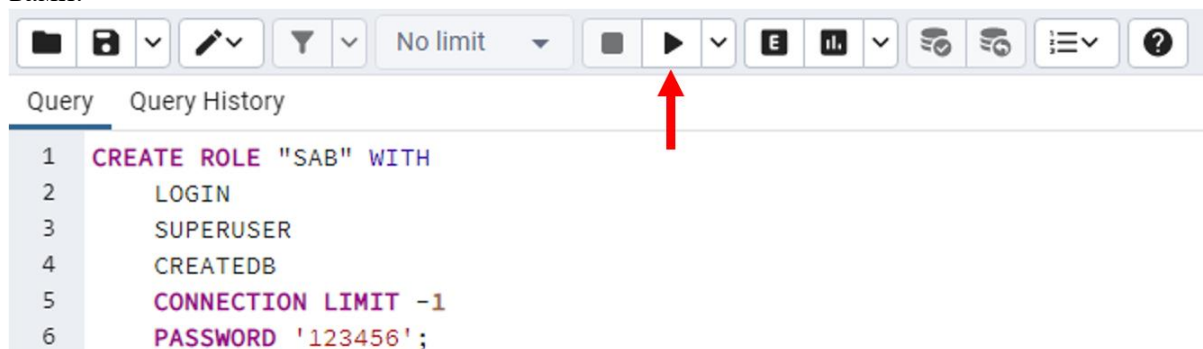
```

| CONNECTION LIMIT предел_подключений
| [ ENCRYPTED | UNENCRYPTED ] PASSWORD 'пароль'
| VALID UNTIL 'дата_время'
| IN ROLE имя_роли [, ...]
| IN GROUP имя_роли [, ...]
| ROLE имя_роли [, ...]
| ADMIN имя_роли [, ...]
| USER имя_роли [, ...]
| SYSID uid

```

Подробно о каждом из параметров возможно прочитать в приложении к документации PostgreSQL [2]

Создадим роль, название которой будет содержать ваши инициалы и наделим её правами администратора. Для этого скопируем следующий скрипт в рабочую область и запустим его с помощью символа ▶. Имя пользователя и пароль должны быть выбраны вами.



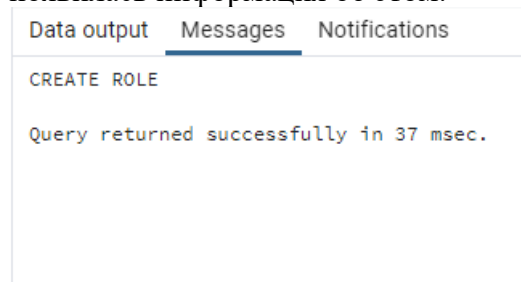
```

CREATE ROLE "SAB" WITH
    LOGIN
    SUPERUSER
    CREATEDB
    CONNECTION LIMIT -1
    PASSWORD '123456';

```

Данный скрипт создает роль SAB, наделяет её привилегиями на вход, создание базы данных и делает её суперпользователем. Созданный пользователь может подключаться неограниченное число раз и имеет пароль для входа 123456.

Обратите внимание, что после успешного выполнения запроса в поле Messages появилась информация об этом.



1.6. Создание базы данных

Перейдем непосредственно к работе с базой данных. Описание предметной области учебной базы данных расположено в приложении.

Первым делом необходимо создать базу данных. Для этого существует команда CREATE DATABASE. Её синтаксис представлен ниже.

```

CREATE DATABASE имя
[ [ WITH ] [ OWNER [=] имя_пользователя ]
  [ TEMPLATE [=] шаблон ]
  [ ENCODING [=] кодировка ]
  [ LC_COLLATE [=] категория_сортировки ]

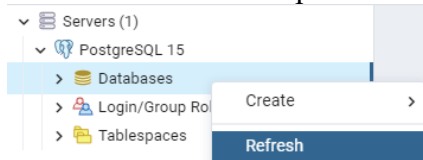
```

```
[ LC_STYPE [=] категория_типов_символов ]  
[ TABLESPACE [=] табл пространство ]  
[ ALLOW_CONNECTIONS [=] разр_подключения ]  
[ CONNECTION LIMIT [=] предел_подключений ]  
[ IS_TEMPLATE [=] это_шаблон ] ]
```

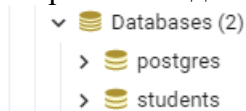
Создадим учебную базу данных с информацией о студентах вуза. Для этого выполним запрос:

```
CREATE DATABASE students;
```

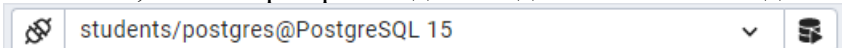
Обновим информацию о базах данных на сервере. Для этого щелкнем по строке Databases в левой колонке правой кнопкой мыши и выберем пункт *Refresh*.



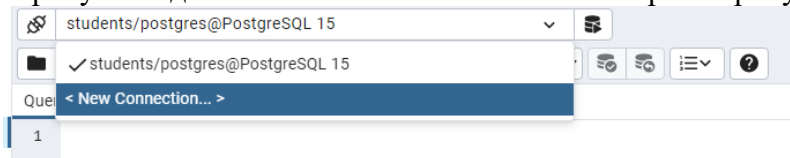
Раскроем выпадающий список и убедимся, что появилась новая база данных.



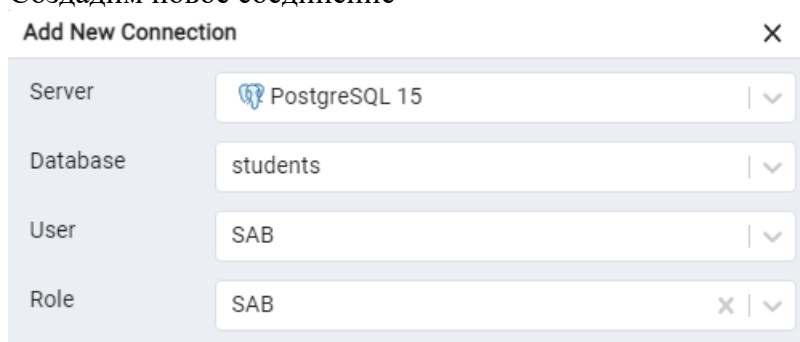
Нажмем на созданную базу данных и создадим новый экземпляр Query tool. Обратите внимание, что теперь произведено подключение к базе данных *students*.



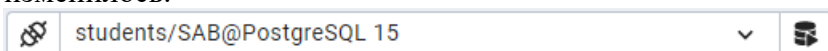
Подключимся к базе данных от имени созданной нами выше роли. Для этого нажмем на строку с соединением и в выпавшем окне выберем строку <New Connection>




Создадим новое соединение



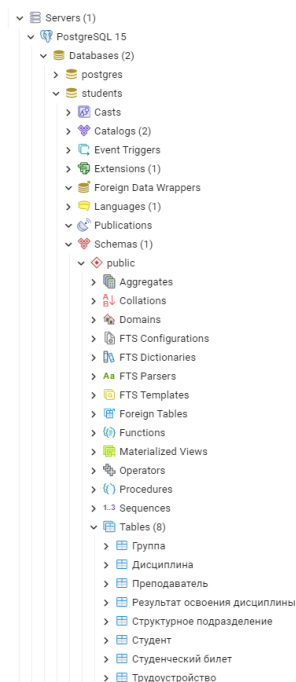
После успешного подключения обратим внимание на то, что имя пользователя изменилось.



Создадим таблицы в базе данных. Для этого откроем скрипт «create database», нажав на

символ  и запустим его. Если все выполнилось верно, то в поле сообщений появится строчка: «CREATE TABLE Query returned successfully»

Аналогично откроем файл «insert students» и заполним базу данными.




Данные скрипты будут более подробно рассмотрено в следующих лабораторных работах. Убедимся в том, что все таблицы были созданы. Для этого в выпадающем списке найдем созданные таблицы по пути: «Servers – PostgreSQL 15 – students – Schemas – public – Tables».

Нажав правой кнопкой мыши на название таблицы, мы можем выбрать некоторые действия. Рассмотрим некоторые из них.

Count Rows возвращает число строк в таблице.

View/Edit Data позволяет вывести на экран содержимое таблицы.

Дважды щелкнув по любой ячейке таблицы возможно изменить её значение. После

внесенных изменений необходимо зафиксировать их, нажав на символ  или клавишу F6.

2. Практическая часть

2.1. Задание 1.

Создайте новую роль «Ваши инициалы» junior». Выделите ей привилегии на просмотр данных. Подключитесь от её имени к базе данных Students и попробуйте удалить её с помощью запроса

```
DROP DATABASE students;
```

2.2. Задание 2.

Изучите учебную базу данных *Students*:

- Ознакомьтесь со схемой данных.
- Ознакомьтесь с содержимым таблиц БД.
- Определите число строк в каждой из таблиц.
- Предположим, что студентка группы ИТД-33 Коровина Мария Георгиевна вышла замуж за студента с номером студенческого билета 820080. Измените её фамилию на фамилию её супруга.

2.3. Задание 3.

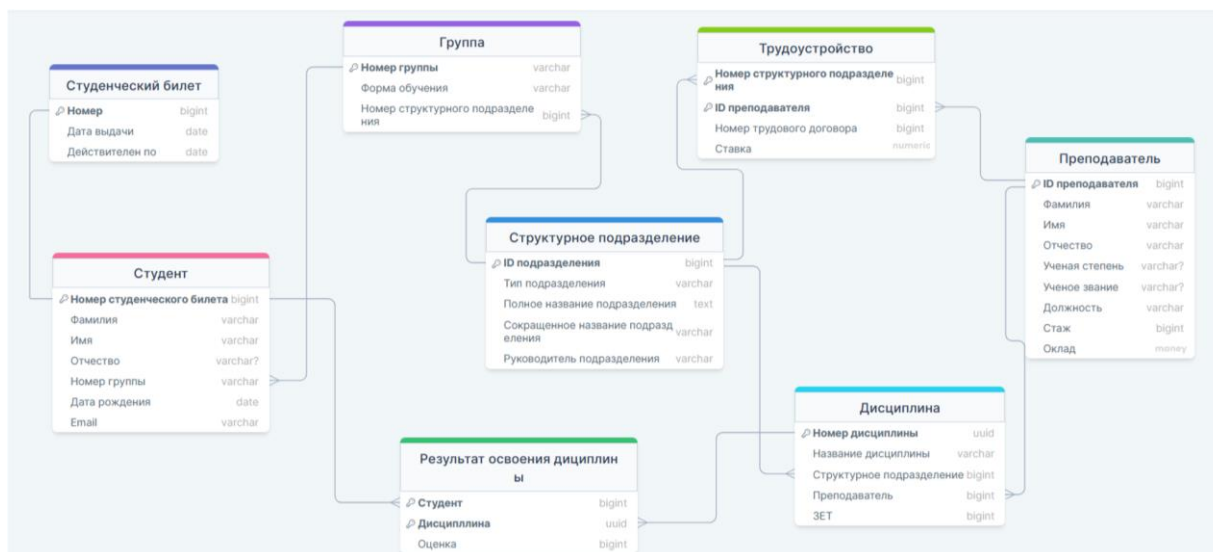
Определите, какие таблицы в базе данных Students являются главными, а какие для них подчиненными.

Список литературы

- [1] «Исходный код СУБД postgres,» [В Интернете]. Available: <https://github.com/postgres/postgres>. [Дата обращения: 30 01 2023].
- [2] Документация к PostgreSQL 15.1, 2022.
- [3] Е. Рогов, PostgreSQL изнутри, 1-е ред., Москва: ДМК Пресс, 2023, р. 662.
- [4] Б. А. Новиков, Е. А. Горшкова и Н. Г. Графеева, Основы технологии баз данных, 2-е ред., Москва: ДМК пресс, 2020, р. 582.
- [5] Е. П. Моргунов, PostgreSQL. Основы языка SQL, 1-е ред., Санкт-Петербург: БХВ-Петербург, 2018, р. 336.

Приложение

База данных университета



В качестве учебной базы данных рассмотрим близкую каждому из нас область – базу данных, хранящую информацию о студентах и преподавателях вуза, а также оценках студентов за дисциплины.

Рассматриваемый вуз состоит из структурных подразделений – институтов и кафедр. Каждое подразделение является выпускающим и к нему могут быть прикреплены группы.

В каждой группе обучаются студенты. Один студент может быть прикреплен только к одной группе. У каждого студента есть свой студенческий билет с уникальным номером.

В структурных подразделениях могут работать преподаватели. Однако возможен случай, когда преподавателя приглашают извне и он не прикреплен ни к одному из подразделений.

Во время обучения студенты изучают дисциплины. За каждую пройденную дисциплину им выставляется оценка от 2 до 5. За каждой дисциплиной закреплен один преподаватель.