

База данных **Академия** (*Academy*) содержит информацию о сотрудниках, внутреннем устройстве академии и читаемых лекциях. Преподаватели, читающие лекции в академии представлены в виде таблицы **Преподаватели** (*Teachers*), в которой собрана основная информация, такая как: имя и фамилия. Кроме этого, есть возможность узнать является ли преподаватель куратором, деканом или заведующим при помощи соответствующих таблиц: **Кураторы** (*Curators*), **Деканы** (*Deans*) и **Заведующие** (*Heads*). Также в базе данных присутствует информация о группах, хранимая в таблице **Группы** (*Groups*). Данные о факультетах и кафедрах содержатся в таблицах **Факультеты** (*Faculties*) и **Кафедры** (*Departments*) соответственно. Помимо этого, база данных хранит информацию, связанную с проводимыми лекциями. Описание дисциплин, по которым читаются лекции, содержится в таблице **Дисциплины** (*Subjects*). Информация о студентах содержится в таблице **Студенты** (*Students*). Ниже представлено детальное описание структуры каждой таблицы.

**1. Преподаватели (Teachers)**

* Идентификатор (Id).
* Имя (Name). Имя преподавателя.
* Фамилия (Surname). Фамилия преподавателя.
* Отчество (MiddleName). Отчество преподавателя.

**2. Студенты (Students)**

* Идентификатор (Id).
* Имя (Name). Имя студента.
* Фамилия (Surname). Фамилия студента.
* Отчество (MiddleName). Отчество преподавателя.

**3. Кураторы (Curators)**

* Идентификатор (Id).
* Идентификатор преподавателя (TeacherId). Преподаватель, который является куратором.

**4. Деканы (Deans)**

* Идентификатор (Id).
* Идентификатор преподавателя (TeacherId). Преподаватель, который является деканом.

**5. Заведующие (Heads)**

* Идентификатор (Id).
* Идентификатор преподавателя (TeacherId). Преподаватель, который является заведующим.

**6. Кафедры (Departments)**

* Идентификатор (Id).
* Название (Name). Название кафедры.
* Корпус (Building). Номер корпуса, в котором располагается кафедра.
* Идентификатор факультета (FacultyId). Факультет, в состав которого входит кафедра.
* Идентификатор заведующего (HeadId). Заведующий кафедрой.

**7. Факультеты (Faculties)**

* Идентификатор (Id).
* Название (Name). Название факультета.
* Корпус (Building). Номер корпуса, в котором располагается факультет.
* Идентификатор декана (DeanId). Декан факультета

**8. Группы (Groups)**

* Идентификатор (Id).
* Название (Name). Название группы.
* Курс (Year). Курс (год) на котором обучается группа.
* Идентификатор кафедры (DepartmentId).

**9. Группы и кураторы (GroupsCurators)**

* Идентификатор (Id).
* Идентификатор куратора (CuratorId). Куратор.
* Идентификатор группы (GroupId). Группа.

**10. Группы и лекции (GroupsLectures)**

* Идентификатор (Id).
* Идентификатор группы (GroupId). Группа.
* Идентификатор лекции (LectureId). Лекция.

**11. Группы и студенты (GroupsStudents)**

* Идентификатор (Id).
* Идентификатор группы (GroupId). Группа.
* Идентификатор студента (StudentId). Студент.

**12. Дисциплины (Subjects)**

* Идентификатор (Id).
* Название (Name). Название дисциплины.

create table [Curators](

[Id] int not null identity(1, 1) primary key,

[TeacherId] int not null

);

create table [Deans](

[Id] int not null identity(1, 1) primary key,

[TeacherId] int not null

);

create table [Departments](

[Id] int not null identity(1, 1) primary key,

[Name] nvarchar(100) not null unique check ([Name] <> N''),

[Building] int not null check ([Building] between 1 and 5),

[FacultyId] int not null,

[HeadId] int not null

);

create table [Faculties](

[Id] int not null identity(1, 1) primary key,

[Name] nvarchar(100) not null unique check ([Name] <> N''),

[Building] int not null check ([Building] between 1 and 5),

[DeanId] int not null

);

create table [Groups](

[Id] int not null identity(1, 1) primary key,

[Name] nvarchar(10) not null unique check ([Name] <> N''),

[Year] int not null check ([Year] between 1 and 5),

[DepartmentId] int not null

);

create table [GroupsCurators](

[Id] int not null identity(1, 1) primary key,

[CuratorId] int not null,

[GroupId] int not null

);

create table [Heads](

[Id] int not null identity(1, 1) primary key,

[TeacherId] int not null

);

create table [Subjects](

[Id] int not null identity(1, 1) primary key,

[Name] nvarchar(100) not null unique check ([Name] <> N''),

[TeacherId] int not null

);

create table [Teachers](

[Id] int not null identity(1, 1) primary key,

[Name] nvarchar(max) not null check ([Name] <> N''),

[Surname] nvarchar(max) not null check ([Surname] <> N''),

[Patronymic] nvarchar(max) not null check ([Patronymic] <> N'')

);

create table [Students](

[Id] int not null identity(1, 1) primary key,

[Name] nvarchar(max) not null check ([Name] <> N''),

[Surname] nvarchar(max) not null check ([Surname] <> N''),

[Patronymic] nvarchar(max) not null check ([Patronymic] <> N'')

);

create table [GroupsStudents](

[Id] int not null identity(1, 1) primary key,

[StudentId] int not null,

[GroupId] int not null

);

alter table [Curators] add foreign key ([TeacherId]) references [Teachers]([Id]);

alter table [Deans] add foreign key ([TeacherId]) references [Teachers]([Id]);

alter table [Departments] add foreign key ([FacultyId]) references [Faculties]([Id]);

alter table [Departments] add foreign key ([HeadId]) references [Heads]([Id]);

alter table [Faculties] add foreign key ([DeanId]) references [Deans]([Id]);

alter table [Groups] add foreign key ([DepartmentId]) references [Departments]([Id]);

alter table [GroupsCurators] add foreign key ([CuratorId]) references [Curators]([Id]);

alter table [GroupsCurators] add foreign key ([GroupId]) references [Groups]([Id]);

alter table [Heads] add foreign key ([TeacherId]) references [Teachers]([Id]);

alter table [GroupsStudents] add foreign key ([GroupId]) references [Groups]([Id]);

alter table [GroupsStudents] add foreign key ([StudentId]) references [Students]([Id]);