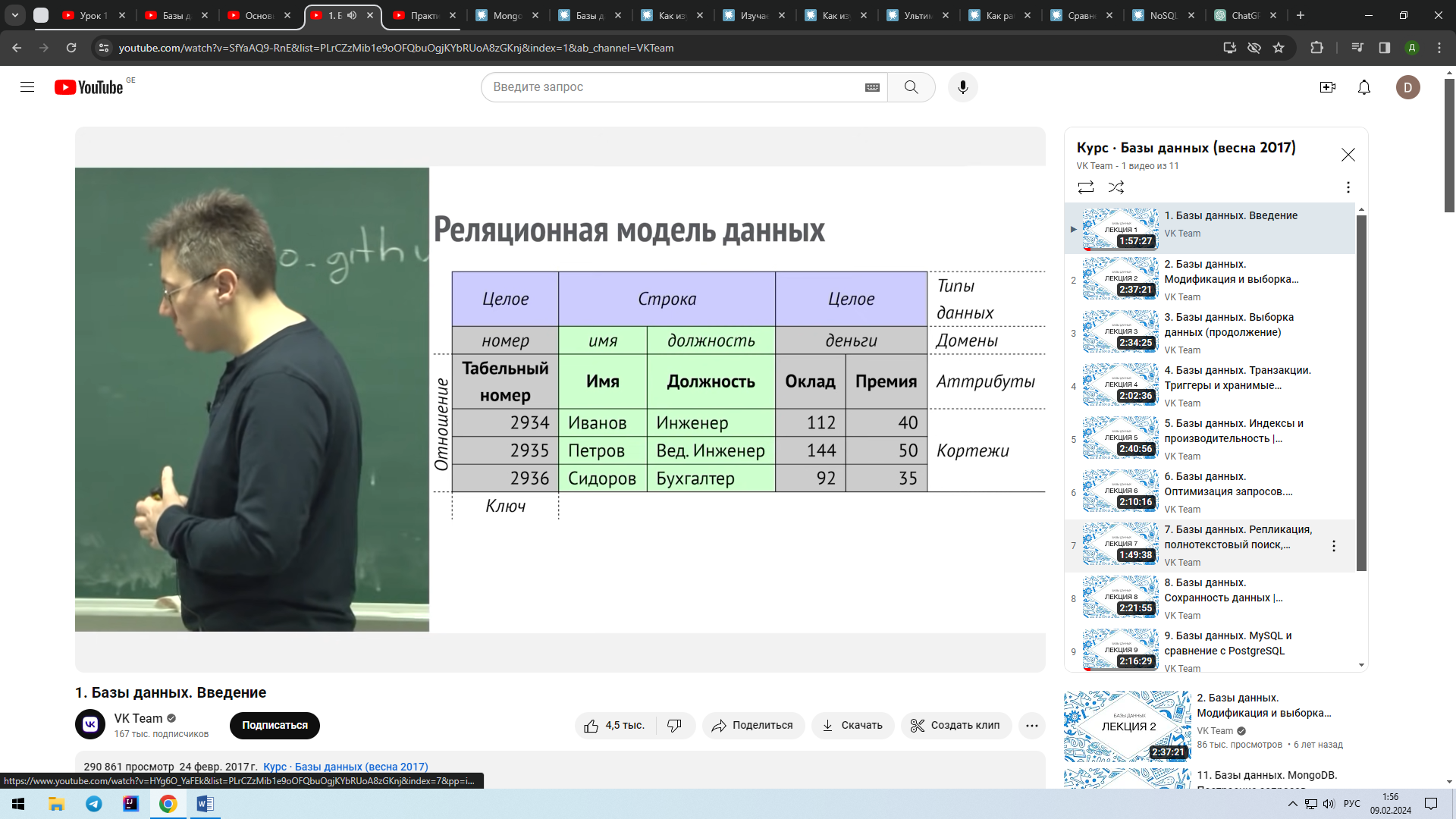
**Терминология:**



*Атрибут (столбец) –* характеристика сущности в БД, состоит из имени и домена.

*Домен –* допустимый набор значений поля, формирующийся из типа данных и ограничений.

*Строка/кортеж* – одна запись в таблице, т.е. набор значений каждого атрибута для определенного ключа.

*Отношение –* неупорядоченное множество кортежей.

*Сущность* – Объект предметной области, который может быть описан

*Результирующий набор* – результат запроса на языке SQL

**Типы отношений**

Один к одному (One-to-One): Каждая запись в одной таблице соответствует одной и только одной записи в другой таблице, и наоборот.

Один ко многим (One-to-Many): Каждая запись в одной таблице может иметь несколько соответствующих записей в другой таблице, но каждая запись во второй таблице соответствует только одной записи в первой таблице.

Многие к одному (Many-to-One): Обратное отношение к “Один ко многим”. Каждая запись в одной таблице имеет только одну соответствующую запись в другой таблице, но каждая запись во второй таблице может иметь несколько связанных записей в первой таблице.

Многие ко многим (Many-to-Many): Каждая запись в одной таблице может соответствовать нескольким записям в другой таблице, и наоборот. Для реализации таких отношений требуется использование дополнительной таблицы. Для такой таблицы primary key будет выражен двумя этими атрибутами, которые и выражают это отношение, так как их комбинации всегда будут уникальны.

Ссылка на саму себя (Self-Referencing): Таблица может ссылаться на другую запись в собственной таблице.

Создание промежуточной таблицы подразумевает отношение one-to-one, т.к. каждая строка такой таблицы будет иметь только одно соответствие для каждой своей связи (таким образом можно разбить любое отношение до one-to-one).

**Constraints** – правило, установленное для данных в таблице, с целью обеспечения их корректности и соответствия с определенными условиями.

*Not null, Unique, Check* (Задает диапазон значений), *Default* (Добавляет значение по умолчанию, в случае отсутствия данных), *Primary Key, Foreign Key (ограничение ссылочной целостности)*.

**Primary key –** ограничениена аргумент или набор аргументов, по которому (-ым) можно гарантированно идентифицировать строку. Это достигается за счет свойства уникальности и наличия значения отличного от null.

В качестве первичного ключа может служить либо естественный ключ (т.е. уже существующий атрибут). Минусы – проблемы, в случае его изменения, так как нужно будет поменять его во всех таблицах, может занимать больше памяти. Либо суррогатный – к отношению добавляется новый атрибут никак его не характеризующий, например, порядковый номер или id, в качестве значения используется число с автоинкрементом.

**Foreign key –** ограничение, заданное аргументом или набором аргументов, обеспечивающее связь между таблицами посредством ссылок. Атрибут, на который ссылается внешний ключ должен иметь свойство уникальности – UNIQUE или PRIMARY KEY.

Предок – таблица, содержащая уникальный ключ, потомок – таблица, содержащая внешний ключ.

Позволяют обеспечивать ссылочную целостность и устанавливать правила отношений между таблицами:

Нарушение целостности (без ключа):

1. Удаление строки предка при имеющихся записях в потомке (потомок будет ссылаться в никуда)
2. Добавление/обновление строки потомка со ссылкой на несуществующую строку предка
3. Обновление первичного ключа в строке предка (потомок будет ссылаться на старые значения предка)

[Видос с примерами.](https://www.youtube.com/watch?v=199vDwIIt5Y&list=PLf30vI0hEi1v435cBmZSHkr1QAJdOk9mb&index=6&ab_channel=RclassTech)

Foreign key позволяет задать поведение при таких нарушениях запрещая нарушения типа 2 и задание поведения для 1 (**ON** **DELETE**), 3 (**ON** **UPDATE**) с помощью **ON** **DELETE** и **ON** **UPDATE** и их опций**:**

**CASCADE**: Удалить/обновить связанные записи в дочерней таблице.  
**SET** **NULL**: Установить значение внешнего ключа в дочерней таблице в NULL.  
**SET** **DEFAULT**: Установить значение внешнего ключа в дочерней таблице в значение по умолчанию.  
**NO** **ACTION** или **RESTRICT**: Запретить удаление/обновление записи, если на неё ссылаются другие таблицы (разница во времени проверки: no action в конце транзакции, restrict сразу)

**Синтаксис constraint:**

Краткая запись без возможности задать имя ограничению (в таком варианте ограничения получат автоматически сгенерированные имена).

**CREATE** **TABLE** employees (  
employee\_id SERIAL **PRIMARY** **KEY**,   
employee\_name **VARCHAR**(**100**) **NOT** **NULL**,   
department\_id INTEGER **REFERENCES** departments (department\_id) );

Добавление CONSTRAINT позволяет задать имя для ограничения.

**CREATE** **TABLE** employees (  
employee\_id SERIAL **PRIMARY** **KEY** CONSTRAINT pk\_employees\_id,  
employee\_name **VARCHAR**(**100**) **NOT** **NULL** CONSTRAINT pk\_employee\_name,   
department\_id INTEGER **REFERENCES** departments (department\_id) );

Именовать внешние ключи и задавать задавать ограничения для нескольких столбцов можно только на уровне таблицы.

**CREATE** **TABLE** employees (  
employee\_id SERIAL,   
employee\_name **VARCHAR**(**100**) **NOT** **NULL**,   
department\_id INTEGER,  
**CONSTRAINT** fk\_department\_id **FOREIGN** **KEY** (department\_id)   
**REFERENCES** departments (department\_id),  
**CONSTRAINT** pk\_employee PRIMARY KEY (employee\_id, employee\_name));

Задает имена для первичного и внешнего ключа и устанавливает композитный первичный ключ (просто для примера).