**[Нормализация данных](https://www.youtube.com/watch?v=zqQxWdTpSIA&ab_channel=ListenIT) –** процесс удаления избыточных (дублирующийся) данных. Избыточность, как правило, устраняется за счет декомпозиции отношений (разбиение одной таблицы на несколько). Избыточность создает предпосылки для появления аномалий, снижает производительность и гибкость в управлении данными.

Избыточность данных создает предпосылки для возникновения аномалий (некоторые примеры):

\*1) Дублирующая информация – ненормализованная таблица может содержать множество одинаковых значений одного типа, что увеличивает объем данных для хранения

2) Изменение значения типа - Если значение типа изменится, то нужно изменить все строки, содержащие это значение

3) Потеря данных - При заполнении таблицы данными типа он может быть записан неверно, что приведет к невозможности включения этих данных в нужный тип.

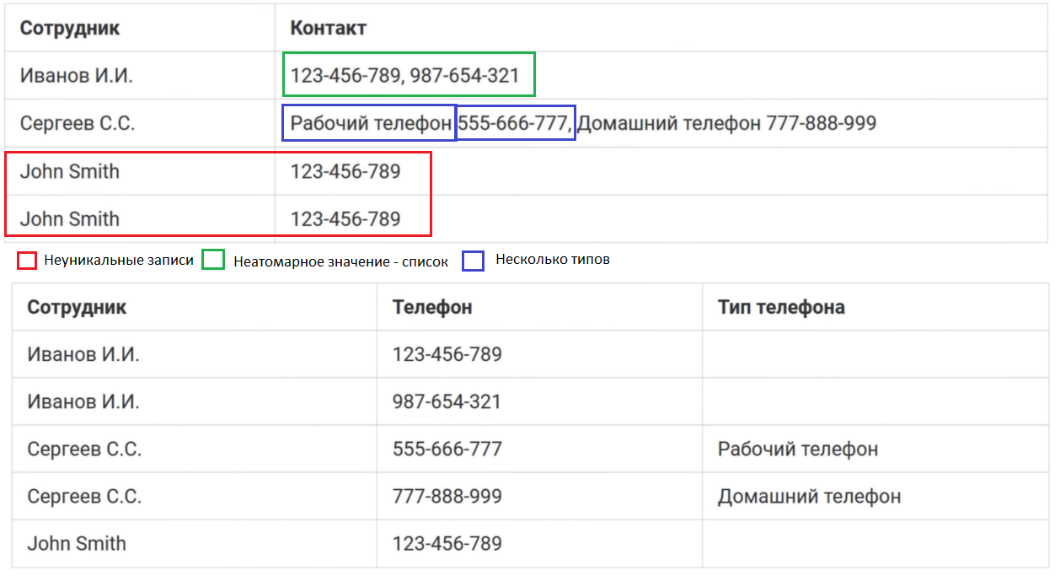
4) Потеря типа - Информация о типах хранится в таблице представляющей сущности. Если тип \*представлен одной сущностью в таблице, то удаление этой сущности приведет к потере типа.

***Нормальные формы*** *–* Набор правил и критериев, которым должна отвечать БД для соответствия той или иной форме БД. Каждая последующая форма подразумевает, что БД должна соответствовать всем предыдущим нормальным формам.

**0НФ** – Приведение БД к виду, соответствующему базовым принципам реляционной теории. Строки в таблицах не должны быть пронумерованы (порядок строк и столбцов не имеет значения). Т.е. невозможно обратиться к строке по ее номеру.

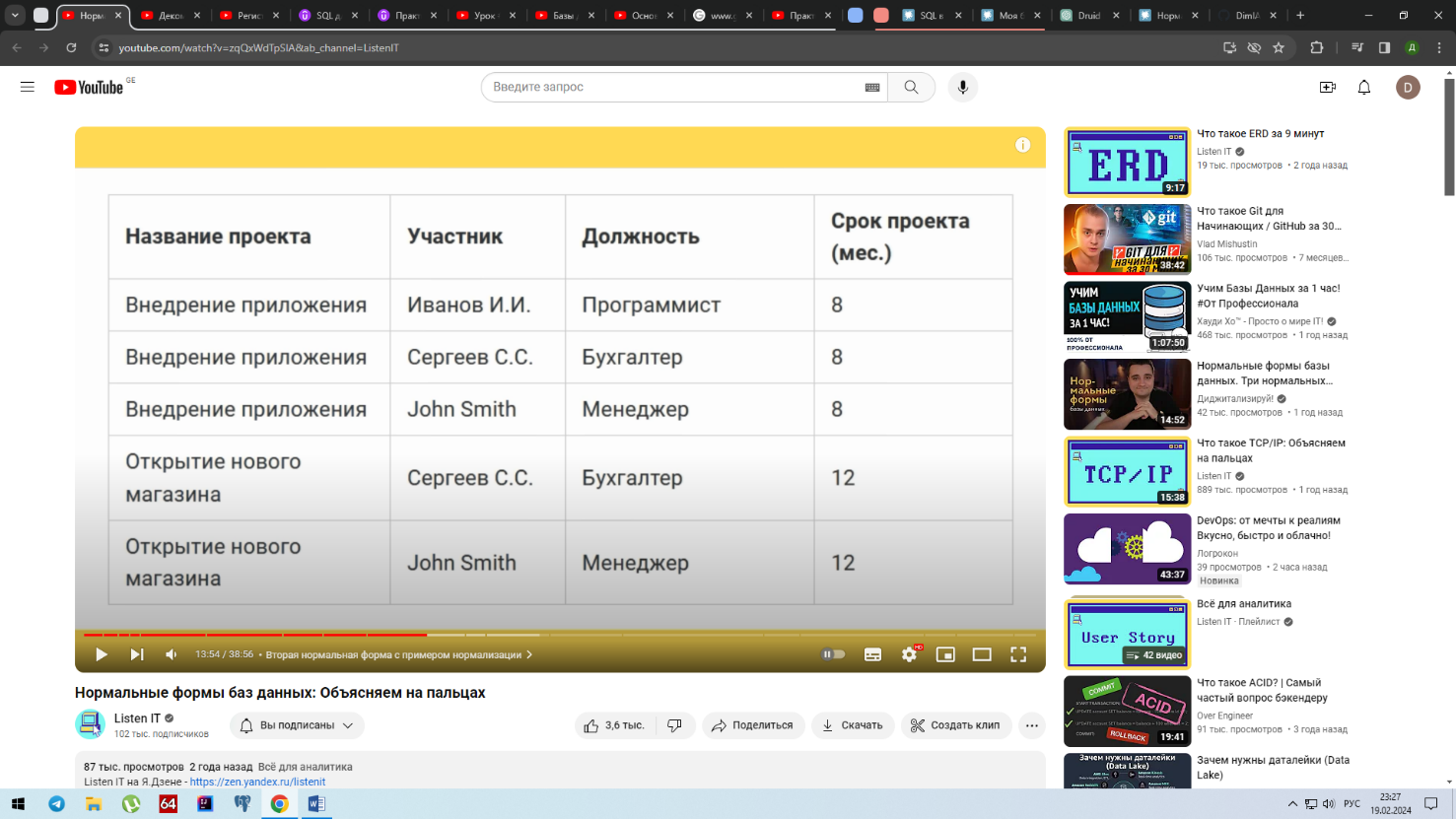
**1НФ** *–*

Строка – уникальна (нет полностью идентичных строк).  
Ячейка - атомарна (хранит одно неделимое по смыслу значение - не должно быть массивов и списков значений).  
Столбец – хранит данные только одного типа.

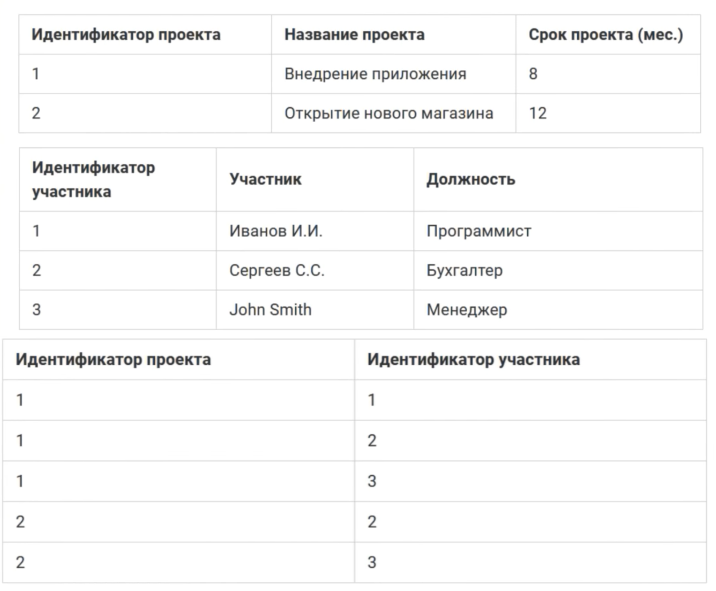


**2НФ (ключи)** –

Таблица должна иметь первичный ключ, для возможности однозначно идентифицировать каждый кортеж (ключ может быть простой или составной).   
Все неключевые столбцы должны зависеть от полного составного ключа (невозможно однозначно определить запись в таблице по части ключа).

Пример неудачного составного ключа:

Пусть составной ключ – Название проекта + участник. Для обеспечения второго свойства, необходимо проверить, можно ли получить должность или срок проекта, зная только часть ключа. В данном случае, зная только название можно определить срок, а зная участника – определить должность – второе свойство не выполняется. Значит нужно прибегнуть к декомпозиции.

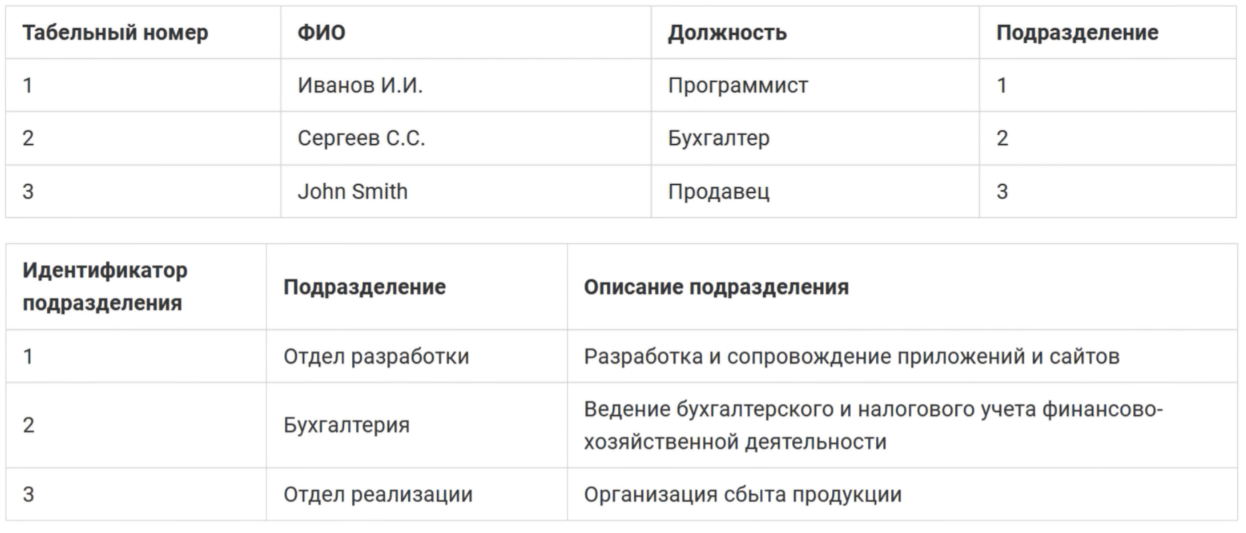


Объединить связанные типы в сущности – участник и должность, проект и срок. И присвоить им простые первичные ключи, так же создать третье отношение, отражающее связь проектов и участников.

**3НФ (неключевые столбцы)** –

Отсутствие транзитивных зависимостей – неключевые столбцы зависят от других неключевых столбцов (невозможно однозначно определить значение одного ключевого столбца исходя из значения другого) – неключевые столбцы не должны играть роль ключа.

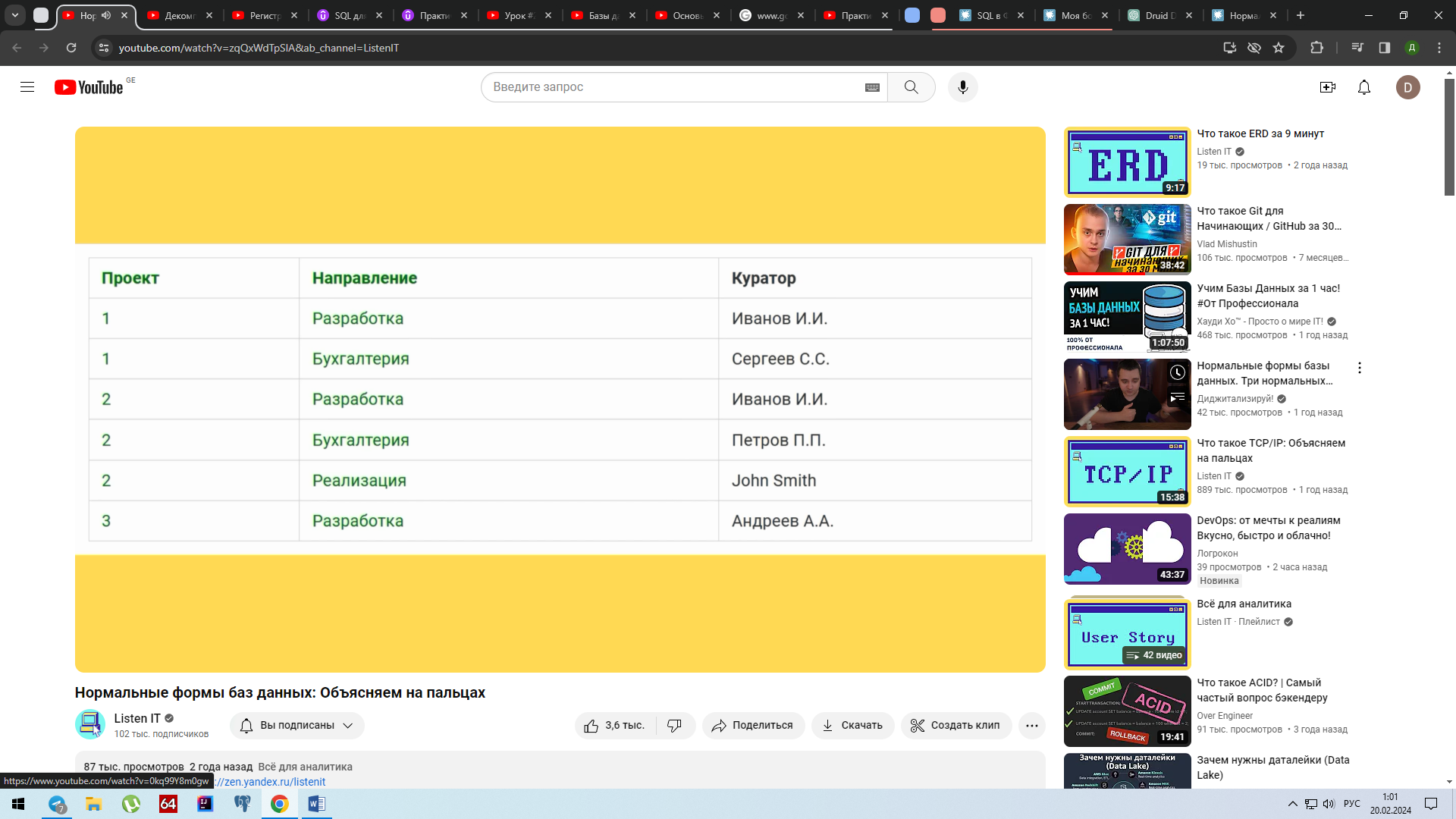
Пример зависимости «Описание подразделения» от «Отдел разработки» - зная отдел разработки или описание подразделения можно однозначно определить соответствующую зависимость.

Решается также декомпозицией:

**Нормальная форма Бойса-Кодда (усиленная 3НФ)** –

Форма, актуальная для отношений с составным первичным ключом. Ключевые атрибуты составного первичного ключа не должны зависеть от неключевых атрибутов (по неключевому атрибуту невозможно однозначно определить часть составного первичного ключа).

Пример зависимости части ключа от неключевого атрибута (направление зависит от куратора).



Решается также декомпозицией



**4НФ (многозначные зависимости) –**

Форма не подразумевает наличие «нетривиальных многозначных зависимостей» (актуально для отношений с более чем 2 атрибутами). Многозначная зависимость – есть атрибуты A, B, C. B и C не связаны между собой, но по отдельности зависят от A. То есть для каждого A есть множество значений B и множество значений C.

Пример многозначной зависимости: Преподаватель и Аудитория не имеют связи, но по отдельности каждый из них зависит от типа Курс.



Удаление преподавателя Иванов приведет к потере типа Аудитория – 101 и 203;

Так же при составлении расписания аудитория может быть известна, а преподаватель не определен, в таком случае тип преподаватель будет равен null или default, что так же является аномалией.



Список хобби студентов, которые посещают курс SQL выводит неверный результат, так как Иванов посещает курс SQL и имеет хобби Футбол и Хоккей, чего нет в результирующей выборке.

*5НФ* –

Таблица соответствует 5НФ, если возможна ее декомпозиция без потерь - разбиение таблицы на несколько и возможность их последующего соединения в точно такую же таблицу (без потерь строк и возникновения новых).

Таблица будет находиться в 5НФ, если она не содержит зависимостей соединения – существование связи между двумя атрибутами, через третий атрибут, который не является ключом.

Невозможно точно определить, находится ли таблица в 5НФ, точно не зная предметной области и всех возможных зависимостей.

**Так же есть доменно-ключевая и 6 нормальные формы** (до них, к сожалению, руки не дошли).

**Минусы сильной нормализации** – теряется понятность данных в таблице и нужно писать более сложные запросы к таблице, снижение производительности СУБД – до 4НФ производительность растет, если использовать 4НФ и последующие, то у такой БД будет проседать производительность за счет увеличения кол-ва таблиц, но при этом уменьшаться кол-во аномалий.

Исходя из этого, нужно искать баланс между удобством работы, ее производительностью и допустимым кол-вом аномалий.

*Денормализация* - пренебрежение формами нормализации для увеличения производительности и упрощения написания запросов.