Темы урока

[**Нормализация Базы Данных**](#_r8dwlw946bhp) **1**

[Чем же полезна нормализация?](#_3wimzsusiiv7) 2

[Нормализация как процесс](#_q0po35ss11pw) 2

[Задача для совместного рассмотрения](#_afb2c61ro7nm) 2

[Выделяем сущности](#_194tzvv3tu4j) 2

[Немного терминов](#_cj8nil6ok1un) 3

[Первая нормальная форма (1НФ)](#_8wsj8hkvpzd4) 3

[Приведение к первой нормальной форме (1НФ)](#_10zoe4jhlsvi) 4

[Ещё немного терминов](#_6vzwt5ikem3k) 4

[Вторая нормальная форма (2НФ)](#_uehokp1jrdvh) 5

[Приведение ко второй нормальной форме (2НФ)](#_co4b1pfaej9g) 5

[Сущность “Отправление”](#_qp4m1832tg3e) 5

[Сущность “Контрагент”](#_mg703w1c460v) 6

[Сущность “Адрес”](#_t2m9sow7mwvy) 6

[Сущность “Статус отправления”](#_d7yzrmsefjjd) 6

[Третья нормальная форма (3НФ)](#_7ou19qqas0te) 7

[Приведение ко второй нормальной форме (2НФ)](#_psi7lm3a34bc) 7

[Отношение “Отправление”](#_oi1dya9o0a00) 7

[Отношение “Контрагент”](#_goo3zg5xkdxa) 7

[Отношение “Должность”](#_hdh0cnzamf5h) 8

[Отношение “Адрес”](#_lb8v7x6qb5ga) 8

[Отношение “Город”](#_t6fm1x8f0ke) 8

[Отношение “Статус отправления”](#_wr5oo0sp2z45) 8

[Отношение “Статус”](#_n2d5zewtohi9) 8

[**SQL-запросы для схемы отношений**](#_lf9jc5h1md8i) **9**

[Пишем SQL-запросы на создание таблиц, пока без первичных ключей](#_30mxe9umq940) 9

[Пишем SQL-запросы на добавление первичных ключей](#_9h99kdzci56t) 9

[Внешний ключ](#_g53mcuiatozf) 9

[Определяем внешние ключи в SQL-запросе](#_t83ymdv9lf1) 9

[**Домашнее задание**](#_xliovyhyekrq) **9**

# Нормализация Базы Данных

Указания для правильного проектирования реляционных баз данных изложены в реляционной модели данных. Они собраны в пять групп, которые называются пятью нормальными формами.

Первая нормальная форма представляет самый низкий уровень нормализации баз данных. Пятый уровень представляет высший уровень нормализации.

Нормальные формы – это рекомендации по проектированию баз данных. Не всегда конкретная задача решается оптимально, если данные приведены к пятой нормальной форме. Тем не менее, рекомендуется нормализовать базу данных в некоторой степени потому, что этот процесс имеет ряд существенных преимуществ с точки зрения эффективности и удобства обращения с вашей базой данных.

## Чем же полезна нормализация?

* Обеспечивает целостность данных (т.е., консистентное состояние)
* Предотвращает появление избыточности в хранимых данных
* Обеспечивает масштабируемость (до определённых пределов)
* Позволяет выполнять запросы на выборку и расчёт данных, используя однотипные подходы

## Нормализация как процесс

* Упорядочивание данных в логические группы или наборы.
* Нахождение связей между наборами данных.
* Минимизация избыточности данных.

На практике к четвёртой и пятой форме базы данных почти никогда не приводят.

Обычно базы данных нормализуются до второй или третьей нормальной формы.

Мы рассмотрим первые три нормальные формы детально на примере.

## Задача для совместного рассмотрения

Спроектировать БД информационной системы управления процессом отправки и получения корреспонденции небольшой организации.

* Описываются отправители и получатели корреспонденции
  + Это люди с именами, должностями и адресами для доставки корреспонденции
* Описывается сама корреспонденция
  + Это бумажные документы с названиями, количеством страниц
* Регистрируется момент времени изменения статуса процесса
  + Возможные статусы
    - ожидает отправки
    - отослано
    - вручено

## Выделяем сущности

* Отправитель
  + ФИО
  + Должность
  + Адрес
* Получатель
  + ФИО
  + Должность
  + Адрес
* Корреспонденция
  + Наименование документа
  + Количество страниц документа
* Статусы отправлений и время
  + Дата и время изменения статуса
  + Статус

Теперь попробуем набросать, как будет выглядеть наша таблица. Удобнее всего для простоты и скорости воспользоваться для этой цели любимы табличным процессором, например Google Spreadsheets. Также заполним её некоторыми данными.

Я подготовил пример такой таблицы: <https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=0>.

В принципе, незадачливый менеджер счёл бы на этом работу готовой, так как она полностью решает поставленные задачи по регистрации и отслеживанию корреспонденции небольшого офиса.

Однако, перед нами стоит более амбициозная задача. Сделать это решение

* целостным,
* оптимизированным с точки зрения избыточности данных,
* масштабируемым,
* легко модифицируемым.

## Немного терминов

Слайд “Снова термины (посложнее)”

* **Атрибут** — свойство некоторой сущности. Часто называется ***полем таблицы***.
* **Домен атрибута** — множество допустимых значений, которые может принимать атрибут.
* **Кортеж** — конечное множество взаимосвязанных допустимых значений атрибутов, которые вместе описывают некоторую сущность (***строка таблицы***).
* **Отношение** — конечное множество кортежей (***таблица***).
* **Схема отношения** — конечное множество атрибутов, определяющих некоторую сущность. Иными словами, это ***структура таблицы***, состоящей из конкретного набора полей.

## Первая нормальная форма (1НФ)

Таблица является отношением в 1НФ, если

1. Отсутствует упорядочивание строк сверху вниз (другими словами, порядок строк не несет в себе никакой информации)
2. Отсутствует упорядочивание столбцов слева направо (другими словами, порядок столбцов не несет в себе никакой информации)
3. Отсутствуют повторяющиеся строки
4. Каждое пересечение строки и столбца содержит ровно одно значение из соответствующего домена (и больше ничего)

## Приведение к первой нормальной форме (1НФ)

В нашем примере первые три пункта выполняются, а вот с последним у нас проблемы. Наше поле “Статусы отправления” содержит множество составных значений.

Преобразуем таблицу к первой нормальной форме. Для этого необходимо:

1. Разбить группу повторяющихся пар даты и статуса на несколько записей так, чтобы в одной записи было только один элемент группы (при этом неразбиваемые атрибуты дублируются)
2. Разнести по разным полям составное значение даты и статуса
3. Выделить город отправителя и город получателя в отдельный атрибут  
   *Адрес можно делить и дальше на улицу, дом и прочее, но в своём примере я остановлюсь на выделении города, так как вижу в этом некий бизнес-смысл. Например, отправления в рамках одного и того же города могут быть выполнены локальной курьерской службой.*

Результат преобразования на странице 1НФ: <https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=232371833>.

## Ещё немного терминов

Перед тем, как переходить ко второй нормальной форме, необходимо ввести ещё немного терминов, слайд “И ещё немного терминов (level:nightmare)”:

* Проекция — отношение, полученное из заданного путём удаления и (или) перестановки некоторых атрибутов.  
    
  Например, отношение (таблица) с только двумя полями “ФИО отправителя” и “Должность отправителя” будет проекцией нашего исходного отношения.
* Функциональная зависимость между атрибутами (множествами атрибутов) X и Y означает, что для любого допустимого набора кортежей в данном отношении: если два кортежа совпадают по значению X, то они совпадают по значению Y.  
    
  Например, если мы посмотрим на нашу таблицу, то должность у получателя в данном примере в каждом кортеже атрибут “Должность отправителя” всегда находится в зависимости от атрибута “ФИО отправителя”.

Теперь можно поговорить о второй нормальной форме.

## Вторая нормальная форма (2НФ)

Отношение находится во 2НФ, если оно находится в 1НФ и

1. Каждый не ключевой атрибут неприводимо зависит от Первичного Ключа (ПК).
2. Неприводимость означает, что в составе потенциального ключа отсутствует меньшее подмножество атрибутов, от которого можно также вывести данную функциональную зависимость.

**Первичный ключ** — это один атрибут или наименьший набор атрибутов, однозначно определяющих один и только один кортеж.

Т.е. все неключевые записи должны однозначно зависеть от первичного ключа.

## Приведение ко второй нормальной форме (2НФ)

Тут можно пойти с нескольких сторон.

Можно вернуться к определению сущностей нашей таблицы выполнить её декомпозицию исходя из того, что сущности как правило разносятся по отдельным таблицам. И в нашем случае у нас есть такие сущности:

* Отправитель (ФИО, Должность)
* Получатель (ФИО, Должность)
* Адрес отправителя (Город, Адрес)
* Адрес получателя (Город, Адрес)
* Отправление (Наименование, количество страниц)
* Статус отправления (Дата перехода в статус, Статус)

Предположим, что это и будут наши таблицы.

Можно пойти с другой стороны, и попытаться определить Первичный Ключ для нашего отношения. Попытаемся определить первичный ключ по имеющимся полям. Наша таблица описывает отправления. Можем отталкиваться от этого и определить первичный ключ для сущности “Отправление”. У нас эта сущность имеет 2 атрибута - “Наименование” и “Количество страниц”. Видно, что у нас нет достойных кандидатов для первичного ключа и мы можем ввести ***суррогатный*** первичный ключ - идентификатор отправления.

### Сущность “Отправление”

Теперь наша **сущность “Отправление”** имеет три атрибута:

1. Идентификатор (атрибуты первичного ключа обычно выносятся в начало),
2. Наименование (слово “отправление” здесь уже не нужно, так как мы говорим о сущности отправления и по умолчанию атрибуты описывают её свойства),
3. Количество страниц

Давайте её вынесем из нашей таблицы в отдельную таблицу и удалим все дубликаты. У нас останется две записи: <https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=811820563>

Теперь выделяем сущности отправителя и получателя. Они очень похожи, у обоих два атрибута: “ФИО” и “Должность”. И, на самом деле, они ничем не отличаются друг от друга. В нашем примере видно, что отправитель и получатель меняются ролями, и это нормально.

Поэтому для начала давайте их объединим в одну сущность: контрагент.

Теперь определим первичный ключ. Конечно, можно попробовать положиться на то, что ФИО уникально, однако, это было бы слишком недальновидно, так что мы введём идентификатор контрагента.

### Сущность “Контрагент”

Теперь наша **сущность “Контрагент”** содержит три атрибута:

1. Идентификатор
2. ФИО
3. Должность

Результат можно посмотреть по ссылке ниже: <https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=1087116955>.

### Сущность “Адрес”

По аналогии выделяем **сущность “Адрес”**, он также будет содержать

1. Идентификатор
2. Город
3. Адрес

Результат можно посмотреть по ссылке ниже: <https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=446512430>.

### Сущность “Статус отправления”

Давайте посмотрим, что осталось в исходной таблице? Это наша сущность “Статус отправления”:

1. Идентификатор отправителя
2. Идентификатор адреса отправителя
3. Идентификатор получателя
4. Идентификатор адреса получателя
5. Идентификатор отправления
6. Статус отправления
7. Дата изменения статуса

Результат можно посмотреть по ссылке ниже: <https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=1566495953>.

Давайте попробуем выделить потенциальный первичный ключ в этом отношении. Ещё раз вспомним, что первичный ключ должен однозначно определять кортеж.

В нашем случае первичным ключом будут все поля, так как их комбинация с различными значениями возможна, с одинаковыми — нет. Во введении идентификатора здесь нет необходимости, так как можно обойтись существующими полями.

Поздравляю, мы создали набор отношений, приведённых ко второй нормальной форме.

## Третья нормальная форма (3НФ)

Отношение находится в 3НФ, когда находится во 2НФ и каждый неключевой атрибут нетранзитивно зависит от первичного ключа. Проще говоря, правило требует выносить все неключевые поля, содержимое которых может относиться к нескольким записям таблицы в отдельные таблицы.

## Приведение ко второй нормальной форме (2НФ)

Ищем потенциально повторяющиеся значения в наших таблицах.

### Отношение “Отправление”

Отношение “Отправление” выглядит удовлетворяющим 3НФ.

<https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=2090189850>

### Отношение “Контрагент”

Отношение “Контрагент” уже выглядит несколько подозрительным с точки 3НФ. В наших данных сейчас этого не видно, но если вы продолжите заполнять этот список, то увидите, что должности рано или поздно начнут повторяться.

А это значит, что мы должны выделить должности в отдельное отношение. При этом в отношении “Контрагент” останется только идентификатор должности.

<https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=1097626516>

### Отношение “Должность”

У нас появляется новое отношение должность состоящее из идентификатора и наименования должности: <https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=315064159>.

### Отношение “Адрес”

Если в уме продолжить заполнять таблицу, то мы скоро начнём всё чаще и чаще повторяться в значении атрибута “Город”. Получается, города надо выносить в отдельное отношение, причём с идентификатором. В будущем мы сможем легко детализировать это отношение новыми атрибутами, например, страной.

В результате отношение “Адрес” будет содержать три атрибута: идентификатор, идентификатор города и адрес с улицей, номером дома и, если нужно, корпусом, квартирой и т.д.

<https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=1130814429>.

### Отношение “Город”

Содержит два атрибута: идентификатор и собственно наименование города:

<https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=1430583329>.

### Отношение “Статус отправления”

В данном отношении видно, что требуется выделить в отдельное отношение наименование статусов. Давайте назовём его просто “Статус”. При этом в исходном отношении соответствующее поле заменится идентификатором:

<https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=1321563603>.

Я бы ещё переставил поля немного таким образом, чтобы наиболее важные поля, по которым будет осуществляться поиск были вначале. Я это делаю, думая о том, в каком порядке буду перечислять поля в первичном ключе.

### Отношение “Статус”

Новое отношение — это просто словарь значений, который по аналогии с городами и должностями будет содержать идентификатор и наименование статуса:

<https://docs.google.com/spreadsheets/d/1DGFBboDA35m4OtGquemqbUt17pFP7ORIMsw924sy4MI/edit#gid=1542878420>.

# SQL-запросы для схемы отношений

## Пишем SQL-запросы на создание таблиц, пока без первичных ключей

Пример можно посмотреть в SQL-скрипте к классной работе L27\_C01\_Сorrespondence.sql.

## Пишем SQL-запросы на добавление первичных ключей

По слайдам.

Показать как создаётся ПК для существующих таблиц.

Дать попробовать.

Показать как создаётся ПК вместе с вновь создаваемой таблицы.

Дать попробовать.

## Внешний ключ

Внешний ключ представляет собой одно из возможных ограничений. Оно ограничивает значения атрибута одного отношения набором значений потенциального ПК другого отношения.

## Определяем внешние ключи в SQL-запросе

По слайдам.

Показать как создаётся ВК для существующих таблиц.

Дать попробовать.

Показать как создаётся ВК вместе с вновь создаваемой таблицы.

Дать попробовать.

# Домашнее задание

Спроектировать и написать SQL-запросы для создания схемы отношений хранилища нашего чат-бота.