GeekBrains

# Знакомство с базами данных

# Оглавление

**[00:01:36]**

# Приветствие

Добрый день, друзья! Рад приветствовать вас на курсе «Знакомство с базами данных». Меня зовут Ильнар Шафигуллин, я методолог программы «Разработчик», кандидат физико-математических наук и преподаватель мехмата Казанского университета. Мой опыт преподавания — более 10 лет.

На этом уроке мы начнём знакомиться с базами данных. Сразу оговорюсь: важно различать концепцию и инструменты для работы с ней. Базы данных — это концепция, а системы управления базами данных (СУБД) — инструмент.

Приведу аналогию: есть математика, и есть калькулятор. Учить базы данных через СУБД — то же самое, что учить математику через калькулятор. Мы хорошо справимся с некоторыми математическими операциями, но все наши знания будут завязаны на инструменте, на том, как решать задачи с его помощью.

На этом курсе мы немного поговорим и про языки, и про программные средства для работы с базами данных. Но основное внимание уделим концепции: посмотрим, почему базы данных работают именно так, что происходит под капотом, как всё это сделать без СУБД.

**[00:03:26]**

# Первые базы данных

Многие думают, что базы данных появились во второй половине XX века, но это не так. Картотеки, библиотеки, книги учёта и результаты переписи населения — всё это базы данных. При этом первая перепись населения прошла в 1897 году.

Получается, базы данных были, а инструменты для удобной и быстрой их обработки — нет. Работать с данными приходилось вручную, поэтому результаты переписи 1897 года появились только к 1905. Процесс обработки информации занял 8 лет.

Сегодня работать с данными можно гораздо быстрее, благодаря компьютерам. Но факт остаётся фактом: базы данных использовались давно, просто по-другому.

**[00:04:52]**

# Функционал баз данных

Базы данных нужны для хранения, обработки и быстрого извлечения информации. Попробуем рассмотреть этот функционал на примере.

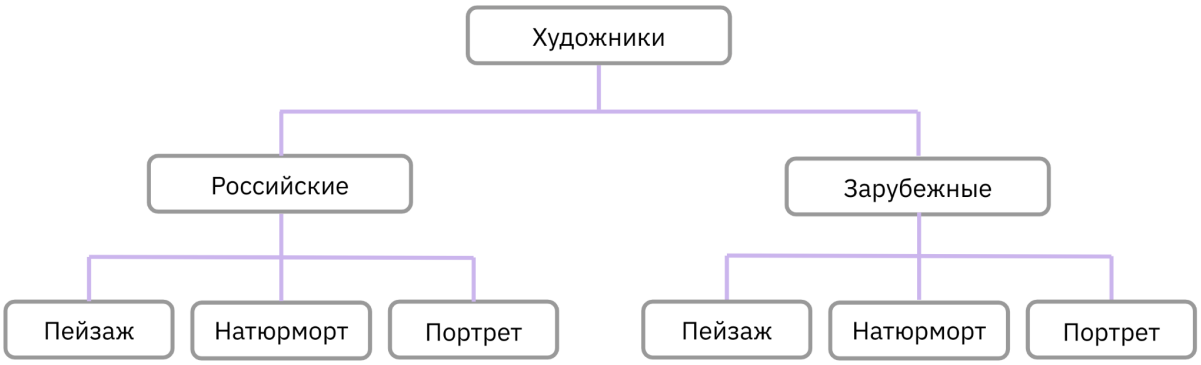
Представим большую картинную галерею. Нам нужно собрать информацию о картинах и каталогизировать её, чтобы было удобно работать. То есть хранить, обрабатывать и быстро извлекать информацию.

Как мы можем организовать информацию? Как будем находить картины определённого художника, жанра или года создания? Давайте попробуем собрать информацию обо всех картинах в каталог.

**[00:06:42]**

# Иерархическая модель

Сперва соберём информацию обо всех художниках и запишем её в базу (наш каталог). Художники будут на вершине иерархии.  Затем информацию о них можно категоризировать: в одной части каталога будут российские художники, в другой — зарубежные. Следующий шаг — выделить разные жанры для картин российских и зарубежных художников: пейзаж, натюрморт, портрет.




Добавим в эту структуру художников. Например, среди российских пейзажистов выделим Шишкина и Айвазовского, а затем внесём информацию об их знаменитых картинах — «Утро в сосновом лесу» и «Девятый вал». Также с зарубежными художниками: натюрморты писал Ван Гог, пример картины — «12 подсолнухов в вазе».

Итак, у нас получился каталог. Можно ли с ним работать? Да, все картины мы можем разложить по этой структуре. Её называют **иерархической**.

Но всё ли в ней в порядке? Полностью ли она нас устраивает? Какие могут быть ограничения? Достаточно простые: художники могут писать в нескольких жанрах, а у картин может быть несколько авторов. Например, у Ван Гога кроме натюрмортов есть известные портреты. Можем положить их в нашу иерархию (для этого выделим ещё одну ячейку).


В этом решении есть сложности. Если нам понадобиться найти все картины Ван Гога, сперва придётся искать его среди зарубежных художников, а затем перебирать разные жанры. Только потом мы сможем получить все картины, которые он написал. Работать с таким каталогом сложно.

Кроме того, у картины может быть несколько авторов. Например, медведей из «Утра в сосновом лесу» писал не Шишкин, а другой художник — Савицкий. Коллекционер Третьяков стёр его подпись, когда покупал картину. Возможно, у Третьякова была иерархическая модель базы данных, и когда он собирал коллекцию, он не мог взять картины нескольких авторов.

Что же делать с такой картиной? Если мы свяжем её сразу с несколькими авторами в каталоге, можем потерять информацию: всех ли авторов мы нашли? Придётся пробежаться по всем картинам и поискать, нет ли у них «Утра в сосновом лесу».

| |
|---|
| При создании иерархической базы данных<br>мы изначально фиксируем сценарий её использования |

В каталоге, который мы создали, удобно искать картины всех российских и зарубежных писателей, а также картины по жанрам. То есть наша иерархическая структура рассчитана на определённый круг задач. Если мы получим задачу, которая не была заложена в структуру, столкнёмся с проблемами.

Можно ли поменять модель так, чтобы решить новую задачу? Да, но это тоже будет решение для частного случая. Мы можем пронумеровать все картины и подготовить для них предметный указатель. Сперва по уже составленной иерархии с российскими / зарубежными

художниками и жанрами. Затем, если у нас будет другая задача, по другой иерархии (в зависимости от сценария использования).

## [00:14:51]

# Задание

Чтобы закрепить знания на практике, выполните небольшое задание: соберите предметный указатель или подготовьте иерархическую структуру, которая позволит находить картины, написанные во второй половине XIX века.

## [00:20:38]

# Вариант решения

У этой задачи нет единственного варианта решения, поэтому рассмотрим самое распространённое.


Мы можем начать каталог с года написания картин. В зависимости от года будем указывать страну художника, потом жанр и другую информацию. Всё, что дальше, нас пока не так интересует. Наша задача — получить картины, написанные в тот или иной год.

**Важно!** Если мы поднимем в иерархии информацию, которая нам нужна в первую очередь, пользоваться каталогом будет проще.

Из такого каталога мы можем взять всю информацию с 1850 по 1900 годы. Но это не просто и не удобно. Иерархическая система долго помогала человечеству структурировать данные, но сейчас, когда у нас есть компьютеры, мы можем посмотреть и на другие модели хранения информации.

**[00:22:29]**

# Задача: телефонный справочник

С иерархической моделью хранения данных мы немного разобрались. Узнали, что она не гибкая, приходится сразу определять сценарий работы с такой базой данных.

Теперь посмотрим, можно ли работать по-другому. Для этого начнём решать задачу — создадим телефонный справочник. Сначала он будет простым, но потом начнёт усложняться: мы будем постепенно решать возникающие проблемы.

Для решения понадобится Excel, Google Таблицы или Open Office. Рекомендуем собирать таблицу параллельно с преподавателем, чтобы понимать, что происходит, и решать возникающие проблемы.

**[00:24:23]**

## Минимальные данные

Минимальные данные, которые нам нужны, — ФИО человека и номер его телефона. Для этого хватит двух столбцов.

| ФИО | Номер телефона |
|---|---|
| Иванов Иван Иванович | 12345678 |
| Петров Пётр Петрович | 12345687 |
| Васильев Василий Васильевич | 12345679 |

Если мне нужен телефон Петрова, я сперва ищу фамилию, а затем смотрю номер телефона, который указан рядом. Кажется, всё просто. Но база данных — это только модель реальности, и реальность не всегда может в неё укладываться. Попробуйте себя в роли тестировщика, подумайте, есть ли какая-то проблема, которую эта таблица не решит?

Думаю, многие догадались: в жизни у нас не один номер телефона. У многих есть личный и рабочий. Как их разместить в таблице? Например, добавить столбец «Дополнительный номер телефона».

| ФИО | Номер телефона | Дополнительный номер телефона |
|---|---|---|
| Иванов Иван Иванович | 12345678 | 87654321 |
| Петров Пётр Петрович | 12345687 | 78654321 |
| Васильев Василий Васильевич | 12345679 | 97654321 |

Итак, проблему с двумя номерами телефонов мы решили: если у кого-то один номер, «дополнительную» ячейку мы можем оставить пустой, если два номера, заполним и её тоже.

Подумайте, какие ещё проблемы могут быть в этой модели, и как их можно решить?

## [00:33:32]

# Увеличиваем количество номеров

Одна из очевидных проблем — номеров может быть больше, чем два: например, личный, рабочий, для поездок, для рекламы в интернете или просто старый, которым мы уже не пользуемся, но хранить информацию хотим. Мы не можем сказать заранее, сколько у человека номеров. Здесь как с массивами: если мы заранее не знаем количество элементов, нам тяжело определить, какой массив будет нужен для хранения информации.

Как решить эту проблему? Сделаем таблицу, которая позволяет хранить больше номеров.

| ФИО | Телефон 1 | Телефон 2 | Телефон 3 | Телефон 4 | Телефон 5 |
|---|---|---|---|---|---|
| Иванов Иван Иванович | 123 | 124 | 125 | 126 | 127 |
| Петров Пётр Петрович | 234 | 231 | - | - | - |
| Васильев Василий Васильевич | 456 | - | - | - | - |

Из-за Иванова с пятью номерами получилась таблица с множеством пустых ячеек. Это тоже проблема — много пространства мы тратим впустую. Решение рабочее, но не такое хорошее, как мы хотели.

Кроме того, название столбцов отличаются только номерами. Непонятно, какой из них личный, какой рабочий, а какой для поездок. Можно переименовать их, но это тоже не 100% решение: мы не можем предусмотреть все ситуации использования этих номеров. Хорошо бы хранить для каждого из них комментарий. Как это сделать?

## [00:36:48]

# Избавляемся от пустых ячеек

Раньше каждому человеку соответствовал какой-то номер или набор номеров. Сейчас я сделаю по-другому: запишу ФИО человека столько раз, сколько у него телефонов и к каждому телефону добавлю комментарий. То есть таблица будет расти не вширь, а вниз.

| ФИО | Телефон | Комментарий |
|---|---|---|
| Иванов Иван Иванович | 123 | личный |

| | | |
|---|---|---|
| Иванов Иван Иванович | 124 | рабочий |
| Иванов Иван Иванович | 125 | для поездок |
| Иванов Иван Иванович | 126 | для объявлений в интернете |
| Иванов Иван Иванович | 127 | старый, возможно, неактуальный |
| Петров Пётр Петрович | 234 | личный |
| Петров Пётр Петрович | 235 | рабочий |
| Васильев Василий Васильевич | 456 | личный |

Теперь я могу ввести в поиск фамилию человека и выбрать нужный телефон из набора его номеров. Ориентироваться буду по комментарию.

Такое решение можно использовать для хранения неограниченного количества номеров для каждого человека. Мы решили часть проблем: таблица более сжатая, в ней нет пустых ячеек, мы добавили комментарии, так что работать с телефонами теперь будет удобнее.

## [00:38:42]

# Работаем с дублями

Но проблемы в таблице ещё остались. Как вы думаете, какие?

Если мы дополним информацию о человеке, появится много дублей. Объём данных, который мы будем хранить, сильно увеличится:

| ФИО | Телефон | Комментарий | Адрес | Д/р | Статус |
|---|---|---|---|---|---|
| Иванов И. И. | 123 | личный | Казань | 12.02.1990 | женат |
| Иванов И. И. | 124 | рабочий | Казань | 12.02.1990 | женат |
| Иванов И. И. | 125 | для поездок | Казань | 12.02.1990 | женат |
| Иванов И. И. | 126 | интернет | Казань | 12.02.1990 | женат |

Если кроме телефона мы захотим хранить ещё адрес, дату рождения и семейное положение, в каждую строку с новым номером человека эту информацию придётся копировать. В примере адрес, день рождения и статус Иванова повторяется 4 раза, от записи к записи они никак не меняются.

Из программирования вы уже знаете термин «захардкорженная информация». Если мы что-то написали руками, а не сделали с помощью переменной, нужно внимательно следить: если условия задачи поменяются, менять придётся всё.

Представим ситуацию, что Иванов развёлся. Тогда в каждой записи придётся исправить его семейное положение. Если мы это сделаем только в одном месте, возникнет путаница. Получается, такая форма хранения информации добавляет нам много сложностей даже за исключением того, что её нужно дублировать каждый раз.

Кроме того, адресов у человека тоже может быть несколько: например, по прописке или фактический. Соответственно, помимо нескольких записей для хранения номеров телефонов добавятся записи с адресами. Информации о человеке тоже можно добавить больше, чем на 3 столбца.

Таблица сильно разрастается, следить за ней становится всё сложнее. Если мы поменяем фактический адрес или семейное положение, нужно будет в каждой копии записи внести изменение.

Пожалуйста, заполните ваши таблицы, чтобы мы видели всю картину. Добавьте ещё 3 столбца и внесите информацию о каждом персонаже. Посмотрите, какой объёмной стала таблица.

### [00:47:10]

## Разбиваем таблицу

Что можно сделать с этой проблемой? Разбить информацию на несколько таблиц и вынести в них всю повторяющуюся информацию:

| Чей телефон | Телефон | Комментарий |
|---|---|---|
| Иванов И. И. | 123 | личный |
| Иванов И. И. | 124 | рабочий |
| Иванов И. И. | 125 | для поездок |
| Иванов И. И. | 126 | интернет |
| Иванов И. И. | 127 | старый |

| | | |
|---|---|---|
| Петров П. П. | 234 | личный |
| Петров П. П. | 235 | рабочий |
| Васильев В. В. | 456 | личный |

*Дополнительная таблица*

Мы выделили столбцы «Чей телефон» (в нём указали ФИО людей), «Телефон» и «Комментарий». По сути, это та таблица, которая у нас была. В ней содержится минимальная информация, которую целесообразно дублировать.

В основной таблице останется информация о персонажах: их адреса, дни рождения и семейное положение.

| ФИО | Адрес | Д/р | Статус |
|---|---|---|---|
| Иванов И. И. | Казань | 12.02.1990 | женат |
| Петров П. П. | Москва | 23.04,1983 | женат |
| Васильев В. В. | Белгород | 21.05.1998 | холост |

*Основная таблица*

Основная таблица сильно уменьшилась в размерах после того, как мы убрали дублирующуюся информацию. Осталось всего три контакта. Если нам нужно найти номер конкретного человека, воспользуемся дополнительной таблицей.

Если вы попробуете экстраполировать ситуацию, то есть посмотрите, что случится, если мы будем увеличивать количество людей и номеров телефонов, вы увидите, что наше предыдущее решение будет очень разрастаться, а в последнее — нет. В нём информация будет добавляться по минимуму. С такой таблицей уже можно работать вручную.

## [00:50:52]

# Добавляем идентификаторы

Но у этого решения тоже есть проблемы. Как вы думаете, какие? И как их можно решить?

Сложность в том, что может быть несколько однофамильцев или полных тёзок. Приведу пример: добавлю в основную таблицу ещё одного Иванова Ивана Ивановича. Но он будет жить не в Казани, а в Санкт-Петербурге, у него будет другой день рождения и семейное положение.

| ФИО | Адрес | Д/р | Статус |
|---|---|---|---|
| Иванов И. И. | Казань | 12.02.1990 | женат |
| Иванов И. И. | Санкт-Петербург | 18.09.2001 | холост |
| Петров П. П. | Москва | 23.04,1983 | женат |
| Васильев В. В. | Белгород | 21.05.1998 | холост |

Если оставить всё как есть ни я, ни компьютер не сможем разобраться, какой номер телефона относится к одному Иванову, а какой — ко второму.

| Чей телефон | Телефон | Комментарий |
|---|---|---|
| Иванов И. И. | 123 | личный |
| Иванов И. И. | 124 | рабочий |
| Иванов И. И. | 125 | для поездок |
| Иванов И. И. | 126 | интернет |
| Иванов И. И. | 127 | старый |
| Иванов И. И. | 527 | личный |
| Петров П. П. | 234 | личный |
| Петров П. П. | 235 | рабочий |
| Васильев В. В. | 456 | личный |

Если бы мы использовали прошлое решение, можно было бы найти нового человека по адресу. Но мы уже разбили таблицу надвое и будем с этим работать.

Чтобы решить проблему, добавим уникальные идентификаторы. Пронумеруем людей в списке: id — число, которое будет своим для каждой записи. У нас четыре человека, значит будет четыре номера.

| id | ФИО | Адрес | Д/р | Статус |
|---|---|---|---|---|
| 1 | Иванов И. И. | Казань | 12.02.1990 | женат |
| 2 | Иванов И. И. | Санкт-Петербург | 18.09.2001 | холост |
| 3 | Петров П. П. | Москва | 23.04,1983 | женат |
| 4 | Васильев В. В. | Белгород | 21.05.1998 | холост |

Теперь в столбце «Чей телефон» будем указывать не ФИО, а идентификатор. Идентификатор будет связывать две таблицы.

| Чей телефон | Телефон | Комментарий |
|---|---|---|
| 1 | 123 | личный |
| 1 | 124 | рабочий |
| 1 | 125 | для поездок |
| 1 | 126 | интернет |
| 1 | 127 | старый |
| 2 | 527 | личный |
| 3 | 234 | личный |
| 3 | 235 | рабочий |
| 4 | 456 | личный |

Если нам понадобится найти номер Иванова из Санкт-Петербурга, сперва нужно будет узнать его идентификатор в основной таблице (id 2), а затем посмотреть, какой номер ему соответствует в связанной таблице. Так мы решим проблему тёзок.

**[01:02:06]**

## Первичный и внешний ключи

Столбик с идентификатором в основной таблице — это первичный ключ. Так называют уникальную информацию, которая помогает нам идентифицировать каждую конкретную запись в таблице.

Столбик «Чей телефон» в дополнительной таблице — это внешний ключ. Он не уникален, но по нему можно найти нужную запись.

Подробнее о ключах мы будем говорить на будущих курсах. Сейчас самое важное — понять, что если мы связываем несколько таблиц, нужен уникальный идентификатор, чтобы понимать, какая запись где находится.

С одной стороны, мы получили огромную пользу от того, что разбили информацию на разные таблицы, но, с другой стороны, нельзя терять связи между данными, которые в этих таблицах хранятся.

**[01:03:18]**

# Реляционные базы данных

Мы решили задачу: разбили одну большую таблицу с неуникальной информацией на две, в которых нет повторов. По сути, и первое и второе решения позволяют работать с информацией одинаково, но физически занимают разное количество ячеек. Первое решение более громоздкое. Если мы воспользуемся вторым, сэкономим память и ресурсы.

Мы создали связи между таблицами и получили **реляционную базу данных** — базу, в которой данные распределены по отдельным связанным между собой таблицам.

Мы связали две таблицы, но ограничиваться ими на обязательно. У человека может быть несколько адресов, и их тоже можно вынести в новую таблицу.

Попробуйте выделить третью таблицу с адресами, связать её с основной по первичному ключу и поискать информацию по конкретному человеку.

**[01:05:05]**

# Заключение

Итак, попробуйте добавить в базу данных третью таблицу с адресами и посмотреть, как она будет работать. Задание простое, но оно поможет закрепить все те шаги, которые мы прошли сегодня на уроке.

На следующей лекции мы разберём, как получать информацию из разных связанных таблиц.

Всем большое спасибо, увидимся на следующих лекциях.