нетология

# ПРОЕКТИРОВАНИЕ БД, СВЯЗИ, 3НФ

АЛЕКСАНДР ИВАНОВ

# АЛЕКСАНДР ИВАНОВ

Ведущий инженер-программист в
«Лаборатория компьютерного моделирования»

oz.sasha.ivanov@gmail.com

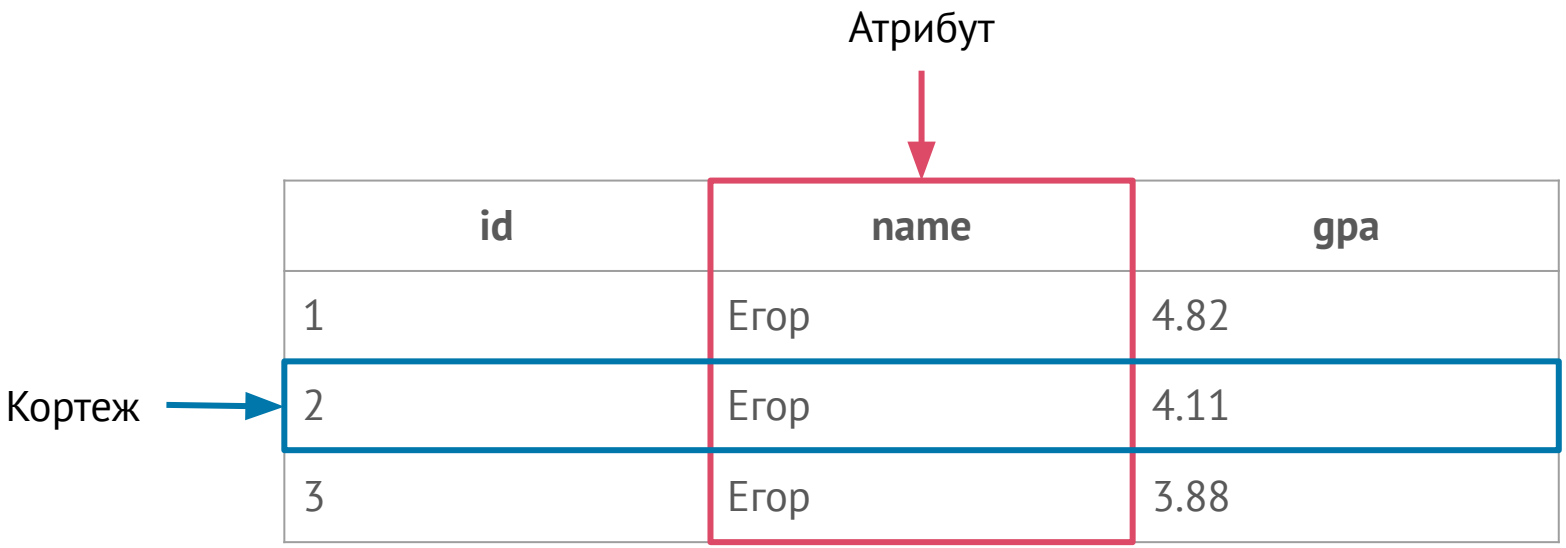
@artreys

# План занятия

1. Проектирование схем БД.
2. Связи между отношениями:
    a. Один к одному.
    b. Один ко многим.
    c. Многие ко многим.
3. Нормальные формы:
    a. Первая нормальная форма.
    b. Вторая нормальная форма.
    c. Третья нормальная форма.

# Вспомним прошлые занятия

```
create table if not exists Student (
     id serial primary key,
     name varchar(40) not null,
     gpa numeric(3, 2) check (gpa >= 0 and gpa <= 5)
);
```




Пример отношения «Успеваемость студентов»

# Проектирование схем БД

# Проектирование схем БД

Зачем лишний атрибут?

| name | gpa |
|---|---|
| Егор | 4.82 |
| Егор | 4.11 |
| Егор | 3.88 |

Как отличить одного Егора от другого?

# Primary key

Первичный ключ — это отдельное поле или комбинация полей, которые однозначно определяют запись — кортеж.

```
create table if not exists Student (
     id serial primary key,
     name varchar(40) not null,
     gpa numeric(3, 2) check (gpa >= 0 and gpa <= 5)
);

create table if not exists Student (
     name varchar(40) primary key,
     gpa numeric(3, 2) check (gpa >= 0 and gpa <= 5)
);  # какой недостаток?

create table if not exists Student (
     name varchar(40),
     gpa numeric(3, 2) check (gpa >= 0 and gpa <= 5),
     constraint student_pk primary key (name, gpa)
);  # здесь все ок?
```

# Primary key

Primary key (первичный ключ) на схемах-таблицах обозначается с помощью подчеркивания. На схеме ниже атрибут **id** — это первичный ключ.

| **id** | **name** | **gpa** |
|---|---|---|
| 1 | Егор | 4.25 |
| 2 | Егор | 3.82 |
| 3 | Егор | 4.25 |

# Связи между отношениями

# Смоделируем ситуацию

Есть система онлайн обучения.

В ней есть пользователи — студенты. У каждого пользователя есть почта (она же является логином), пароль и имя.

Также у пользователя есть возможность указать дополнительную информацию: дату рождения, город проживания, свои интересы.

Пользователи могут записываться на курсы.

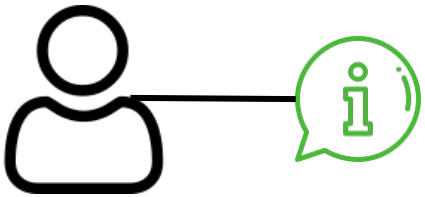
В рамках курса пользователи должны выполнять домашние задания и загружать их в систему.

# Связи между отношениями

**Типы связей**

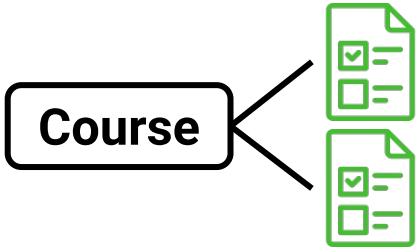
**один к одному**

пользователь и дополнительная информация о нём



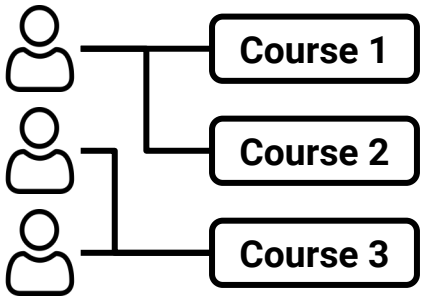
**один ко многим**

домашние задания на курсе




**многие ко многим**

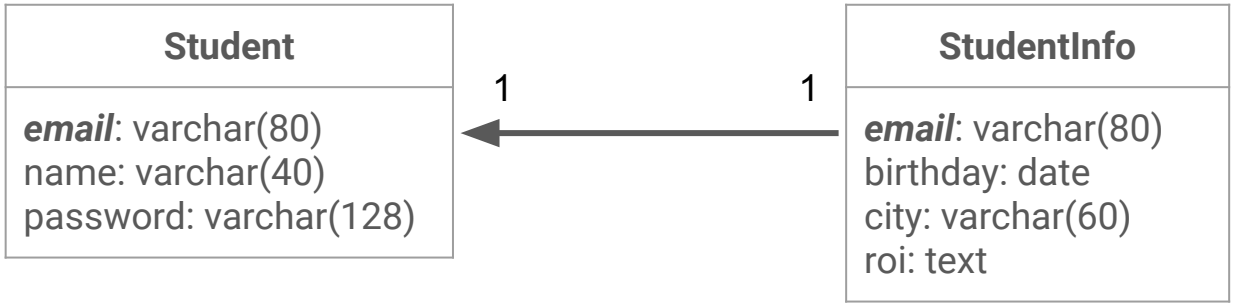
пользователи и курсы




Связи, как и первичный ключ, можно попросить контролировать СУБД. Для этого используется ограничение **foreign key**.

# Один к одному. Вариант 1

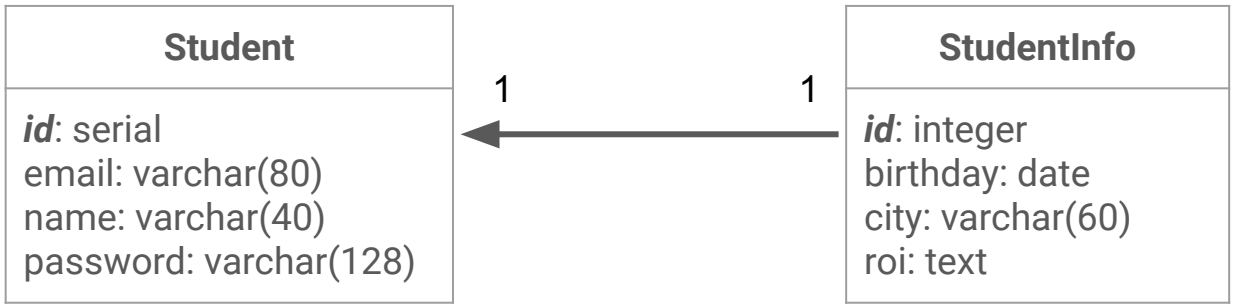
Связываем студента и дополнительную информацию о нём.




```
create table if not exists Student (
     email varchar(80) primary key,
     name varchar(40) not null,
     password varchar(128) not null
);

create table if not exists StudentInfo (
     email varchar(80) primary key references Student(email),
     birthday date,
     city varchar(60),
     roi text
);
```

# Один к одному. Вариант 2

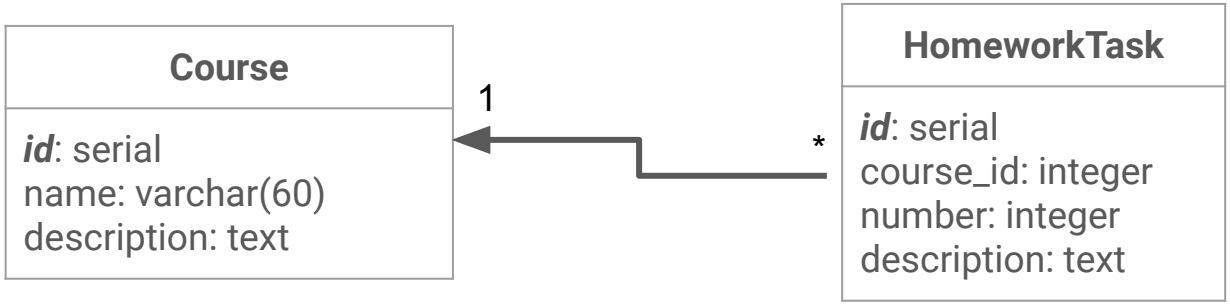



```
create table if not exists Student (
     id serial primary key,
     email varchar(80) unique not null,
     name varchar(40) not null,
     password varchar(128) not null
);

create table if not exists StudentInfo (
     id integer primary key references Student(id),
     birthday date,
     city varchar(60),
     roi text
);
```

# Один ко многим

Связываем описание домашних заданий с курсами, к которым они относятся.

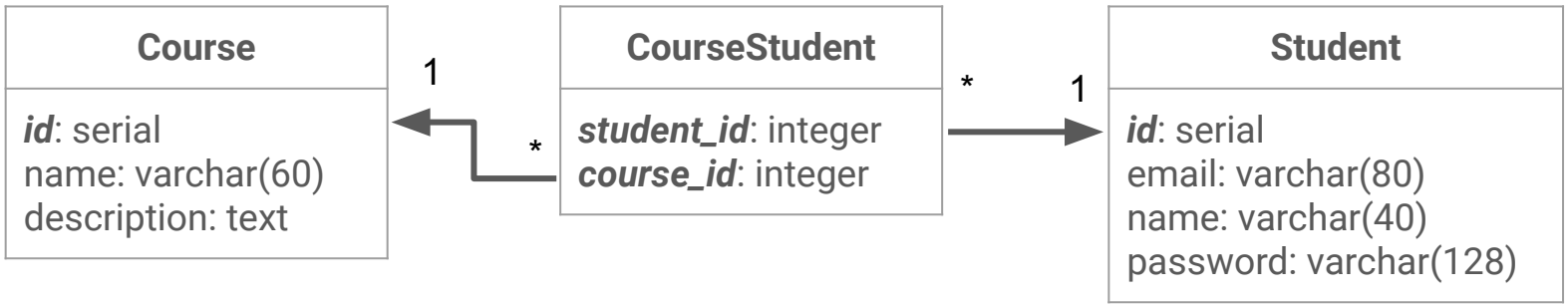



```
create table if not exists Course (
     id serial primary key,
     name varchar(60) not null,
     description text
);

create table if not exists HomeworkTask (
     id serial primary key,
     course_id integer not null references Course(id),
     number integer not null,
     description text not null
);
```

# Многие ко многим. Вариант 1

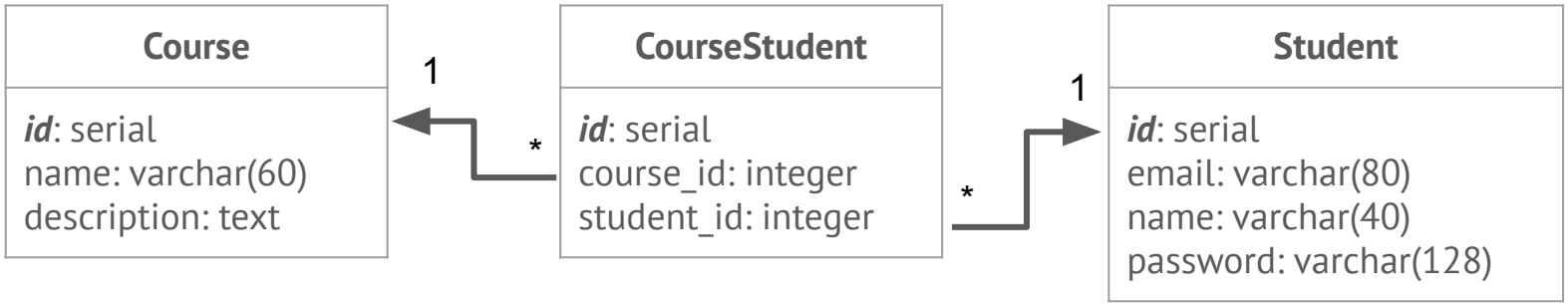
Связываем студентов и курсы, на которые они записаны.




```
create table if not exists CourseStudent (
     course_id integer references Course(id),
     student_id integer references Student(id),
     constraint pk primary key (course_id, student_id)
);
```

| student_id | course_id |
|---|---|
| 1 (Вова) | 1 (Python) |
| 1 (Вова) | 2 (Java) |
| 2 (Дима) | 1 (Python) |

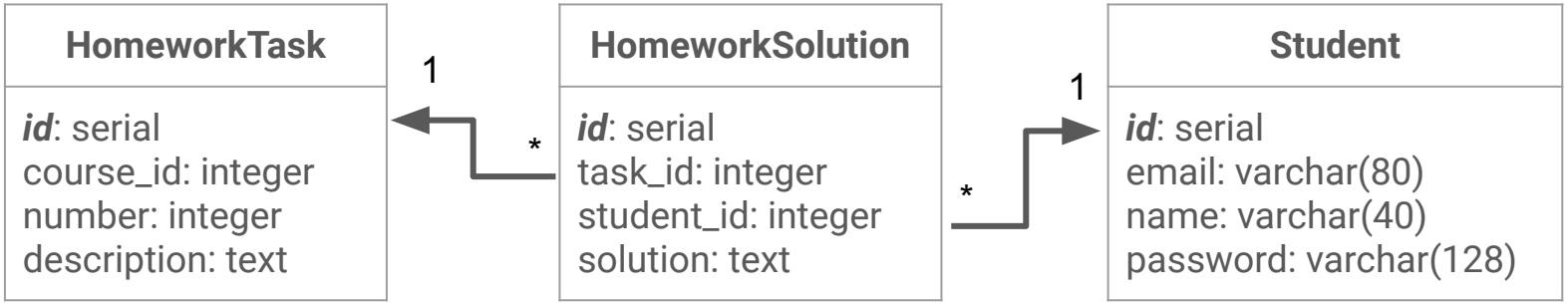
# Многие ко многим. Вариант 2




```
create table if not exists CourseStudent (
     id serial primary key,
     course_id integer not null references Course(id),
     student_id integer not null references Student(id)
);
```

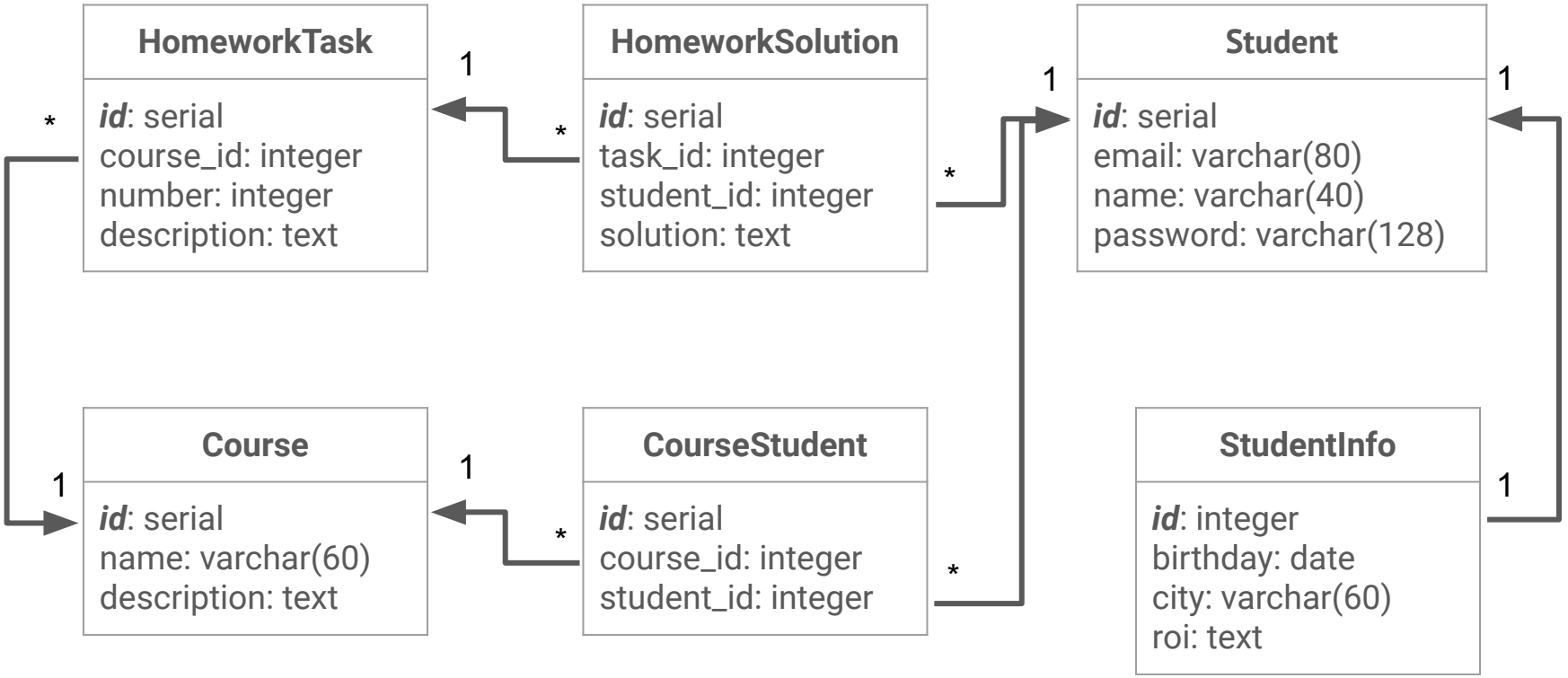
# Многие ко многим

Связываем студента и домашние работы, которые он отправил в систему.




```
create table if not exists HomeworkSolution (
     id serial primary key,
     task_id integer not null references HomeworkTask(id),
     student_id integer not null references Student(id),
     solution text not null
);
```

# Итоговая схема



# Нормальные формы

# Нормальные формы (НФ)

Нормализация — процесс постепенного преобразования отношений (таблиц) для того, чтобы убрать дублирование данных. Это помогает уменьшить потенциальную противоречивость в БД.

**HomeworkTask**

| id | course | number | description |
|---|---|---|---|
| 1 | Python | 1 | ... |
| 2 | Python | 2 | ... |
| 3 | JavaScript | 1 | ... |
| 4 | Python | 3 | ... |

# Первая нормальная форма (1НФ)

Сохраняемые данные на пересечении строк и столбцов должны представлять скалярное значение, а таблицы не должны содержать повторяющихся строк.

| **Сотрудник** | **Номер телефона** |
|---|---|
| Иванов И. И. | 283-56-82, 390-57-34 |
| Петров П. П. | 708-62-34 |

| **Сотрудник** | **Номер телефона** |
|---|---|
| Иванов И. И. | 283-56-82 |
| Иванов И. И. | 390-57-34 |
| Петров П. П. | 708-62-34 |

# Вторая нормальная форма (2НФ)

1НФ + каждый столбец, который не является ключом, должен зависеть от первичного ключа.

| **Филиал компании** | **Должность** | **Зарплата** | **Наличие графического планшета** |
|---|---|---|---|
| Филиал в Томске | Дизайнер | 40000 | Есть |
| Филиал в Москве | Программист | 60000 | Нет |
| Филиал в Томске | Программист | 40000 | Нет |

| **Филиал компании** | **Должность** | **Зарплата** |
|---|---|---|
| Филиал в Томске | Дизайнер | 40000 |
| Филиал в Томске | Программист | 60000 |
| Филиал в Москве | Программист | 40000 |

| **Должность** | **Наличие графического планшета** |
|---|---|
| Дизайнер | Есть |
| Программист | Нет |

# Третья нормальная форма (3НФ)

2НФ + каждый столбец, который не является ключом, должен зависеть от первичного ключа **напрямую**.

| **Сотрудник** | **Отдел** | **Телефон** |
|---|---|---|
| Гришин | Бухгалтерия | 11-22-33 |
| Васильев | Бухгалтерия | 11-22-33 |
| Петров | Снабжение | 44-55-66 |

| **Сотрудник** | **Отдел** |
|---|---|
| Гришин | Бухгалтерия |
| Васильев | Бухгалтерия |
| Петров | Снабжение |

| **Отдел** | **Телефон** |
|---|---|
| Бухгалтерия | 11-22-33 |
| Снабжение | 44-55-66 |

# Остальные нормальные формы

- Нормальная форма Бойса-Кодда (НФБК)
- Четвертая нормальная форма (4НФ)
- Пятая нормальная форма (5НФ)
- Шестая нормальная форма (6НФ)

Хорошая статья с примерами на Хабре.

# Плюсы и минусы нормализации

| Плюсы | Минусы |
| --- | --- |
| Декомпозиция информации | Время на приведение к нормальным формам |
| Строгое хранение данных без возможности хранить дублированную и противоречивую информацию | Накладные расходы при извлечении информации на объединение таблиц |

# Выводы

Нормализация позволяет избегать дублирования данных и упрощать редактирование сущностей.

Есть несколько видов нормальных форм. Все их знать и помнить не обязательно, но важно всегда задавать себе вопрос:

*«Должны ли эти данные быть в отдельной таблице или достаточно дополнительной колонки?».*

# Итоги

Сегодня на занятии мы выяснили, как проектировать БД, а именно:

1. рассмотрели типы связей между отношениями (таблицами),
2. научились приводить таблицы к нормальным формам.

# Домашнее задание

Давайте посмотрим ваше домашнее задание.

- Вопросы по домашней работе задаём в чате Slack!
- Задачи можно сдавать по частям.
- Зачёт по домашней работе проставляется после того, как приняты **все задачи**.

нетология

**Задавайте вопросы и напишите отзыв о лекции!**


# АЛЕКСАНДР ИВАНОВ

oz.sasha.ivanov@gmail.com

@artreys