**1.1 Отношение (таблица)**

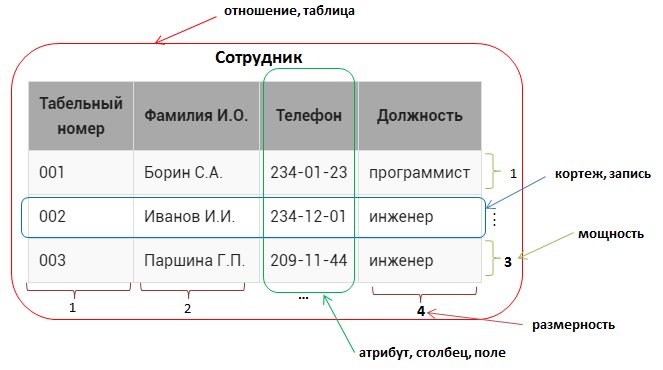
**Основные понятия реляционных баз данных**

Реляционная модель была разработана в конце 1960-х годов Е.Ф.Коддом . Она определяет способ представления данных (структуру данных), методы защиты данных (целостность данных), и операции, которые можно выполнять с данными (манипулирование данными). Эта модель лежит в основе всех реляционных баз данных до настоящего времени.

**Основные принципы реляционных баз данных:**

* все данные на концептуальном уровне представляются в виде объектов, заданных в виде строк и столбцов, называемых отношением, более распространенное название – таблица;
* в пересечение строки и столбца таблицы можно занести только одно значение;
* все операции выполняются над целыми отношениями и результатом этих операций является отношение.

Пример отношения:



На примере таблицы **Сотрудник** рассмотрим **терминологию реляционных баз данных:**

* ***отношение***  – это структура данных целиком, набор записей (в обычном понимании – таблица) , в  примере –это **Сотрудник**;
* ***кортеж***– это каждая строка , содержащая данные (более распространенный термин – запись ), например, <00*1, Борин С.А, 234-01-23, программист*>, все кортежи в отношении должны быть различны;
* ***мощность***– число кортежей в таблице (проще говоря, число записей), в данном случае 3, мощность отношения может быть любой (от 0 до бесконечности), порядок следования кортежей - неважен;
* ***атрибут*** – это столбец в таблице (более распространенный термин – поле ), в примере – **Табельный номер, Фамилия И.О., Телефон, Должность**)
* ***размерность*** – это число атрибутов в таблице, в данном случае – 4;
* размерность отношения должна быть больше 0, порядок следования атрибутов существенен;
* ***домен атрибута***– это допустимые значения (неповторяющиеся), которые можно занести в поле , например для атрибута **Должность** домен – {инженер, программист}.

**Задание**

Для таблицы **Сотрудник** отметьте верные ячейки (В каждом столбце и строке - один правильный ответ).

**Отметьте верные ячейки**

| **Ряды:** | **Домен** | **Атрибут** | **Отношение** | **Размерность** | **Кортеж** | **Мощность** |
| --- | --- | --- | --- | --- | --- | --- |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |
| Сотрудник |  |  |  |  |  |  |
| {###-##-##}, где # - цифра от 0 до 9 |  |  |  |  |  |  |
| Фамилия И.О. |  |  |  |  |  |  |
| 002, Иванов И.И., 234-12-01, инженер |  |  |  |  |  |  |

**Отношение, реляционная модель**

База данных, в том числе и реляционная, используется для формального описания  некоторой предметной области реального мира, например, склада, учебного процесса и пр. Обязательным этапом перед созданием базы данных является ее проектирование (этот процесс разбирается в следующих модулях).

В первом модуле будем рассматривать простейшие предметные области, информацию о которых можно описать в виде одной таблицы. Каждая такая таблица ассоциируется с неким информационным объектом или событием реального мира – человеком, документом, посещением и т.д.

**Пример.**

Рассмотрим некоторый склад, на котором хранятся книги. Известно название книги, ее автор, количество экземпляров на складе и ее цена.

Всю эту информацию можно представить в виде таблицы, состоящей из 4 столбцов (приведено только 4 записи, на самом деле их значительно больше):

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Автор** | **Цена, руб** | **Количество** |
| Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |
| Белая гвардия | Булгаков М.А. | 540.50 | 5 |
| Идиот | Достоевский Ф.М. | 460 | 10 |
| Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

Перед созданием таблицы в базе данных необходимо описать ее структуру. Для этого выполняется следующая последовательность шагов:

1. Дать таблице имя, пусть она будет называться **book**, вот некоторые ***правила для выбора имен таблиц***:

* может включать английские буквы, цифры и знак подчеркивания, должно начинаться с буквы;
* имя должно быть уникальным в пределах базы данных.

Также **рекомендуется**:

* чтобы имя было существительным в единственном числе;
* имя должно быть понятным и соответствовать тому объекту, который оно описывает;
* имя должно быть как можно короче, максимум до 10 символов.

**Важно.** Имена таблиц являются регистрозависимыми из-за операционной системы на которой работает **stepik**, то есть имя **book** и **Book** – разные имена. Рекомендуется для записи имен таблиц использовать только строчные (маленькие) буквы.

2. Определить структуру таблицы, из каких атрибутов(столбцов, полей) она будет состоять,  в нашем случае это:

* **title** – поле для хранения названия книги;
* **author** – поле с фамилией автора книги ;
* **priсe** – цена книги;
* **amount** – количество книг.

***Правила по выбору имени поля информационного объекта:***

* может включать английские буквы, цифры и знак подчеркивания, должно начинаться с буквы;
* имя поля должно быть уникальным в пределах таблицы.

***Рекомендации по выбору имени поля информационного объекта:***

* имя должно быть понятным и соответствовать тем данным, которые хранятся в поле;
* имя может состоять из нескольких слов, тогда слова разделяются подчеркиванием, после подчеркивания слово пишется с маленькой буквы.

3. Включить ключевое поле **book\_id**, которое является ОБЯЗАТЕЛЬНЫМ ЭЛЕМЕНТОМ каждой реляционной таблицы. Ключевое поле является уникальным для каждой записи, однозначно определяет запись и в дальнейшем будет использоваться для связей с другими таблицами.

***Рекомендации по именованию  ключевых полей:***

* имя должно состоять  из двух частей: начинаться с названия таблицы, которой поле принадлежит,   затем через подчеркивание необходимо указать **id.**

Таким образом, наша таблица**book**будет выглядеть следующим образом:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author** | **price** | **amount** |
| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |
| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |
| 3 | Идиот | Достоевский Ф.М. | 460 | 10 |
| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

**Задание**

Отметьте ПРАВИЛЬНЫЕ имена, которые можно выбрать в качестве названий таблиц или полей.

**Выберите все подходящие ответы из списка**

name -author price(rub) name\_book store price\_rub book store

**Выбор типов данных для полей**

После описания структуры таблицы необходимо выбрать типы данных для каждого поля.

**Основные типы данных SQL:**

|  |  |  |
| --- | --- | --- |
| **Тип данных** | **Описание** | **Пример** |
| INT  INTEGER | Целое число, могут принимать значения от -2 147 483 648 до 2 147 483 647 | -567 1205 |
| DECIMAL NUMERIC | Вещественное число, в скобках указывается максимальная длина числа (включает символы слева и справа от десятичной запятой) и количество знаков после запятой. Можно использовать оба этих типа, они эквивалентны, принимают значения в диапазоне -1038+1 до 1038-1. DECIMAL(4,1)  NUMERIC(6,3) | 34.6 -3.294 |
| DATE | Дата в формате ГГГГ-ММ-ДД  *26 июля 2020 года 3 января 2021 года* | 2020-07-26 2021-01-03 |
| VARCHAR | Строка длиной не более 255 символов, в скобках указывается максимальная длина строки, которая может храниться в поле VARCHAR(10)(рассматриваются однобайтовые кодировки, для которых число в скобках соответствует максимальному количеству символов в строке) | пример описание |

***Рекомендации по выбору типов данных для полей таблицы.***

* Выбирайте минимальный тип данных исходя из максимального значения поля. Например, если максимальный текст, который может быть записан в поле, имеет длину 25 символов, значит нужно использовать тип VARCHAR(25).
* Для описания ключевого поля используйте описание INT PRIMARY KEY AUTO\_INCREMENT. Это значит, что в поле будут заноситься различные целые числа, при этом они будут автоматически генерироваться (каждая следующая строка будет иметь значение ключа на 1 больше предыдущего).

Определим тип данных для каждого поля таблицы **book**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author** | **price** | **amount** |
| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |
| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |
| 3 | Идиот | Достоевский Ф.М. | 460 | 10 |
| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

* **book\_id** - ключевой столбец, целое число, которое должно генерироваться автоматически  -INT PRIMARY KEY AUTO\_INCREMENT;
* **title** - строка текста, ее длина выбирается в зависимости от данных, которые предполагается хранить в поле, предположим, что название книги не превышает 50 символов - VARCHAR(50);
* **author** - строка текста - VARCHAR(30);
* **priсe** - для описание денежного значения используется числовой тип данных с двумя знаками после запятой - DECIMAL(8,2);
* **amount** - целое число -INT.

**Задание**

Сопоставьте значения и типы данных, с помощью которых их можно описать.

**Сопоставьте значения из двух списков**

5.12 DECIMAL(5, 2)

-0.99  
10.12

56.1 DECIMAL(7, 1)

456.1  
3000

-34078 INT

55  
0

sql VARCHAR(5)

mySql  
data

Слово VARCHAR(11)

Предложение  
Абзац

На этом шаге нужно написать и проверить SQL запрос. Сначала кратко описывается структура и особенности запроса, приводится пример. А затем формулируется задание, для которого нужно реализовать запрос.

**Создание таблицы**

Для создания таблицы используется SQL-запрос. В нем указывается какая таблица создается, из каких атрибутов(полей) она состоит и какой тип данных имеет каждое поле, при необходимости указывается описание полей (ключевое поле и т.д.). Его структура :

* ключевые слова : CREATE TABLE
* имя создаваемой таблицы;
* открывающая круглая скобка «(»;
* название поля и его описание, которое включает тип поля и другие необязательные характеристики;
* запятая;
* название поля и его описание;
* ...
* закрывающая скобка «)».

**Пример.** Создадим таблицу **genre** следующей структуры:

|  |  |
| --- | --- |
| **Поле** | **Тип, описание** |
| genre\_id | INT PRIMARY KEY AUTO\_INCREMENT |
| name\_genre | VARCHAR(30) |

*Запрос:*

CREATE TABLE genre(

genre\_id INT PRIMARY KEY AUTO\_INCREMENT,

name\_genre VARCHAR(30)

);

Созданная таблица - пустая.

***Рекомендации по записи SQL запроса***

* Ключевые слова: SQL не является регистрозависимым языком (CREATE и create - одно и тоже ключевое слово).
* Ключевые слова SQL и типы данных рекомендуется  записывать прописными (большими) буквами.
* Имена таблиц и полей - строчными (маленькими) буквами.
* SQL-запрос можно писать на нескольких строках.
* В конце SQL-запроса ставится точка с запятой (хотя если Вы пишете один запрос, это необязательно).

**Задание**

Сформулируйте SQL запрос для создания таблицы **book**, занесите  его в окно кода (расположено ниже)  и отправьте на проверку (кнопка **Отправить**). Структура таблицы **book**:

|  |  |
| --- | --- |
| **Поле** | **Тип, описание** |
| book\_id | INT PRIMARY KEY AUTO\_INCREMENT |
| title | VARCHAR(50) |
| author | VARCHAR(30) |
| price | DECIMAL(8, 2) |
| amount | INT |

**Пояснение** - При записи сохраняйте порядок следования полей.

**Введите SQL запрос**

CREATE TABLE book(

book\_id INT PRIMARY KEY AUTO\_INCREMENT,

title VARCHAR(50),

author VARCHAR(30),

price DECIMAL(8,2),

amount INT

);

На каждом шаге можно посмотреть, как  работает запрос примера. Для этого нужно скопировать его код в окно для ввода и нажать на черную кнопку **Запустить код** (не отправляя на проверку). Те запросы, которые уже проверены, можно не удалять, а просто закомментировать, используя /\* и \*/.

**Вставка записи в таблицу**

Для занесения новой записи в таблицу используется SQL запрос, в котором указывается в какую таблицу, в какие поля заносить новые значения. Структура запроса:

* ключевые слова INSERT INTO (ключевое слово INTO можно пропустить);
* имя таблицы, в которую добавляется запись;
* открывающая круглая скобка «(»;
* список полей через запятую, в которые следует занести новые данные;
* закрывающая скобка «)»;
* ключевое слово VALUES;
* открывающая круглая скобка «(»;
* список значений через запятую, которые заносятся в соответствующие поля, при этом текстовые значения заключаются в кавычки, числовые значения записываются без кавычек, в качестве разделителя целой и дробной части используется точка;
* закрывающая скобка «)».

**Пример.** В **таблицу**, состоящую из двух столбцов добавим новую строку, при этом в **поле1** заносится **значение1**,  в **поле2** - **значение2**.

INSERT INTO таблица(поле1, поле2)

VALUES (значение1, значение2);

В результате выполнения запроса новая запись заносится в конец обновляемой таблицы.

При составлении списка полей и списка значений необходимо учитывать следующее:

1. количество полей и количество значений в списках должны совпадать;
2. должно существовать прямое соответствие между позицией одного и того же элемента в обоих списках, поэтому первый элемент списка значений должен относиться к первому столбцу в списке столбцов, второй – ко второму столбцу и т.д.;
3. типы данных элементов в списке значений должны быть совместимы с типами данных соответствующих столбцов таблицы ( целое число можно занести в поле типа DECIMAL, обратная операция - недопустима);
4. новые значения нельзя добавлять в поля, описанные как PRIMARY KEY AUTO\_INCREMENT;
5. рекомендуется заполнять все поля записи, если же поле пропущено, значение этого поля зависит от установленных по умолчанию значений, если значения не установлены - на данной платформе вставляется пустое значение (**NULL**).

**Пример**

Вставим новую запись в таблицу **genre**, созданную на предыдущем шаге ( в первых двух строках показана структура таблицы, далее - ее содержимое):

|  |  |
| --- | --- |
| **genre\_id** | **name\_genre** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(30)** |
| 1 | Роман |

*Запрос:*

INSERT INTO genre (name\_genre)

VALUES ('Роман');

Заносится только значение поля **name\_genre**, значение ключевого поля формируется автоматически.

*Результат*:  в таблицу будет вставлена новая строка, после запуска запроса на платформе **stepik**, имеем:



Чтобы увидеть как именно выглядит таблица **genre**, можно добавить SQL запрос, который выберет все записи из таблицы:

SELECT \* FROM genre;

*Результат:*



**Задание**

Занесите новую строку в таблицу **book** (текстовые значения (тип **VARCHAR**) заключать либо в двойные, либо в одинарные кавычки):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author** | **price** | **amount** |
| INT PRIMARY KEY AUTO\_INCREMENT | VARCHAR(50) | VARCHAR(30) | DECIMAL(8,2) | INT |
| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

**Рекомендация:** текстовые поля копируйте из таблицы, представленной в задании, и вставляйте в запрос во избежание ошибок...

*Результат:*

Affected rows: 1

Query result:

+---------+--------------------+---------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+--------------------+---------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

+---------+--------------------+---------------+--------+--------+

**Введите SQL запрос**

INSERT INTO book (title, author, price, amount)

VALUES ('Мастер и Маргарита', 'Булгаков М.А.', 670.99, 3);

## Задание

Занесите три последние записи в таблицу**book**,  первая запись уже добавлена на предыдущем шаге:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author** | **price** | **amount** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(50)** | **VARCHAR(30)** | **DECIMAL(8,2)** | **INT** |
| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |
| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |
| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |
| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

**Пояснение**- Каждая строка вставляется отдельным SQL запросом, запросы обязательно разделять точкой с запятой. Для просмотра полученной таблицы после запросов на добавление записей вставить:

SELECT \* FROM book;

Результат :

Affected rows: 1

Affected rows: 1

Affected rows: 1

Query result:

+---------+--------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+--------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

+---------+--------------------+------------------+--------+--------+

### Введите SQL запрос

INSERT INTO book (title, author, price, amount) VALUES

('Белая гвардия', 'Булгаков М.А.', 540.50, 5),

('Идиот', 'Достоевский Ф.М.', 460.00, 10),

('Братья Карамазовы', 'Достоевский Ф.М.', 799.01, 2)

;

SELECT \* FROM book;

1.2 Выборка данных

**Содержание урока**

В этом уроке будет рассмотрен синтаксис и семантика следующих SQL запросов:

* [выборка всех данных из таблицы](https://stepik.org/lesson/297509/step/2?unit=279269);
* [выборка данных из отдельных столбцов](https://stepik.org/lesson/297509/step/3?unit=279269);
* [выборка отдельных столбцов и присвоение им новых имен](https://stepik.org/lesson/297509/step/4?unit=279269);
* [создание вычисляемых столбцов;](https://stepik.org/lesson/297509/step/5?unit=279269)
* [вычисляемые столбцы, математические функции](https://stepik.org/lesson/297509/step/6?unit=279269);
* [вычисляемые столбцы, логические функции](https://stepik.org/lesson/297509/step/7?unit=279269);
* [выборка данных по простому условию](https://stepik.org/lesson/297509/step/8?unit=279269);
* [выборка данных с использованием логических выражений и операций](https://stepik.org/lesson/297509/step/9?unit=279269);
* [выборка данных, операторы BETWEEN, IN;](https://stepik.org/lesson/297509/step/10?unit=279269)
* [выборка текстовых данных по шаблону, оператор LIKE](https://stepik.org/lesson/297509/step/11?unit=279269);
* [выборка данных с сортировкой.](https://stepik.org/lesson/297509/step/12?unit=279269)

**Структура и наполнение таблицы**

Все запросы будут формулироваться для таблицы **book**([создание](https://stepik.org/lesson/297508/step/8?unit=279268), [заполнение](https://stepik.org/lesson/297508/step/9?unit=279268)):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author** | **price** | **amount** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(50)** | **VARCHAR(30)** | **DECIMAL(8,2)** | **INT** |
| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |
| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |
| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |
| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |
| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

**Выборка всех данных из таблицы**

Для того чтобы отобрать все данные из таблицы используется SQL запрос следующей структуры:

* ключевое слово **SELECT**;
* символ « **\***» ;
* ключевое слово **FROM**;
* имя таблицы.

Результатом является таблица, в которую включены все строки и столбцы указанной в запросе таблицы.

**Пример**

Выбрать все записи таблицы book .

*Запрос:*

SELECT \* FROM book;

**Задание**

Вывести информацию о всех книгах, хранящихся на складе.

Для этого:

* Напишите SQL запрос в окне кода;
* Отправьте на проверку (кнопка  **Отправить**);
* Если запрос работает неверно, исправьте его и снова отправьте на проверку.

**Важно!** В окне кода можно использовать комментарии для сохранения разных вариантов запросов или пояснений. Комментарии заключаются в **/\*** и **\*/**

**Результат:**

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

+---------+-----------------------+------------------+--------+--------+

Affected rows: 5

**Введите SQL запрос**

SELECT \* FROM book;

**Выборка отдельных столбцов**

Для того чтобы отобрать данные из определенных столбцов таблицы используется SQL запрос следующей структуры:

* ключевое слово **SELECT** ;
* список столбцов таблицы через запятую;
* ключевое слово **FROM** ;
* имя таблицы.

Результатом является таблица, в которую включены все данные из указанных после **SELECT** столбцов исходной таблицы.

**Пример**

Выбрать названия книг и их количества из таблицы **book** .

*Запрос:*

SELECT title, amount FROM book;

*Результат:*

+-----------------------+--------+

|title | amount |

+-----------------------+--------+

| Мастер и Маргарита | 3 |

| Белая гвардия | 5 |

| Идиот | 10 |

| Братья Карамазовы | 2 |

| Стихотворения и поэмы | 15 |

+-----------------------+--------+

**Пояснение.** Чтобы посмотреть, как работает запрос примера, скопируйте его код в окно для ввода и нажмите на черную кнопку **Запустить код**. После запуска выведется результат запроса, который можно сравнить с приведенным образцом. Задание

**Задание**

Выбрать авторов, название книг и их цену из таблицы **book**.

*Результат:*

+------------------+-----------------------+--------+

| author | title | price |

+------------------+-----------------------+--------+

| Булгаков М.А. | Мастер и Маргарита | 670.99 |

| Булгаков М.А. | Белая гвардия | 540.50 |

| Достоевский Ф.М. | Идиот | 460.00 |

| Достоевский Ф.М. | Братья Карамазовы | 799.01 |

| Есенин С.А. | Стихотворения и поэмы | 650.00 |

+------------------+-----------------------+--------+

Affected rows: 5

**Введите SQL запрос**

SELECT author, title, price FROM book;

**Выборка новых столбцов и присвоение им новых имен**

Для того чтобы отобрать данные из определенных столбцов таблицы и одновременно задать столбцам новые имена используется SQL запрос следующей структуры:

* ключевое слово **SELECT** ;
* имя столбца;
* ключевое слово **AS** ;
* новое название столбца (можно русскими буквами), но это должно быть одно слово, если название состоит из двух слов – соединяйте их подчеркиванием, например, **Количество\_книг** ;
* запятая;
* имя столбца;
* .... •
* ключевое слово **FROM** ;
* имя таблицы.

В одном запросе можно использовать и имена столбцов из таблицы, и новые названия.

Результатом является таблица, в которую включены все данные из указанных после **SELECT** столбцов исходной таблицы. Каждому столбцу присваивается новое имя, заданное после **AS**, или столбец получает имя столбца исходной таблицы, если **AS** отсутствует.

**Пример**

Выбрать все названия книг и их количества из таблицы **book** , для столбца **title** задать новое имя **Название**.

*Запрос:*

SELECT title AS Название, amount

FROM book;

*Результат:*

+-----------------------+--------+

| Название | amount |

+-----------------------+--------+

| Мастер и Маргарита | 3 |

| Белая гвардия | 5 |

| Идиот | 10 |

| Братья Карамазовы | 2 |

| Стихотворения и поэмы | 15 |

+-----------------------+--------+

**Задание**

Выбрать названия книг и авторов из таблицы **book**, для поля **title** задать имя(псевдоним) **Название**, для поля **author** –  **Автор**.

*Результат:*

+-----------------------+------------------+

| Название | Автор |

+-----------------------+------------------+

| Мастер и Маргарита | Булгаков М.А. |

| Белая гвардия | Булгаков М.А. |

| Идиот | Достоевский Ф.М. |

| Братья Карамазовы | Достоевский Ф.М. |

| Стихотворения и поэмы | Есенин С.А. |

+-----------------------+------------------+

Affected rows: 5

**Структура и наполнение таблицы book:**

**Введите SQL запрос**

SELECT title AS Название, author AS Автор FROM book;

## Выборка данных с созданием вычисляемого столбца

С помощью SQL запросов можно осуществлять вычисления по каждой строке таблицы с помощью вычисляемого столбца. Для него в списке полей после оператора **SELECT** указывается выражение и задается имя.

Выражение может включать имена столбцов, константы, знаки операций, встроенные функции.

Результатом является таблица, в которую включены все данные из указанных после **SELECT** столбцов, а также новый столбец, в каждой строке которого вычисляется заданное выражение.

**Пример**

Вывести всю информацию о книгах, а также для каждой позиции посчитать ее стоимость (произведение цены на количество). Вычисляемому столбцу дать имя **total** .

Запрос:

SELECT title, author, price, amount,

price \* amount AS total

FROM book;

Результат:

+-----------------------+------------------+--------+--------+---------+

| title | author | price | amount | total |

+-----------------------+------------------+--------+--------+---------+

| Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 | 2012.97 |

| Белая гвардия | Булгаков М.А. | 540.50 | 5 | 2702.50 |

| Идиот | Достоевский Ф.М. | 460.00 | 10 | 4600.00 |

| Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 | 1598.02 |

| Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 | 9750.00 |

+-----------------------+------------------+--------+--------+---------+

## ****Задание****

Для упаковки каждой книги требуется один лист бумаги, цена которого 1 рубль 65 копеек. Посчитать стоимость упаковки для каждой книги (сколько денег потребуется, чтобы упаковать все экземпляры книги). В запросе вывести название книги, ее количество и стоимость упаковки, последний столбец назвать **pack**.

Результат:

+-----------------------+--------+-------+

| title | amount | pack |

+-----------------------+--------+-------+

| Мастер и Маргарита | 3 | 4.95 |

| Белая гвардия | 5 | 8.25 |

| Идиот | 10 | 16.50 |

| Братья Карамазовы | 2 | 3.30 |

| Стихотворения и поэмы | 15 | 24.75 |

+-----------------------+--------+-------+

Affected rows: 5

### Введите SQL запрос

SELECT title, amount, amount \* 1.65 AS pack FROM book;

## Выборка данных, вычисляемые столбцы, математические функции

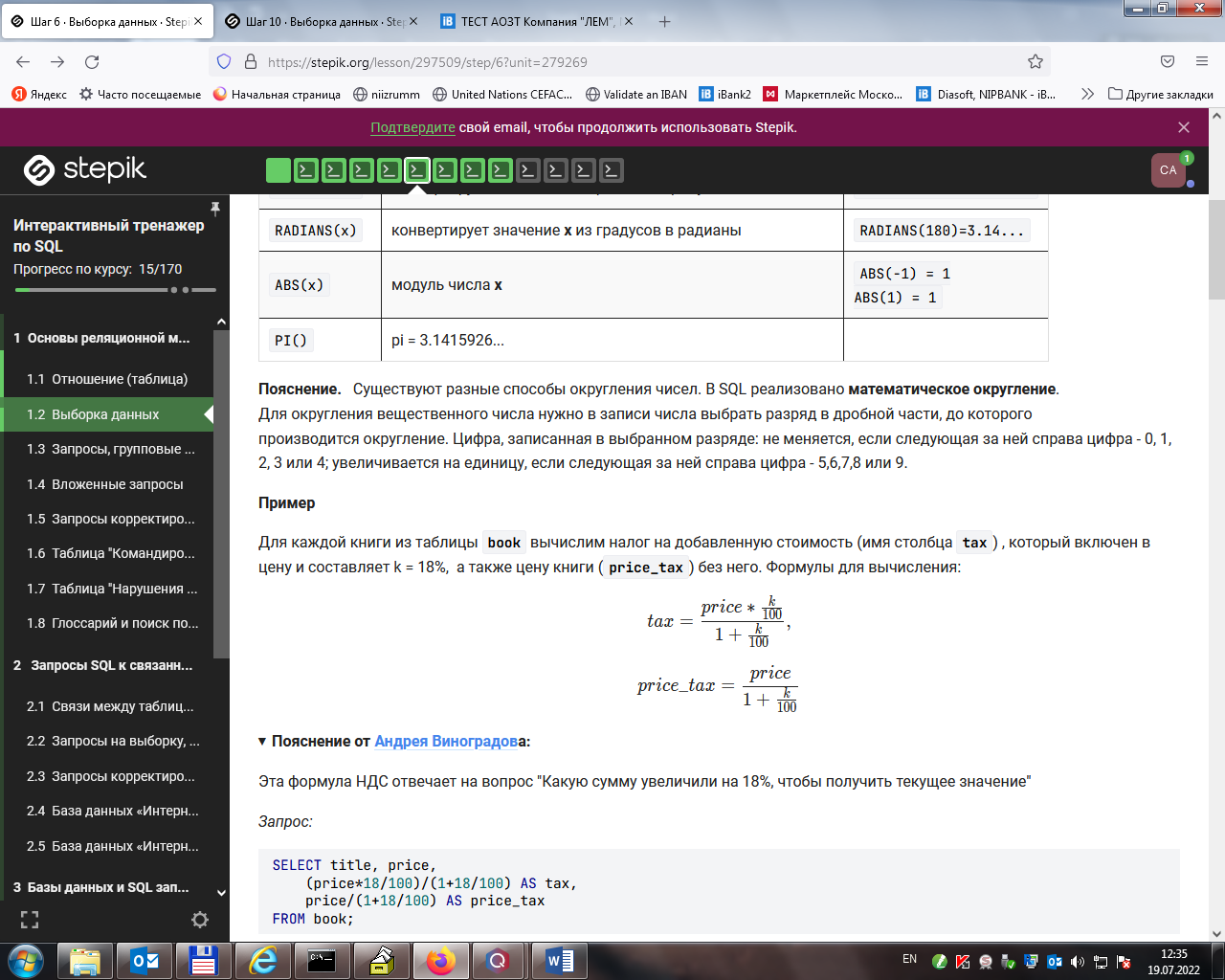
В SQL реализовано множество  [математических функций](https://docs.microsoft.com/ru-ru/sql/t-sql/functions/mathematical-functions-transact-sql?view=sql-server-ver15) для работы с числовыми данными. В таблице приведены некоторые из них.

| **Функция** | **Описание** | **Пример** |
| --- | --- | --- |
| CEILING(x) | возвращает наименьшее целое число, большее или равное **x** (округляет до целого числа в большую сторону) | CEILING(4.2)=5 CEILING(-5.8)=-5 |
| ROUND(x, k) | округляет значение **x** до **k** знаков после запятой, если **k** не указано – **x** округляется до целого | ROUND(4.361)=4 ROUND(5.86592,1)=5.9 |
| FLOOR(x) | возвращает наибольшее целое число, меньшее или равное **x** (округляет до  целого числа в меньшую сторону) | FLOOR(4.2)=4 FLOOR(-5.8)=-6 |
| POWER(x, y) | возведение **x** в степень **y** | POWER(3,4)=81.0 |
| SQRT(x) | квадратный корень из **x** | SQRT(4)=2.0 SQRT(2)=1.41... |
| DEGREES(x) | конвертирует значение **x** из радиан в градусы | DEGREES(3) = 171.8... |
| RADIANS(x) | конвертирует значение **x** из градусов в радианы | RADIANS(180)=3.14... |
| ABS(x) | модуль числа **x** | ABS(-1) = 1 ABS(1) = 1 |
| PI() | pi = 3.1415926... |  |

**Пояснение.**   Существуют разные способы округления чисел. В SQL реализовано **математическое округление**. Для округления вещественного числа нужно в записи числа выбрать разряд в дробной части, до которого производится округление. Цифра, записанная в выбранном разряде: не меняется, если следующая за ней справа цифра - 0, 1, 2, 3 или 4; увеличивается на единицу, если следующая за ней справа цифра - 5,6,7,8 или 9.

**Пример**

Для каждой книги из таблицы **book** вычислим налог на добавленную стоимость (имя столбца **tax**) , который включен в цену и составляет k = 18%,  а также цену книги (**price\_tax**) без него. Формулы для вычисления:



**Пояснение от**[**Андрея Виноградов**](https://stepik.org/users/137112290)**а:**

Эта формула НДС отвечает на вопрос "Какую сумму увеличили на 18%, чтобы получить текущее значение"

Запрос:

SELECT title, price,

(price\*18/100)/(1+18/100) AS tax,

price/(1+18/100) AS price\_tax

FROM book;

Результат:

+-----------------------+--------+----------------+------------+

| title | price | tax | price\_tax |

+-----------------------+--------+----------------+------------+

| Мастер и Маргарита | 670.99 | 102.3544067797 | 568.635593 |

| Белая гвардия | 540.50 | 82.4491525424 | 458.050847 |

| Идиот | 460.00 | 70.1694915254 | 389.830508 |

| Братья Карамазовы | 799.01 | 121.8828813559 | 677.127119 |

| Стихотворения и поэмы | 650.00 | 99.1525423729 | 550.847458 |

+-----------------------+--------+----------------+------------+

Сумма налога и цена книги без налога – это деньги, поэтому количество знаков после запятой у этих чисел должно быть 2. Следовательно необходимо округлить полученные значения.

Запрос:

SELECT title,

price,

ROUND((price\*18/100)/(1+18/100),2) AS tax,

ROUND(price/(1+18/100),2) AS price\_tax

FROM book;

Результат:

+-----------------------+--------+--------+-----------+

| title | price | tax | price\_tax |

+-----------------------+--------+--------+-----------+

| Мастер и Маргарита | 670.99 | 102.35 | 568.64 |

| Белая гвардия | 540.50 | 82.45 | 458.05 |

| Идиот | 460.00 | 70.17 | 389.83 |

| Братья Карамазовы | 799.01 | 121.88 | 677.13 |

| Стихотворения и поэмы | 650.00 | 99.15 | 550.85 |

+-----------------------+--------+--------+-----------+

## ****Задание****

В конце года цену всех книг на складе пересчитывают – снижают ее на 30%. Написать SQL запрос, который из таблицы **book** выбирает названия, авторов, количества и вычисляет новые цены книг. Столбец с новой ценой назвать **new\_price**, цену округлить до 2-х знаков после запятой.

Результат:

+-----------------------+------------------+--------+-----------+

| title | author | amount | new\_price |

+-----------------------+------------------+--------+-----------+

| Мастер и Маргарита | Булгаков М.А. | 3 | 469.69 |

| Белая гвардия | Булгаков М.А. | 5 | 378.35 |

| Идиот | Достоевский Ф.М. | 10 | 322.00 |

| Братья Карамазовы | Достоевский Ф.М. | 2 | 559.31 |

| Стихотворения и поэмы | Есенин С.А. | 15 | 455.00 |

+-----------------------+------------------+--------+-----------+

**Структура и наполнение таблицы book:**

### Введите SQL запрос

SELECT title, author, amount,

ROUND(price \* 0.7, 2) AS new\_price

FROM book;

## Выборка данных, вычисляемые столбцы, логические функции

В SQL реализована возможность заносить в поле значение в зависимости от условия. Для этого используется функция IF:

IF(логическое\_выражение, выражение\_1, выражение\_2)

Функция вычисляет логическое\_выражение, если оно истина – в поле заносится значение выражения\_1, в противном случае –  значение выражения\_2. Все три параметра IF() являются обязательными.

Допускается использование вложенных функций, вместо выражения\_1 или выражения\_2 может стоять новая функция **IF**.

**Пример**

Для каждой книги из таблицы **book** установим скидку следующим образом: если количество книг меньше 4, то скидка будет составлять 50% от цены, в противном случае 30%.

Запрос:

SELECT title, amount, price,

IF(amount<4, price\*0.5, price\*0.7) AS sale

FROM book;

Результат:

+-----------------------+--------+--------+---------+

| title | amount | price | sale |

+-----------------------+--------+--------+---------+

| Мастер и Маргарита | 3 | 670.99 | 335.495 |

| Белая гвардия | 5 | 540.50 | 378.350 |

| Идиот | 10 | 460.00 | 322.000 |

| Братья Карамазовы | 2 | 799.01 | 399.505 |

| Стихотворения и поэмы | 15 | 650.00 | 455.000 |

+-----------------------+--------+--------+---------+

Цена по скидке должна отображаться с двумя знаками после запятой, добавим в запрос округление:

SELECT title, amount, price,

ROUND(IF(amount<4, price\*0.5, price\*0.7),2) AS sale

FROM book;

Результат:

+-----------------------+--------+--------+--------+

| title | amount | price | sale |

+-----------------------+--------+--------+--------+

| Мастер и Маргарита | 3 | 670.99 | 335.50 |

| Белая гвардия | 5 | 540.50 | 378.35 |

| Идиот | 10 | 460.00 | 322.00 |

| Братья Карамазовы | 2 | 799.01 | 399.51 |

| Стихотворения и поэмы | 15 | 650.00 | 455.00 |

+-----------------------+--------+--------+--------+

**Пример**

Усложним вычисление скидки в зависимости от количества книг. Если количество книг меньше 4 – то скидка 50%, меньше 11 – 30%, в остальных случаях – 10%. И еще укажем какая именно скидка на каждую книгу.

Запрос:

SELECT title, amount, price,

ROUND(IF(amount < 4, price \* 0.5, IF(amount < 11, price \* 0.7, price \* 0.9)), 2) AS sale,

IF(amount < 4, 'скидка 50%', IF(amount < 11, 'скидка 30%', 'скидка 10%')) AS Ваша\_скидка

FROM book;

Результат:

+-----------------------+--------+--------+--------+-------------+

| title | amount | price | sale | Ваша\_скидка |

+-----------------------+--------+--------+--------+-------------+

| Мастер и Маргарита | 3 | 670.99 | 335.50 | скидка 50% |

| Белая гвардия | 5 | 540.50 | 378.35 | скидка 30% |

| Идиот | 10 | 460.00 | 322.00 | скидка 30% |

| Братья Карамазовы | 2 | 799.01 | 399.51 | скидка 50% |

| Стихотворения и поэмы | 15 | 650.00 | 585.00 | скидка 10% |

+-----------------------+--------+--------+--------+-------------+

## ****Задание****

При анализе продаж книг выяснилось, что наибольшей популярностью пользуются книги Михаила Булгакова, на втором месте книги Сергея Есенина. Исходя из этого решили поднять цену книг Булгакова на 10%, а цену книг Есенина - на 5%. Написать запрос, куда включить автора, название книги и новую цену, последний столбец назвать **new\_price**. Значение округлить до двух знаков после запятой.

**Пояснение**

* фамилию автора задавать с инициалами (как занесено в таблице), заключая в одинарные или двойные кавычки;
* для сравнения на равенство использовать знак =, например **author = "Булгаков М.А."**.

Результат:

+------------------+-----------------------+-----------+

| author | title | new\_price |

+------------------+-----------------------+-----------+

| Булгаков М.А. | Мастер и Маргарита | 738.09 |

| Булгаков М.А. | Белая гвардия | 594.55 |

| Достоевский Ф.М. | Идиот | 460.00 |

| Достоевский Ф.М. | Братья Карамазовы | 799.01 |

| Есенин С.А. | Стихотворения и поэмы | 682.50 |

+------------------+-----------------------+-----------+

**Структура и наполнение таблицы book:**

### Введите SQL запрос

SELECT author, title,

ROUND(IF(author = 'Булгаков М.А.', price \* 1.1, IF(author = 'Есенин С.А.', price \* 1.05, price)), 2) AS new\_price

FROM book;

## Выборка данных по условию

С помощью запросов можно включать в итоговую выборку не все строки исходной таблицы, а только те, которые отвечают некоторому условию. Для этого после указания таблицы, откуда выбираются данные, задается ключевое слово WHERE и логическое выражение, от результата которого зависит будет ли включена строка в выборку или нет. Если условие – истина, то строка(запись)  включается в выборку, если ложь – нет.

Логическое выражение может включать **операторы сравнения**(равно «**=**», не равно «**<>**», больше «**>**», меньше «**<**», больше или равно«**>=**», меньше или равно «**<=**») и выражения, допустимые в SQL.

**Пример**

Вывести название и цену тех книг, цены которых меньше 600 рублей.

Запрос:

SELECT title, price

FROM book

WHERE price < 600;

Результат:

+---------------+--------+

| title | price |

+---------------+--------+

| Белая гвардия | 540.50 |

| Идиот | 460.00 |

+---------------+--------+

**Пример**

Вывести название, автора  и стоимость (цена умножить на количество) тех книг, стоимость которых больше 4000 рублей

Запрос:

SELECT title, author, price \* amount AS total

FROM book

WHERE price \* amount > 4000;

Результат:

+-----------------------+------------------+---------+

| title | author | total |

+-----------------------+------------------+---------+

| Идиот | Достоевский Ф.М. | 4600.00 |

| Стихотворения и поэмы | Есенин С.А. | 9750.00 |

+-----------------------+------------------+---------+

**Пояснение**

В логическом выражении после **WHERE** нельзя использовать названия столбцов, присвоенные им с помощью **AS**,  так как при выполнении запроса сначала вычисляется логическое выражение для каждой строки исходной таблицы, выбираются строки, для которых оно истинно. А только после этого формируется "шапка запроса" – столбцы, включаемые в запрос.

## Задание

Вывести автора, название  и цены тех книг, количество которых меньше 10.

Результат:

+------------------+--------------------+--------+

| author | title | price |

+------------------+--------------------+--------+

| Булгаков М.А. | Мастер и Маргарита | 670.99 |

| Булгаков М.А. | Белая гвардия | 540.50 |

| Достоевский Ф.М. | Братья Карамазовы | 799.01 |

+------------------+--------------------+--------+

**Структура и наполнение таблицы book:**

### Введите SQL запрос

SELECT author, title, price

FROM book

WHERE amount < 10;

**Выборка данных, логические операции**

Логическое выражение после ключевого слова WHERE кроме операторов сравнения  и выражений может включать  **логические операции** (И «**and**», ИЛИ «**or**», НЕ «**not**») и круглые скобки, изменяющие приоритеты выполнения операций.

Приоритеты операций:

1. круглые скобки
2. умножение  (\*),  деление (/)
3. сложение  (+), вычитание (-)
4. операторы сравнения (=, >, <, >=, <=, <>)
5. NOT
6. AND
7. OR

**Пример**

Вывести название, автора и цену тех книг, которые написал Булгаков, ценой больше 600 рублей

*Запрос:*

SELECT title, author, price

FROM book

WHERE price > 600 AND author = 'Булгаков М.А.';

*Результат:*

+--------------------+---------------+--------+

| title | author | price |

+--------------------+---------------+--------+

| Мастер и Маргарита | Булгаков М.А. | 670.99 |

+--------------------+---------------+--------+

**Пример**

Вывести название, цену  тех книг, которые написал Булгаков или Есенин, ценой больше 600 рублей

*Запрос:*

SELECT title, author, price

FROM book

WHERE (author = 'Булгаков М.А.' OR author = 'Есенин С.А.') AND price > 600;

*Результат:*

+-----------------------+---------------+--------+

| title | author | price |

+-----------------------+---------------+--------+

| Мастер и Маргарита | Булгаков М.А. | 670.99 |

| Стихотворения и поэмы | Есенин С.А. | 650.00 |

+-----------------------+---------------+--------+

**Пояснение.**

В данном запросе обязательно нужно поставить скобки, так как без них сначала вычисляется  author = 'Есенин С.А.' and price > 600, а потом уже выражение через or. Без скобок были бы отобраны все книги Булгакова и те книги Есенина, цена которых больше 600.

*Запрос:*

SELECT title, author, price

FROM book

WHERE author = 'Булгаков М.А.' OR author = 'Есенин С.А.' AND price > 600;

*Результат (сравните с предыдущим):*

+-----------------------+---------------+--------+

| title | author | price |

+-----------------------+---------------+--------+

| Мастер и Маргарита | Булгаков М.А. | 670.99 |

| Белая гвардия | Булгаков М.А. | 540.50 |

| Стихотворения и поэмы | Есенин С.А. | 650.00 |

+-----------------------+---------------+--------+

**Задание**

Вывести название, автора,  цену  и количество всех книг, цена которых меньше 500 или больше 600, а стоимость всех экземпляров этих книг больше или равна 5000.

*Результат:*

+-----------------------+-------------+--------+--------+

| title | author | price | amount |

+-----------------------+-------------+--------+--------+

| Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

+-----------------------+-------------+--------+--------+

**Структура и наполнение таблицы book:**

**Введите SQL запрос**

SELECT title, author, price, amount

FROM book

WHERE(price < 500 OR price > 600) AND price \* amount >= 5000;

## Выборка данных, операторы BETWEEN, IN

Логическое выражение после ключевого слова WHERE может включать операторы  **BETWEEN** и **IN**. Приоритет  у этих операторов такой же как у операторов сравнения, то есть они выполняются раньше, чем **NOT**, **AND**, **OR**.

Оператор BETWEEN позволяет отобрать данные, относящиеся к некоторому интервалу, включая его границы.

**Пример**

Выбрать названия и количества тех книг, количество которых от 5 до 14 включительно.

Запрос:

SELECT title, amount

FROM book

WHERE amount BETWEEN 5 AND 14;

Результат:

+---------------+--------+

| title | amount |

+---------------+--------+

| Белая гвардия | 5 |

| Идиот | 10 |

+---------------+--------+

Этот запрос можно реализовать по-другому, результат будет точно такой же.

SELECT title, amount

FROM book

WHERE amount >= 5 AND amount <=14;

Оператор  IN  позволяет выбрать данные, соответствующие значениям из списка.

**Пример**

Выбрать названия и цены книг, написанных Булгаковым или Достоевским.

Запрос:

SELECT title, price

FROM book

WHERE author IN ('Булгаков М.А.', 'Достоевский Ф.М.');

Результат:

+--------------------+--------+

| title | price |

+--------------------+--------+

| Мастер и Маргарита | 670.99 |

| Белая гвардия | 540.50 |

| Идиот | 460.00 |

| Братья Карамазовы | 799.01 |

+--------------------+--------+

Этот запрос можно реализовать по-другому, результат будет точно такой же.

SELECT title, price

FROM book

WHERE author = 'Булгаков М.А.' OR author = 'Достоевский Ф.М.';

## Задание

Вывести название и авторов тех книг, цены которых принадлежат интервалу от 540.50 до 800 (включая границы),  а количество или 2, или 3, или 5, или 7 .

Результат:

+--------------------+------------------+

| title | author |

+--------------------+------------------+

| Мастер и Маргарита | Булгаков М.А. |

| Белая гвардия | Булгаков М.А. |

| Братья Карамазовы | Достоевский Ф.М. |

+--------------------+------------------+

**Структура и наполнение таблицы book:**

### Введите SQL запрос

SELECT title, author

FROM book

WHERE price BETWEEN 540.50 AND 800 AND amount IN (2, 3, 5, 7);

**Выборка данных с сортировкой**

При выборке можно указывать столбец или несколько столбцов, по которым необходимо отсортировать отобранные строки. Для этого используются ключевые слова ORDER BY, после которых задаются имена столбцов. При этом строки сортируются по первому столбцу, если указан второй столбец, сортировка осуществляется только для тех строк, у которых значения первого столбца одинаковы. По умолчанию ORDER BY выполняет сортировку по возрастанию. Чтобы управлять направлением сортировки вручную, после имени столбца указывается ключевое слово ASC (по возрастанию) или DESC (по убыванию).

Столбцы после ключевого слова ORDER BY можно задавать:

* названием столбца;
* номером столбца;
* именем столбца (указанным после AS).

**Пример**

Вывести название, автора и цены книг. Информацию  отсортировать по названиям книг в алфавитном порядке.

*Запрос:*

SELECT title, author, price

FROM book

ORDER BY title;

*Результат:*

+-----------------------+------------------+--------+

| title | author | price |

+-----------------------+------------------+--------+

| Белая гвардия | Булгаков М.А. | 540.50 |

| Братья Карамазовы | Достоевский Ф.М. | 799.01 |

| Идиот | Достоевский Ф.М. | 460.00 |

| Мастер и Маргарита | Булгаков М.А. | 670.99 |

| Стихотворения и поэмы | Есенин С.А. | 650.00 |

+-----------------------+------------------+--------+

Аналогичный результат получится при использовании запроса:

SELECT title, author, price

FROM book

ORDER BY 1;

**Пример**

Вывести автора, название и количество книг, в отсортированном в алфавитном порядке по автору и по убыванию количества, для тех книг, цены которых меньше 750 рублей.

*Запрос:*

SELECT author, title, amount AS Количество

FROM book

WHERE price < 750

ORDER BY author, amount DESC;

*Результат:*

+------------------+-----------------------+------------+

| author | title | Количество |

+------------------+-----------------------+------------+

| Булгаков М.А. | Белая гвардия | 5 |

| Булгаков М.А. | Мастер и Маргарита | 3 |

| Достоевский Ф.М. | Идиот | 10 |

| Есенин С.А. | Стихотворения и поэмы | 15 |

+------------------+-----------------------+------------+

Можно использовать другие варианты записи запроса:

SELECT author, title, amount AS Количество

FROM book

WHERE price < 750

ORDER BY author, Количество DESC;

SELECT author, title, amount AS Количество

FROM book

WHERE price < 750

ORDER BY 1, 3 DESC;

**Важно!** Если названия столбцов заключены в кавычки, то при использовании их в сортировке, необходимо записывать их БЕЗ КАВЫЧЕК.

**Задание**

Вывести  автора и название  книг, количество которых принадлежит интервалу от 2 до 14 (включая границы). Информацию  отсортировать сначала по авторам (в обратном алфавитном порядке), а затем по названиям книг (по алфавиту).

*Результат:*

+------------------+--------------------+

| author | title |

+------------------+--------------------+

| Достоевский Ф.М. | Братья Карамазовы |

| Достоевский Ф.М. | Идиот |

| Булгаков М.А. | Белая гвардия |

| Булгаков М.А. | Мастер и Маргарита |

+------------------+--------------------+

**Структура и наполнение таблицы book:**

**Введите SQL запрос**

SELECT author, title

FROM book

WHERE amount BETWEEN 2 AND 14

ORDER BY author DESC, title;

## Выборка данных, оператор LIKE

Оператор LIKE используется для сравнения строк. В отличие от операторов отношения равно (**=**) и не равно (**<>**), LIKE позволяет сравнивать строки не на полное совпадение (не совпадение), а в соответствии с шаблоном. Шаблон может включать **обычные символы** и **символы-шаблоны**. При сравнении с шаблоном, его обычные символы должны в точности совпадать с символами, указанными в строке. Символы-шаблоны могут совпадать с произвольными элементами символьной строки.

| **Символ-шаблон** | **Описание** | **Пример** |
| --- | --- | --- |
| **%** | Любая строка, содержащая ноль или более символов | SELECT \* FROM book WHERE author LIKE '%М.%'  выполняет поиск и выдает все книги, инициалы авторов которых содержат «М.» |
| **\_** (подчеркивание) | Любой одиночный символ | SELECT \* FROM book WHERE title LIKE 'Поэм\_'  выполняет поиск и выдает все книги, названия которых либо «Поэма», либо «Поэмы» и пр. |

**Пример 1**

Вывести названия книг, начинающихся с буквы «Б».

Запрос:

SELECT title

FROM book

WHERE title LIKE 'Б%';

/\* эквивалентное условие

title LIKE 'б%'

\*/

Результат:

+-------------------+

| title |

+-------------------+

| Белая гвардия |

| Братья Карамазовы |

+-------------------+

**Пояснение** Строчные и прописные буквы в строках эквивалентны.

**Пример 2**

Вывести название книг, состоящих ровно из 5 букв.

Запрос:

SELECT title FROM book

WHERE title LIKE "\_\_\_\_\_"

Результат:

+-------+

| title |

+-------+

| Идиот |

| Поэмы |

+-------+

**Пояснение** Для обозначения одного любого символа используется  "\_", следовательно для обозначения 5 символов используется 5 подряд символов подчеркивания.

**Пример 3**

Вывести книги, название которых длиннее 5 символов:

Запрос:

SELECT title FROM book

WHERE title LIKE "\_\_\_\_\_\_%";

/\* эквивалентные условия

title LIKE "%\_\_\_\_\_\_"

title LIKE "%\_\_\_\_\_\_%"

\*/

Результат:

+-----------------------+

| title |

+-----------------------+

| Мастер и Маргарита |

| Белая гвардия |

| Братья Карамазовы |

| Стихотворения и поэмы |

| Дети полуночи |

| Лирика |

| Капитанская дочка |

+-----------------------+

**Пояснение**

Для того чтобы вывести названия, состоящие из любого количества символов после**LIKE** можно использовать шаблон **"%"**, с помощью которого отбираются строки, состоящие из любого количества символов, в том числе и "пустые", поскольку % заменяет любое количество символов, в том числе и нулевое.

Чтобы указать, что в названии должен быть хотя бы один символ, можно использовать один из эквивалентных шаблонов:

* **"\_%"** - сначала идет символ, а за ним любое количество символов;
* **"%\_"** - сначала идет любое количество символов, а затем обязательный символ;
* **"%\_%"** - сначала идет любое количество символов, потом обязательный символ, а за ним любое количество символов.

**Пример 4**

Вывести названия книг, которые содержат букву **"и"** как отдельное слово, если считать, что слова в названии отделяются друг от друга пробелами и не содержат знаков препинания.

 Запрос:

SELECT title FROM book

WHERE title LIKE "\_% и \_%" /\*отбирает слово И внутри названия \*/

OR title LIKE "и \_%" /\*отбирает слово И в начале названия \*/

OR title LIKE "\_% и" /\*отбирает слово И в конце названия \*/

OR title LIKE "и" /\* отбирает название, состоящее из одного слова И \*/

Результат:

+-----------------------+

| title |

+-----------------------+

| Мастер и Маргарита |

| Стихотворения и поэмы |

+-----------------------+

**Пояснение**

Слово "и" может располагаться в названии в следующих позициях (при условии, что слова отделяются друг от друга пробелами):

* в середине -  **"\_% и \_%"** - сначала идет любое количество символов (один обязательный), потом обязательный пробел, а за ним "и", снова обязательный пробел, и наконец любое количество символов (один обязательный);
* в начале - **"и \_%"** - сначала идет "и", обязательный пробел и любое количество символов (один обязательный);
* в конце - **"\_% и"** - сначала идет любое количество символов, затем обязательный пробел и буква "и":
* одно слово в названии - **"и"**.

Вместо "**\_%"** можно использовать эквивалентные шаблоны **"%\_"** и **"%\_%"** .

В качестве обязательного символа ( "\_"), может быть и пробел, но, к сожалению, шаблоны для **LIKE** не позволяют исключить какой-то символ. Это можно сделать только с помощью регулярных выражений (будут рассмотрены в уроке 3.5)

**Пример 5**

Вывести названия книг, которые состоят ровно из одного слова, если считать, что слова в названии отделяются друг от друга пробелами .

 Запрос:

SELECT title FROM book

WHERE title NOT LIKE "% %";

Результат:

+--------+

| title |

+--------+

| Идиот |

| Лирика |

| Поэмы |

+--------+

**Пояснение** Отсутствие пробела в названии означает, что оно состоит из одного слова. Чтобы это проверить используется оператор , который в данном случае отберет все названия, в которых нет пробелов.

## Задание

Вывести название и автора тех книг, название которых состоит из двух и более слов, а инициалы автора содержат букву «С». Считать, что в названии слова отделяются друг от друга пробелами и не содержат знаков препинания, между фамилией автора и инициалами обязателен пробел, инициалы записываются без пробела в формате: буква, точка, буква, точка. Информацию отсортировать по названию книги в алфавитном порядке.

Результат:

+-----------------------+-------------+

| title | author |

+-----------------------+-------------+

| Капитанская дочка | Пушкин А.С. |

| Стихотворения и поэмы | Есенин С.А. |

+-----------------------+-------------+

### Введите SQL запрос

SELECT title, author

FROM book

WHERE title LIKE '\_% %'

AND author LIKE '%С.%'

ORDER BY title;

## 1.3 Запросы, групповые операции

## Выбор уникальных элементов столбца

Чтобы отобрать уникальные элементы некоторого столбца используется ключевое слово DISTINCT, которое размещается сразу после SELECT.

**Пример**

Выбрать различных авторов, книги которых хранятся в таблице **book**.

Запрос:

SELECT DISTINCT author

FROM book;

Результат:

+------------------+

| author |

+------------------+

| Булгаков М.А. |

| Достоевский Ф.М. |

| Есенин С.А. |

+------------------+

Другой способ – использование оператора GROUP BY, который группирует данные при выборке, имеющие одинаковые значения в некотором столбце. Столбец, по которому осуществляется группировка, указывается после GROUP BY .

С помощью GROUP BY можно выбрать уникальные элементы столбца, по которому осуществляется группировка. Результат будет точно такой же как при использовании DISTINCT.

Запрос:

SELECT author

FROM book

GROUP BY author;

## Задание

Отобрать различные (уникальные) элементы столбца **amount** таблицы **book**.

**Результат**(**попробуйте решать задания без известного результата, если не получится - раскройте его**) **Структура и наполнение таблицы book**

### Введите SQL запрос

SELECT DISTINCT amount

FROM book;

## Выборка данных, групповые функции SUM и COUNT

При группировке над элементами столбца, входящими в группу можно выполнить различные действия, например, просуммировать их или найти количество элементов в группе.

Подробно рассмотрим, как осуществляется группировка данных по некоторому столбцу и вычисления над группой на следующем примере:

SELECT author, sum(amount), count(amount)

FROM book

GROUP BY author;

1. В таблице **book** определяются строки, в которых в столбце **author** одинаковые значения:



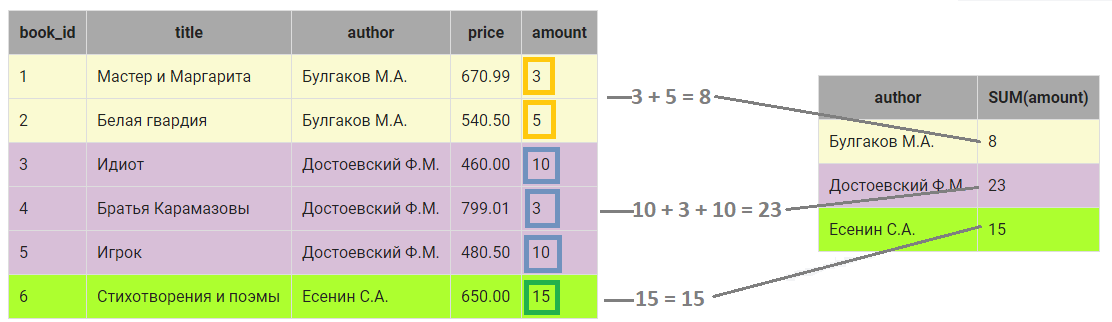
Получили 3 различные группы:

* **группа I** объединяет две записи, у которых в столбце **author** значение Булгаков М.А.;
* **группа II** объединяет три записи, у которых в столбце **author** значение Достоевский Ф.М.;
* **группа III** объединяет одну запись, у которой в столбце **author** значение Есенин С.А.

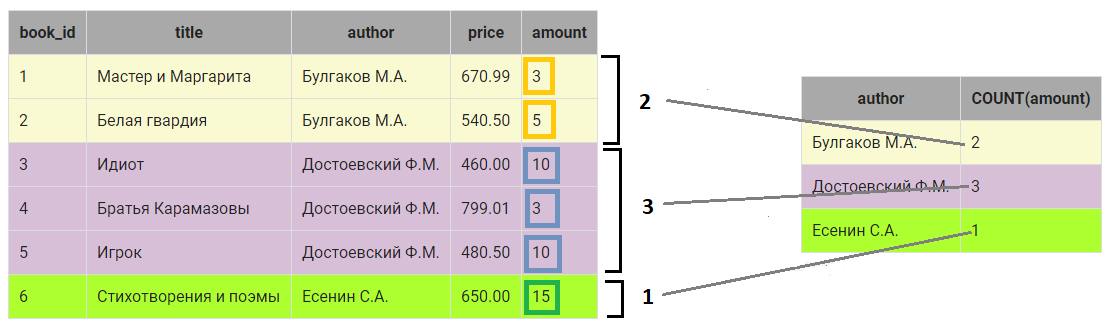
2. Вместо каждой группы в результирующий запрос включается  одна запись. Запись как минимум включает значение столбца, по которому осуществляется группировка (в нашем случае это **author**):



3. Дальше можно выполнить вычисления над элементами КАЖДОЙ группы в отдельности, например, посчитать общее количество экземпляров книг каждого автора. Для этого используется групповая функция SUM(), а в скобках указывается столбец, по которому нужно выполнить суммирование ( в нашем случае **amount**):



4. Также можно посчитать, сколько записей относится к группе. Для этого используется функция COUNT(), в скобках можно указать ЛЮБОЙ столбец из группы, если группа не содержит пустых значений (ниже приведен пример, в котором показано, как работает **COUNT()**, если в группе есть пустые значения):



**Пример**

Посчитать, сколько экземпляров книг каждого автора хранится на складе.

*Запрос:*

SELECT author, SUM(amount)

FROM book

GROUP BY author;

*Результат:*

+------------------+-------------+

| author | SUM(amount) |

+------------------+-------------+

| Булгаков М.А. | 8 |

| Достоевский Ф.М. | 23 |

| Есенин С.А. | 15 |

+------------------+-------------+

**Примечание**

Обратите внимание, что в качестве названия вычисляемого столбца в результирующей таблице используется выражение. Рекомендуется всем  вычисляемым столбцам давать имя.

**Пример**

Посчитать, сколько различных книг каждого автора хранится на складе.

Только для этого примера в таблицу **book** добавлена запись с пустыми значениями в столбцах **amount** и **price**:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 3 |

| 5 | Игрок | Достоевский Ф.М. | 480.50 | 10 |

| 6 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

| 7 | Черный человек | Есенин С.А. | Null | Null |

+---------+-----------------------+------------------+--------+--------+

*Запрос:*

/\* чтобы проверить запрос, добавьте в таблицу строку \*/

INSERT INTO book (title, author, price, amount) VALUES ('Черный человек','Есенин С.А.', Null, Null);

SELECT author, COUNT(author), COUNT(amount), COUNT(\*)

FROM book

GROUP BY author;

*Результат:*

+------------------+---------------+---------------+----------+

| author | COUNT(author) | COUNT(amount) | COUNT(\*) |

+------------------+---------------+---------------+----------+

| Булгаков М.А. | 2 | 2 | 2 |

| Достоевский Ф.М. | 3 | 3 | 3 |

| Есенин С.А. | 2 | 1 | 2 |

+------------------+---------------+---------------+----------+

Из таблицы с результатами запроса видно, что функцию **COUNT()** можно применять к любому столбцу, в том числе можно использовать и **\***, если таблица не содержит пустых значений. Если же в столбцах есть значения **Null**, (для группы по автору Есенин в нашем примере), то

* **COUNT(\*)** —  подсчитывает  все записи, относящиеся к группе, в том числе и со значением **NULL**;
* **COUNT(имя\_столбца)** — возвращает количество записей конкретного столбца (только **NOT NULL**), относящихся к группе.

**ВАЖНО.**

1. Если столбец указан в SELECT  **БЕЗ**применения групповой функции, то он обязательно должен быть указан и вGROUP BY.Иначе получим ошибку.
2. Между названием функции и скобкой **НЕЛЬЗЯ СТАВИТЬ ПРОБЕЛ**. Это особенность платформы.

**Задание**

Посчитать, количество различных книг и количество экземпляров книг каждого автора , хранящихся на складе.  Столбцы назвать **Автор,** **Различных\_книг** и **Количество\_экземпляров** соответственно.

**Результат**

+------------------+----------------+------------------------+

| Автор | Различных\_книг | Количество\_экземпляров |

+------------------+----------------+------------------------+

| Булгаков М.А. | 2 | 8 |

| Достоевский Ф.М. | 3 | 23 |

| Есенин С.А. | 1 | 15 |

+------------------+----------------+------------------------+

**Пояснение**

Название столбцов может состоять из нескольких слов, тогда их нужно заключать в кавычки. Но если слова написать через подчеркивание, тогда получится , что название состоит из одного слова, и кавычки можно не ставить.

**Структура и наполнение таблицы book**

**Введите SQL запрос**

SELECT author AS 'Автор', COUNT(title) AS 'Различных\_книг', SUM(amount) AS 'Количество\_экземпляров'

FROM book

GROUP BY author;

## Выборка данных, групповые функции MIN, MAX и AVG

К групповым функциям SQL относятся: MIN(), MAX() и AVG(), которые вычисляют минимальное, максимальное и среднее значение элементов столбца, относящихся к группе.

**Пример**

Вывести минимальную цену книги каждого автора

Запрос:

SELECT author, MIN(price) AS min\_price

FROM book

GROUP BY author;

Результат:

+------------------+-----------+

| author | min\_price |

+------------------+-----------+

| Булгаков М.А. | 540.50 |

| Достоевский Ф.М. | 460.00 |

| Есенин С.А. | 650.00 |

+------------------+-----------+

## Задание

Вывести фамилию и инициалы автора, минимальную, максимальную и среднюю цену книг каждого автора . Вычисляемые столбцы назвать **Минимальная\_цена, Максимальная\_цена** и **Средняя\_цена** соответственно.

**Результат**

+------------------+------------------+-------------------+--------------+

| author | Минимальная\_цена | Максимальная\_цена | Средняя\_цена |

+------------------+------------------+-------------------+--------------+

| Булгаков М.А. | 540.50 | 670.99 | 605.745000 |

| Достоевский Ф.М. | 460.00 | 799.01 | 579.836667 |

| Есенин С.А. | 650.00 | 650.00 | 650.000000 |

+------------------+------------------+-------------------+--------------+

**Структура и наполнение таблицы book**

### Введите SQL запрос

SELECT author, MIN(price) AS Минимальная\_цена,

MAX(price) AS Максимальная\_цена, AVG(price) AS Средняя\_цена

FROM book

GROUP BY author;

## Выборка данных c вычислением, групповые функции

В качестве аргумента групповых функций  SQL может использоваться не только столбец, но и любое допустимое в SQL арифметическое выражение.

**Пример**

Вывести суммарную стоимость книг каждого автора.

Запрос:

SELECT author, SUM(price \* amount) AS Стоимость

FROM book

GROUP BY author;

Результат:

+------------------+-----------+

| author | Стоимость |

+------------------+-----------+

| Булгаков М.А. | 4715.47 |

| Достоевский Ф.М. | 11802.03 |

| Есенин С.А. | 9750.00 |

+------------------+-----------+

Групповые функции могут быть элементами выражений. Например, при вычислении средней стоимости книг каждого автора на предыдущем шаге получились значения с шестью знаками после запятой. А поскольку это деньги, значения нужно округлить до 2 знаков после запятой.

**Пример**

Найти среднюю цену книг каждого автора.

Запрос:

SELECT author, ROUND(AVG(price),2) AS Средняя\_цена

FROM book

GROUP BY author;

Результат:

+------------------+--------------+

| author | Средняя\_цена |

+------------------+--------------+

| Булгаков М.А. | 605.75 |

| Достоевский Ф.М. | 579.84 |

| Есенин С.А. | 650.00 |

+------------------+--------------+

## Задание

Для каждого автора вычислить суммарную стоимость книг **S** (имя столбца **Стоимость**), а также вычислить налог на добавленную стоимость  для полученных сумм (имя столбца **НДС** ) , который **включен в стоимость** и составляет k = 18%,  а также стоимость книг  (**Стоимость\_без\_НДС**) без него. Значения округлить до двух знаков после запятой. В запросе для расчета НДС(**tax**)  и Стоимости без НДС(**S\_without\_tax**) использовать следующие формулы:

tax=S∗k1001+k100,tax= {{S \*{ k \over 100}} \over {1+{k\over 100}}},tax=1+100k​S∗100k​​,

S\_without\_tax=S1+k100S\\_without\\_tax= {{S} \over {1+{k\over 100}}}S\_without\_tax=1+100k​S​

**Результат**

+------------------+-----------+---------+-------------------+

| author | Стоимость | НДС | Стоимость\_без\_НДС |

+------------------+-----------+---------+-------------------+

| Булгаков М.А. | 4715.47 | 719.31 | 3996.16 |

| Достоевский Ф.М. | 11802.03 | 1800.31 | 10001.72 |

| Есенин С.А. | 9750.00 | 1487.29 | 8262.71 |

+------------------+-----------+---------+-------------------+

**Пояснение**

Имена столбцов, присвоенные им с помощью AS, нельзя использовать в выражениях, используйте названия столбцов исходной таблицы.

**Структура и наполнение таблицы book**

### Введите SQL запрос

SELECT author, SUM(price \* amount) AS Стоимость,

ROUND(SUM(price \* amount \* 18/100)/(1 + 18/100),2) AS НДС,

ROUND(SUM(price \* amount)/(1 + 18/100),2) AS Стоимость\_без\_НДС

FROM book

GROUP BY author;

## Вычисления по таблице целиком

Групповые функции позволяют вычислять итоговые значения по всей таблице. Например, можно посчитать общее количество книг на складе, вычислить суммарную стоимость и пр. Для этого после ключевого слова SELECT указывается групповая функция для выражения или имени столбца, а ключевые слова GROUP BY опускаются.

**Пример**

Посчитать количество экземпляров книг на складе.

Запрос:

SELECT SUM(amount) AS Количество

FROM book;

Результат:

+------------+

| Количество |

+------------+

| 46 |

+------------+

Результатом таких запросов является единственная строка с вычисленными по таблице значениями.

**Пример**

Посчитать общее количество экземпляров книг на складе и их стоимость .

Запрос:

SELECT SUM(amount) AS Количество,

SUM(price \* amount) AS Стоимость

FROM book;

Результат:

+------------+-----------+

| Количество | Стоимость |

+------------+-----------+

| 46 | 26267.50 |

+------------+-----------+

## Задание

Вывести  цену самой дешевой книги, цену самой дорогой и среднюю цену уникальных книг на складе. Названия столбцов **Минимальная\_цена, Максимальная\_цена, Средняя\_цена** соответственно. Среднюю цену округлить до двух знаков после запятой.

**Пояснение**. В задании нужно посчитать среднюю цену уникальных книг на складе, а не среднюю цену всех экземпляров книг.

**Результат**

+------------------+-------------------+--------------+

| Минимальная\_цена | Максимальная\_цена | Средняя\_цена |

+------------------+-------------------+--------------+

| 460.00 | 799.01 | 600.17 |

+------------------+-------------------+--------------+

**Структура и наполнение таблицы book**

### Введите SQL запрос

SELECT MIN(price) AS Минимальная\_цена,

MAX(price) AS Максимальная\_цена,

ROUND(AVG(DISTINCT price), 2) AS Средняя\_цена

FROM book;

## Выборка данных по условию, групповые функции

В запросы с групповыми функциями можно включать условие отбора строк, которое в обычных запросах записывается после WHERE. В запросах с групповыми функциями вместо WHERE используется ключевое слово HAVING , которое размещается после оператора GROUP BY.

**Пример**

Найти минимальную и максимальную цену книг всех авторов, общая стоимость книг которых больше 5000.

Запрос:

SELECT author,

MIN(price) AS Минимальная\_цена,

MAX(price) AS Максимальная\_цена

FROM book

GROUP BY author

HAVING SUM(price \* amount) > 5000;

Результат:

+------------------+------------------+-------------------+

| author | Минимальная\_цена | Максимальная\_цена |

+------------------+------------------+-------------------+

| Достоевский Ф.М. | 460.00 | 799.01 |

| Есенин С.А. | 650.00 | 650.00 |

+------------------+------------------+-------------------+

Также в запросах с группировкой можно сортировать данные.

**Пример**

Найти минимальную и максимальную цену книг всех авторов, общая стоимость книг которых больше 5000. Результат вывести по убыванию минимальной цены.

Запрос:

SELECT author,

MIN(price) AS Минимальная\_цена,

MAX(price) AS Максимальная\_цена

FROM book

GROUP BY author

HAVING SUM(price \* amount) > 5000

ORDER BY Минимальная\_цена DESC;

Результат:

+------------------+------------------+-------------------+

| author | Минимальная\_цена | Максимальная\_цена |

+------------------+------------------+-------------------+

| Есенин С.А. | 650.00 | 650.00 |

| Достоевский Ф.М. | 460.00 | 799.01 |

+------------------+------------------+-------------------+

**Пояснение**

При указании столбца, по которому выполняется сортировка, если столбцу присвоено имя  с помощью AS, можно использовать это имя.

## Задание

Вычислить среднюю цену и суммарную стоимость тех книг, количество экземпляров которых принадлежит интервалу от 5 до 14, включительно. Столбцы назвать **Средняя\_цена** и **Стоимость**, значения округлить до 2-х знаков после запятой.

**Результат**

+--------------+-----------+

| Средняя\_цена | Стоимость |

+--------------+-----------+

| 493.67 | 12107.50 |

+--------------+-----------+

**Пояснение**

Если в запросе с групповыми функциями отсутствуетGROUP BY, то для отбора записей используется ключевое слово WHERE**.**

**Структура и наполнение таблицы book**

### Введите SQL запрос

SELECT ROUND(AVG(price), 2) AS Средняя\_цена,

ROUND(SUM(price \* amount), 2) AS Стоимость

FROM book

WHERE amount BETWEEN 5 AND 14;

## Выборка данных по условию, групповые функции, WHERE и HAVING

Для этого урока теоретическая часть подготовлена [*Alexandra Klinnikova*](https://stepik.org/users/2290685), спасибо большое!

WHERE и HAVING могут использоваться в одном запросе. При этом необходимо учитывать **порядок выполнения  SQL запроса на выборку на СЕРВЕРЕ**:

1. FROM
2. WHERE
3. GROUP BY
4. HAVING
5. SELECT
6. ORDER BY

Сначала определяется таблица, из которой выбираются данные (FROM), затем из этой таблицы отбираются записи в соответствии с условием  WHERE, выбранные данные агрегируются (GROUP BY),  из агрегированных записей выбираются те, которые удовлетворяют условию после HAVING. Потом формируются данные результирующей выборки, как это указано после SELECT ( вычисляются выражения, присваиваются имена и пр. ). Результирующая выборка сортируется, как указано после ORDER BY.

**Важно!** Порядок ВЫПОЛНЕНИЯ запросов - это не порядок ЗАПИСИ ключевых слов в запросе на выборку. Порядок записи (синтаксис запроса) остается таким же, как рассматривался ранее в курсе. Порядок ВЫПОЛНЕНИЯ  нужен для того, чтобы понять, почему, например, в **WHERE** нельзя использовать имена выражений из **SELECT**. Просто **SELECT** выполняется компилятором позже, чем **WHERE**, поэтому ему неизвестно, какое там выражение написано.

**Пример**

Вывести максимальную и минимальную цену книг каждого автора, кроме Есенина, количество экземпляров книг которого больше 10.

SELECT author,

MIN(price) AS Минимальная\_цена,

MAX(price) AS Максимальная\_цена

FROM book

WHERE author <> 'Есенин С.А.'

GROUP BY author

HAVING SUM(amount) > 10;

Результат:

+------------------+------------------+-------------------+

| author | Минимальная\_цена | Максимальная\_цена |

+------------------+------------------+-------------------+

| Достоевский Ф.М. | 460.00 | 799.01 |

+------------------+------------------+-------------------+

Другим способом решения примера является запрос:

SELECT author,

MIN(price) AS Минимальная\_цена,

MAX(price) AS Максимальная\_цена

FROM book

GROUP BY author

HAVING SUM(amount) > 10 AND author <> 'Есенин С.А.';

Не смотря на то что результат будет одинаковым, так делать **не рекомендуется**. «Потому что как написано - запрос сначала выбирает всех авторов, потом выводит данные, рассчитывая минимальное и максимальное значение цены для каждого, и только после всего убирает Есенина. Можно убрать Есенина в данном случае раньше и не использовать ресурсы базы для расчета его минимального и максимального значения, как это сделано в первом варианте. На небольшой базе быстродействия не ощутить, но если выполнять такое на продуктивной, то второй вариант значительно проигрывает...»[[Alexandra Klinnikova](https://stepik.org/users/2290685)].

## ****Задание****

Посчитать стоимость всех экземпляров каждого автора без учета книг «Идиот» и «Белая гвардия». В результат включить только тех авторов, у которых суммарная стоимость книг (без учета книг «Идиот» и «Белая гвардия») более 5000 руб. Вычисляемый столбец назвать **Стоимость**. Результат отсортировать по убыванию стоимости.

**Результат**

+------------------+-----------+

| author | Стоимость |

+------------------+-----------+

| Есенин С.А. | 9750.00 |

| Достоевский Ф.М. | 7202.03 |

+------------------+-----------+

**Структура и наполнение таблицы book**

### Введите SQL запрос

SELECT author,

SUM(price \* amount) AS Стоимость

FROM book

WHERE title <> 'Идиот' AND title <> 'Белая гвардия'

GROUP BY author

HAVING SUM(price \* amount) > 5000

ORDER BY Стоимость DESC;

## Задание

Придумайте один или несколько запросов к нашей таблице **book**, используя групповые функции. Проверьте, правильно ли они работают.

При желании можно формулировку запросов  разместить в комментариях.

Размещенные задания можно использовать для закрепления материала урока.

Оценивайте понравившиеся Вам запросы.

В последнем модуле создан отдельный урок, в котором мы разместим запросы, набравшие наибольшее количество лайков.

**Структура и наполнение таблицы book:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author** | **price** | **amount** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(50)** | **VARCHAR(30)** | **DECIMAL(8,2)** | **INT** |
| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |
| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |
| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |
| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 3 |
| 5 | Игрок | Достоевский Ф.М. | 480.50 | 10 |
| 6 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

### Введите SQL запрос

SELECT book\_id AS Порядковый\_номер, author AS Автор, ROUND(AVG(price)) AS Средняя\_цена

FROM book

GROUP BY book\_id, author

HAVING Порядковый\_номер > 2;

## 1.4 Вложенные запросы

## Содержание урока

SQL позволяет создавать вложенные запросы. Вложенный запрос (подзапрос, внутренний запрос) – это запрос внутри другого запроса SQL.

Вложенный запрос используется для выборки данных, которые будут использоваться в условии отбора записей основного запроса. Его применяют для:

* сравнения выражения с результатом вложенного запроса;
* определения того, включено ли выражение в результаты вложенного запроса;
* проверки того, выбирает ли запрос определенные строки.

Вложенный запрос имеет следующие компоненты:

* ключевое слово SELECT  после которого указываются имена столбцов или выражения (чаще всего список содержит один элемент) ;
* ключевое слово FROM и имя таблицы, из которой выбираются данные;
* необязательное предложение WHERE;
* необязательное предложение GROUP BY:
* необязательное предложение HAVING.

 Вложенные запросы  могут включаться в WHERE или HAVING так (в квадратных скобках указаны необязательные элементы, через | – один из элементов):

* [WHERE](https://stepik.org/lesson/297514/step/2?unit=279274) | [HAVING](https://stepik.org/lesson/297515/step/7?unit=279275) выражение оператор\_сравнения (вложенный запрос);
* [WHERE](https://stepik.org/lesson/297514/step/3?unit=279274) | [HAVING](https://stepik.org/lesson/297515/step/7?unit=279275) выражение, включающее вложенный запрос**;**
* [WHERE](https://stepik.org/lesson/297514/step/4?unit=279274) | [HAVING](https://stepik.org/lesson/297515/step/7?unit=279275) выражение [NOT] IN (вложенный запрос);
* [WHERE](https://stepik.org/lesson/297514/step/5?unit=279274) | [HAVING](https://stepik.org/lesson/297515/step/7?unit=279275) выражение  оператор\_сравнения  ANY | ALL (вложенный запрос).

Также вложенные запросы могут вставляться в основной запрос после ключевого слова [SELECT](https://stepik.org/lesson/297514/step/6?unit=279274).

## Вложенный запрос, возвращающий одно значение

Вложенный запрос, возвращающий одно значение, может использоваться в условии отбора записей WHERE как обычное значение совместно с операциями =, <>, >=, <=, >, <.

**Пример**

Вывести информацию о самых дешевых книгах, хранящихся на складе.

Для реализации этого запроса нам необходимо получить минимальную цену из столбца **price** таблицы **book**, а затем вывести информацию о тех книгах, цена которых  равна минимальной. Первая часть  – поиск  минимума – реализуется вложенным запросом.

Запрос:

SELECT title, author, price, amount

FROM book

WHERE price = (

SELECT MIN(price)

FROM book

);

Результат:

+-------+------------------+--------+--------+

| title | author | price | amount |

+-------+------------------+--------+--------+

| Идиот | Достоевский Ф.М. | 460.00 | 10 |

+-------+------------------+--------+--------+

Вложенный запрос определяет минимальную цену книг во всей таблице (это 460.00), а затем в основном запросе для каждой записи проверяется, равна ли цена минимальному значению, если равна, информация о книге включается в результирующую таблицу запроса.

**Рекомендация.** При использовании вложенного запроса рекомендуется сначала проверить, правильно ли он работает (занести текст запроса в окно кода и нажать черную кнопку **Запустить**), если выдается верный результат – использовать код в качестве вложенного запроса.

## Задание

Вывести информацию (автора, название и цену) о  книгах, цены которых меньше или равны средней цене книг на складе. Информацию вывести в отсортированном по убыванию цены виде. Среднее вычислить как среднее по цене книги.

**Результат**

+------------------+---------------+--------+

| author | title | price |

+------------------+---------------+--------+

| Булгаков М.А. | Белая гвардия | 540.50 |

| Достоевский Ф.М. | Игрок | 480.50 |

| Достоевский Ф.М. | Идиот | 460.00 |

+------------------+---------------+--------+

**Результат** **Структура и наполнение таблицы book**

### Введите SQL запрос

SELECT author, title, price

FROM book

WHERE price <= (

SELECT AVG(price)

FROM book)

ORDER BY price DESC

;

## Использование вложенного запроса в выражении

Вложенный запрос, возвращающий одно значение, может использоваться в выражениях как обычный операнд, например, к нему можно что-то прибавить, вычесть и пр.

**Пример**

Вывести информацию о книгах, количество экземпляров которых отличается от среднего количества экземпляров книг на складе более чем на 3. То есть нужно вывести и те книги, количество экземпляров которых меньше среднего на 3, или больше среднего на 3.

Запрос:

SELECT title, author, amount

FROM book

WHERE ABS(amount - (SELECT AVG(amount) FROM book)) >3;

Результат:

+-----------------------+------------------+--------+

| title | author | amount |

+-----------------------+------------------+--------+

| Мастер и Маргарита | Булгаков М.А. | 3 |

| Братья Карамазовы | Достоевский Ф.М. | 3 |

| Стихотворения и поэмы | Есенин С.А. | 15 |

+-----------------------+------------------+--------+

## Задание

Вывести информацию (автора, название и цену) о тех книгах, цены которых превышают минимальную цену книги на складе не более чем на 150 рублей в отсортированном по возрастанию цены виде.

**Результат**

+------------------+----------------+--------+

| author | title | price |

+------------------+----------------+--------+

| Достоевский Ф.М. | Идиот | 460.00 |

| Достоевский Ф.М. | Игрок | 480.50 |

| Булгаков М.А. | Белая гвардия | 540.50 |

| Пушкин А.С. | Евгений Онегин | 610.00 |

+------------------+----------------+--------+

**Структура и наполнение таблицы book**

### Введите SQL запрос

SELECT author, title, price

FROM book

WHERE (price - (SELECT MIN(price) FROM book)) <= 150

ORDER BY price;

**Вложенный запрос, оператор IN**

Вложенный запрос может возвращать несколько значений одного столбца.  Тогда его можно использовать в разделе WHERE совместно с оператором IN.

WHERE имя\_столбца IN (вложенный запрос, возвращающий один столбец)

Оператор IN определяет, совпадает ли значение столбца с одним из значений, содержащихся во вложенном запросе. При этом логическое выражение после WHERE получает значение истина. Оператор NOT IN выполняет обратное действие – выражение истинно, если значение столбца не содержится во вложенном запросе.

**Пример**

Вывести информацию о книгах тех авторов, общее количество экземпляров книг которых не менее 12.

*Запрос:*

SELECT title, author, amount, price

FROM book

WHERE author IN (

SELECT author

FROM book

GROUP BY author

HAVING SUM(amount) >= 12

);

*Результат:*

+-----------------------+------------------+--------+--------+

| title | author | amount | price |

+-----------------------+------------------+--------+--------+

| Идиот | Достоевский Ф.М. | 10 | 460.00 |

| Братья Карамазовы | Достоевский Ф.М. | 3 | 799.01 |

| Игрок | Достоевский Ф.М. | 10 | 480.50 |

| Стихотворения и поэмы | Есенин С.А. | 15 | 650.00 |

+-----------------------+------------------+--------+--------+

Вложенный запрос отбирает двух авторов (Достоевского и Есенина). А в основном запросе для каждой записи таблицы **book**  проверяется, входит ли автор книги в отобранный список, если входит - информация о книге включается в запрос.

**Задание**

Вывести информацию (автора, книгу и количество) о тех книгах, количество экземпляров которых в таблице **book** не дублируется.

**Пояснение к заданию**

В таблице **book** в столбце **amount** хранится количество экземпляров каждой книги:

+-----------------------+--------+

| title | amount |

+-----------------------+--------+

| Мастер и Маргарита | 3 |

| Белая гвардия | 5 |

| Идиот | 10 |

| Братья Карамазовы | 3 |

| Игрок | 10 |

| Стихотворения и поэмы | 15 |

+-----------------------+--------+

В соответствии с этой таблицей:

* количество экземпляров книг "Мастер и Маргарита" и "Братья Карамазовы" одинаково и равно 3 (так как число 3 встречается в таблице два раза, книги с таким количеством не подходят под условие);
* количество экземпляров книг "Идиот" и "Игрок" тоже одинаково и равно 10 (не подходят под условие);
* количество экземпляров книги "Белая гвардия равно" 5, при этом в таблице нет других книг, количество экземпляров которых равно 5, следовательно, эта книга подходит под условие задачи (так как количество экземпляров 5 в таблице не дублируется);
* количество экземпляров книги "Стихотворение и поэмы"  - 15, в таблице нет других книг, количество экземпляров которых тоже равно 15, следовательно, и эта книга подходит под условие.

Таким образом, необходимо вывести те строки таблицы, у которых числа в столбце **amount** не повторяются.

**Результат**

+---------------+-----------------------+--------+

| author | title | amount |

+---------------+-----------------------+--------+

| Булгаков М.А. | Белая гвардия | 5 |

| Есенин С.А. | Стихотворения и поэмы | 15 |

+---------------+-----------------------+--------+

**Пояснение к решению**

Во вложенном запросе отберите те значения столбца **amount**, количество которых, вычисленное с помощью функции **count()**, равно 1.

**Структура и наполнение таблицы book**

**Введите SQL запрос**

SELECT author, title, amount

FROM book

WHERE amount IN (

SELECT amount

FROM book

GROUP BY amount

HAVING COUNT(amount)=1);

**Вложенный запрос, операторы ANY и ALL**

 Вложенный запрос, возвращающий несколько значений одного столбца, можно использовать для отбора записей с помощью операторов ANY и ALL совместно с операциями отношения (=, <>, <=, >=, <, >).

Операторы ANY и ALL используются  в SQL для сравнения некоторого значения с результирующим набором вложенного запроса, состоящим из одного столбца. При этом тип данных столбца, возвращаемого вложенным запросом, должен совпадать с типом данных столбца (или выражения), с которым происходит сравнение.

При использовании оператора ANY в результирующую таблицу будут включены все записи, для которых  выражение со знаком отношения верно хотя бы для одного элемента результирующего запроса. Как работает оператор ANY:

* amount > ANY (10, 12) эквивалентно amount > 10
* amount < ANY (10, 12) эквивалентно amount < 12
* amount = ANY (10, 12) эквивалентно (amount = 10) OR (amount = 12), а также amount IN  (10,12)
* amount <> ANY (10, 12) вернет все записи с любым значением amount, включая 10 и 12

При использовании оператора ALL в результирующую таблицу будут включены все записи, для которых  выражение со знаком отношения верно для всех элементов результирующего запроса. Как работает оператор ALL:

* amount > ALL (10, 12) эквивалентно amount > 12
* amount < ALL (10, 12) эквивалентно amount < 10
* amount = ALL (10, 12) не вернет ни одной записи, так как эквивалентно (amount = 10) AND (amount = 12)
* amount <> ALL (10, 12) вернет все записи кроме тех,  в которыхamount равно 10 или 12

**Важно!** Операторы **ALL** и **ANY** можно использовать т**олько с вложенными запросами**. В примерах выше (10, 12) приводится как результат вложенного запроса просто для того, чтобы показать как эти операторы работают. В запросах так записывать нельзя.

**Пример**

Вывести информацию о тех книгах, количество которых меньше самого маленького среднего количества книг каждого автора.

*Запрос:*

SELECT title, author, amount, price

FROM book

WHERE amount < ALL (

SELECT AVG(amount)

FROM book

GROUP BY author

);

*Результат:*

+--------------------+------------------+--------+--------+

| title | author | amount | price |

+--------------------+------------------+--------+--------+

| Мастер и Маргарита | Булгаков М.А. | 3 | 670.99 |

| Братья Карамазовы | Достоевский Ф.М. | 3 | 799.01 |

+--------------------+------------------+--------+--------+

**Пояснение**

1. Вложенный запрос

SELECT AVG(amount)

        FROM book

        GROUP BY author

отбирает следующие записи:

+-------------+

| AVG(amount) |

+-------------+

| 4.0000 |

| 7.6667 |

| 15.0000 |

+-------------+

2. Условие отбора в основном запросе

amount < ALL (

        SELECT AVG(amount)

        FROM book

        GROUP BY author

      )

можно переписать (если заменить вложенный запрос списком отобранных значений):

amount < ALL ( 4.0000, 7.6667, 15.0000)

что в соответствии с определением **ALL**, это значит, что подходят все **amount** меньшие **4.000**.

Таким образом, наш запрос отобрал все книги книги **Мастер и Маргарита** и **Братья Карамазовы**, количество которых равно 3.

**Пример**

Вывести информацию о тех книгах, количество которых меньше самого большого среднего количества книг каждого автора.

*Запрос:*

SELECT title, author, amount, price

FROM book

WHERE amount < ANY (

SELECT AVG(amount)

FROM book

GROUP BY author

);

*Результат:*

+--------------------+------------------+--------+--------+

| title | author | amount | price |

+--------------------+------------------+--------+--------+

| Мастер и Маргарита | Булгаков М.А. | 3 | 670.99 |

| Белая гвардия | Булгаков М.А. | 5 | 540.50 |

| Идиот | Достоевский Ф.М. | 10 | 460.00 |

| Братья Карамазовы | Достоевский Ф.М. | 3 | 799.01 |

| Игрок | Достоевский Ф.М. | 10 | 480.50 |

+--------------------+------------------+--------+--------+

**Пояснение**

В этом примере **amount < ANY ( 4.0000, 7.6667, 15.0000)** означает, что подходят **amount** меньше самого большого значения из списка.

**Задание**

Вывести информацию о книгах(автор, название, цена), цена которых меньше самой большой из минимальных цен, вычисленных для каждого автора.

**Результат**

+------------------+---------------+--------+

| author | title | price |

+------------------+---------------+--------+

| Булгаков М.А. | Белая гвардия | 540.50 |

| Достоевский Ф.М. | Идиот | 460.00 |

| Достоевский Ф.М. | Игрок | 480.50 |

+------------------+---------------+--------+

**Структура и наполнение таблицы book**

**Введите SQL запрос**

SELECT author, title, price

FROM book

WHERE price < ANY (

SELECT MIN(price)

FROM book

GROUP BY author

);

## Вложенный запрос после SELECT

Вложенный запрос может располагаться после ключевого слова SELECT. В этом случае результат выполнения запроса выводится в отдельном столбце результирующей таблицы. При этом результатом запроса может быть только одно значение, тогда оно будет повторяться во всех строках. Также вложенный запрос может использоваться в выражениях.

**Пример**

Вывести информацию о книгах, количество экземпляров которых отличается от среднего количества экземпляров книг на складе более чем на 3,  а также указать среднее значение количества экземпляров книг.

Запрос:

SELECT title, author, amount,

(

SELECT AVG(amount)

FROM book

) AS Среднее\_количество

FROM book

WHERE abs(amount - (SELECT AVG(amount) FROM book)) >3;

**Пояснение**

В запросе используется функция [модуля](https://stepik.org/lesson/297509/step/6?unit=279269), которая позволяет учесть, что количество может отличаться от среднего как в большую, так и в меньшую сторону.

Результат:

+-----------------------+------------------+--------+--------------------+

| title | author | amount | Среднее\_количество |

+-----------------------+------------------+--------+--------------------+

| Мастер и Маргарита | Булгаков М.А. | 3 | 7.6667 |

| Братья Карамазовы | Достоевский Ф.М. | 3 | 7.6667 |

| Стихотворения и поэмы | Есенин С.А. | 15 | 7.6667 |

+-----------------------+------------------+--------+--------------------+

Во вложенном запросе вычисляется среднее количество экземпляров книг на складе. Этот запрос используется и в условии отбора, и для создания столбца **Среднее\_количество** в результирующей таблице запроса. Значения  столбца одинаковы во всех строках, поскольку  вложенный запрос возвращает одно значение.

Среднее количество в виде дробного числа выглядит не очень правильно. Полученное значение можно [округлить "вниз"](https://stepik.org/lesson/297509/step/6?unit=279269) - до ближайшего меньшего целого.

Запрос:

SELECT title, author, amount,

FLOOR((SELECT AVG(amount) FROM book)) AS Среднее\_количество

FROM book

WHERE ABS(amount - (SELECT AVG(amount) FROM book)) >3;

Результат:

+-----------------------+------------------+--------+--------------------+

| title | author | amount | Среднее\_количество |

+-----------------------+------------------+--------+--------------------+

| Мастер и Маргарита | Булгаков М.А. | 3 | 7 |

| Братья Карамазовы | Достоевский Ф.М. | 3 | 7 |

| Стихотворения и поэмы | Есенин С.А. | 15 | 7 |

+-----------------------+------------------+--------+--------------------+

## Задание

Посчитать сколько и каких экземпляров книг нужно заказать поставщикам, чтобы на складе стало одинаковое количество экземпляров каждой книги, равное значению самого большего количества экземпляров одной книги на складе. Вывести название книги, ее автора, текущее количество экземпляров на складе и количество заказываемых экземпляров книг. Последнему столбцу присвоить имя **Заказ**. В результат не включать книги, которые заказывать не нужно.

**Результат**

+--------------------+------------------+--------+-------+

| title | author | amount | Заказ |

+--------------------+------------------+--------+-------+

| Мастер и Маргарита | Булгаков М.А. | 3 | 12 |

| Белая гвардия | Булгаков М.А. | 5 | 10 |

| Идиот | Достоевский Ф.М. | 10 | 5 |

| Братья Карамазовы | Достоевский Ф.М. | 3 | 12 |

| Игрок | Достоевский Ф.М. | 10 | 5 |

+--------------------+------------------+--------+-------+

**Пояснение**

Поскольку книгу с максимальным количеством экземпляров заказывать не нужно, в условии отбора запроса укажите, что книгу с максимальным значением количества в результирующую таблицу не включать.

**Структура и наполнение таблицы book**

### Введите SQL запрос

SELECT title, author, amount,

(SELECT MAX(amount) FROM book) - amount as Заказ

FROM book

WHERE ((SELECT MAX(amount) FROM book) - amount) >0;

## 1.5 Запросы корректировки данных

## Содержание урока

SQL позволяет не только выбирать данные из таблиц базы данных, но и корректировать информацию в них. Для этого используются запросы корректировки данных, с помощью которых можно:

* [создать пустую таблицу](https://stepik.org/lesson/305012/step/2?unit=287020);
* [добавить в таблицу записи как совокупность значений](https://stepik.org/lesson/305012/step/3?unit=287020);
* [добавить записи из другой таблицы](https://stepik.org/lesson/305012/step/4?unit=287020);
* [добавить записи из другой таблицы, используя вложенный запрос](https://stepik.org/lesson/305012/step/5?unit=287020);
* [изменить значения в одном столбце](https://stepik.org/lesson/305012/step/6?unit=287020);
* [изменить значения в нескольких столбцах](https://stepik.org/lesson/305012/step/7?unit=287020);
* [изменить данные, используя несколько таблиц](https://stepik.org/lesson/305012/step/8?unit=287020);
* [удалить записи из таблицы](https://stepik.org/lesson/305012/step/9?unit=287020);
* [создать таблицу на основе данных других таблиц](https://stepik.org/lesson/305012/step/10?unit=287020).

На данном уроке будут рассматриваться запросы для реализации типичных для склада действий:

* получение нового товара ([добавление](https://stepik.org/lesson/305012/step/5?unit=287020), [обновление](https://stepik.org/lesson/305012/step/8?unit=287020), [удаление данных](https://stepik.org/lesson/305012/step/9?unit=287020));
* продажа товара ([обновление данных](https://stepik.org/lesson/305012/step/7?unit=287020));
* формирование заказа на новый товар ([создание таблицы](https://stepik.org/lesson/305012/step/10?unit=287020)).

## Создание пустой таблицы

Создание таблицы осуществляется с помощью запроса CREATE, подробно рассмотренного в [первом уроке модуля](https://stepik.org/lesson/297508/step/7?unit=279268).

## Задание

Создать таблицу поставка (**supply**), которая имеет ту же структуру, что и таблиц **book**.

|  |  |
| --- | --- |
| **Поле** | **Тип, описание** |
| **supply\_id** | INT PRIMARY KEY AUTO\_INCREMENT |
| **title** | VARCHAR(50) |
| **author** | VARCHAR(30) |
| **price** | DECIMAL(8, 2) |
| **amount** | INT |

### Введите SQL запрос

CREATE TABLE supply (

supply\_id INT PRIMARY KEY AUTO\_INCREMENT,

title VARCHAR(50),

author VARCHAR(30),

price DECIMAL(8,2),

amount INT

);

## Добавление записей в таблицу

Добавление одной записи в таблицу осуществляется с помощью запроса INSERT, подробно рассмотренного в [первом уроке](https://stepik.org/lesson/297508/step/8?unit=279268). Запросы обязательно разделять точкой с запятой.

Допускается вставка нескольких записей одновременно, для этого используется SQL запрос следующего вида:

INSERT INTO имя\_таблицы(столбец\_1, столбец\_2, ..., столбец\_N)

VALUES

(значение\_1\_1, значение\_1\_2, ..., значение\_1\_N),

(значение\_2\_1, значение\_2\_2, ..., значение\_2\_N),

...

(значение\_M\_1, значение\_M\_2, ..., значение\_M\_N);

Например, чтобы добавить в таблицу **book** две новые записи используется запрос:

INSERT INTO book (title, author, price, amount)

VALUES

('Война и мир','Толстой Л.Н.', 1070.20, 2),

('Анна Каренина', 'Толстой Л.Н.', 599.90, 3);

## Задание

Занесите в таблицу**supply** четыре записи, чтобы получилась следующая таблица:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **supply\_id** | **title** | **author** | **price** | **amount** |
| 1 | Лирика | Пастернак Б.Л. | 518.99 | 2 |
| 2 | Черный человек | Есенин С.А. | 570.20 | 6 |
| 3 | Белая гвардия | Булгаков М.А. | 540.50 | 7 |
| 4 | Идиот | Достоевский Ф.М. | 360.80 | 3 |

**Пояснение**

**Результат**

Affected rows: 1

Affected rows: 1

Affected rows: 1

Affected rows: 1

Query result:

+-----------+----------------+------------------+--------+--------+

| supply\_id | title | author | price | amount |

+-----------+----------------+------------------+--------+--------+

| 1 | Лирика | Пастернак Б.Л. | 518.99 | 2 |

| 2 | Черный человек | Есенин С.А. | 570.20 | 6 |

| 3 | Белая гвардия | Булгаков М.А. | 540.50 | 7 |

| 4 | Идиот | Достоевский Ф.М. | 360.80 | 3 |

+-----------+----------------+------------------+--------+--------+

### Введите SQL запрос

INSERT INTO supply (title, author, price, amount) VALUES

('Лирика', 'Пастернак Б.Л.', 518.99, 2),

('Черный человек', 'Есенин С.А.', 570.20, 6),

('Белая гвардия', 'Булгаков М.А.', 540.50, 7),

('Идиот', 'Достоевский Ф.М.', 360.80, 3);

SELECT \* FROM supply;

**Добавление записей из другой таблицы**

С помощью запроса на добавление можно не только добавить в таблицу конкретные значения (список VALUES), но и записи из другой таблицы, отобранные с помощью запроса на выборку.  В этом случае вместо раздела VALUES записывается запрос на выборку, начинающийся с SELECT.  В нем можно использовать WHERE, GROUP BY, ORDER BY.

Правила соответствия между полями таблицы и вставляемыми значениями из запроса:

1. количество полей в таблице и количество полей в запросе должны совпадать;
2. должно существовать прямое соответствие между позицией одного и того же элемента в обоих списках, поэтому первый столбец запроса должен относиться к первому столбцу в списке столбцов таблицы, второй – ко второму столбцу и т.д.
3. типы столбцов запроса должны быть совместимы с типами данных соответствующих столбцов таблицы ( целое число можно занести в поле типа DECIMAL, обратная операция – недопустима).

**Пример**

Занести все книги из таблицы **supply** в таблицу **book**.

*Запрос:*

INSERT INTO book (title, author, price, amount)

SELECT title, author, price, amount

FROM supply;

SELECT \* FROM book;

*Результат:*

Affected rows: 4

Query result:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

| 6 | Лирика | Пастернак Б.Л. | 518.99 | 2 |

| 7 | Черный человек | Есенин С.А. | 570.20 | 6 |

| 8 | Белая гвардия | Булгаков М.А. | 540.50 | 7 |

| 9 | Идиот | Достоевский Ф.М. | 360.80 | 3 |

+---------+-----------------------+------------------+--------+--------+

Affected rows: 9

С помощью этого запроса в таблицу **book** включены все книги из **supply**, даже те, которые в **book** уже есть («Белая гвардия» и «Идиот»). В результате в таблице одна и та же книга, например «Белая гвардия», имеет код 2 и 8. Для реляционной модели это нежелательная ситуация. Устранить эту проблему можно с помощью вложенных запросов, которые будут рассмотрены в следующем шаге.

**Задание**

Добавить из таблицы **supply** в таблицу **book**, все книги, кроме книг, написанных Булгаковым М.А. и Достоевским Ф.М.

**Результат**

Affected rows: 2

Query result:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

| 6 | Лирика | Пастернак Б.Л. | 518.99 | 2 |

| 7 | Черный человек | Есенин С.А. | 570.20 | 6 |

+---------+-----------------------+------------------+--------+--------+

**Пояснение**

Задание нужно выполнить без вложенных запросов.

**Структура и наполнение таблиц book и supply**

**Введите SQL запрос**

INSERT INTO book (title, author, price, amount)

SELECT title, author, price, amount

FROM supply

WHERE author NOT IN( 'Булгаков М.А.' , 'Достоевский Ф.М.');

SELECT \* FROM book;

## Добавление записей, вложенные запросы

В запросах на добавление можно использовать вложенные запросы.

**Пример**

Занести из таблицы **supply** в таблицу **book** только те книги, названия которых отсутствуют в таблице **book.**

Запрос:

INSERT INTO book (title, author, price, amount)

SELECT title, author, price, amount

FROM supply

WHERE title NOT IN (

SELECT title

FROM book

);

SELECT \* FROM book;

Результат:

Affected rows: 2

Query result:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

| 6 | Лирика | Пастернак Б.Л. | 518.99 | 2 |

| 7 | Черный человек | Есенин С.А. | 570.20 | 6 |

+---------+-----------------------+------------------+--------+--------+

Вложенным запросом отбираются все названия книг, которые есть в таблице **book**. Основным запросом SELECT из таблицы **supply** выбираются книги, названия которых нет в результате вложенного запроса. Отобранные записи добавляются в конец таблицы **book**запросом на добавление INSERT.

## Задание

Занести из таблицы **supply** в таблицу **book** только те книги, авторов которых нет в  **book.**

**Результат**

Affected rows: 1

Query result:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

| 6 | Лирика | Пастернак Б.Л. | 518.99 | 2 |

+---------+-----------------------+------------------+--------+--------+

**Структура и наполнение таблиц book и supply**

### Введите SQL запрос

INSERT INTO book (title, author, price, amount)

SELECT title, author, price, amount

FROM supply

WHERE author NOT IN (

SELECT author

FROM book

);

SELECT \* FROM book;

## Запросы на обновление

Под обновлением данных подразумевается изменение значений в существующих записях таблицы. При этом возможно как изменение значений полей в группе строк (даже всех строк таблицы), так и правка значения поля отдельной строки.

Изменение записей в таблице реализуется с помощью запроса UPDATE. Простейший запрос на  обновление выглядит так:

UPDATE таблица SET поле = выражение

где   
**таблица** – имя таблицы, в которой будут проводиться изменения;  
**поле** – поле таблицы, в которое будет внесено изменение;  
**выражение** – выражение,  значение которого будет занесено в поле.

**Пример**

Уменьшить на 30% цену книг в таблице **book**.

Запрос:

UPDATE book

SET price = 0.7 \* price;

SELECT \* FROM book;

Результат:

Affected rows: 5

Query result:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 469.69 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 378.35 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 322.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 559.31 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 455.00 | 15 |

+---------+-----------------------+------------------+--------+--------+

С помощью запросов на обновление можно изменять не все записи в таблице (как в предыдущем запросе), а только часть из них. Для этого в запрос включается ключевое слово WHERE, после которого указывается условие отбора строк для изменения.

**Пример**

Уменьшить на 30% цену тех книг в таблице **book**, количество которых меньше 5.

Запрос:

UPDATE book

SET price = 0.7 \* price

WHERE amount < 5;

SELECT \* FROM book;

Результат:

Affected rows: 2

Query result:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 469.69 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 559.31 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

+---------+-----------------------+------------------+--------+--------+

В этом запросе обновляется только 2 записи (цена книг «Мастер и Маргарита» и «Братья Карамазовы»).

## Задание

Уменьшить на 10% цену тех книг в таблице **book**, количество которых принадлежит интервалу от 5 до 10, включая границы.

**Результат**

Affected rows: 2

Query result:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 486.45 | 5 |

| 3 | Идиот | Достоевский Ф.М. | 414.00 | 10 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

+---------+-----------------------+------------------+--------+--------+

**Структура и наполнение таблицы book**

### Введите SQL запрос

UPDATE book

SET price = price \* 0.9

WHERE amount BETWEEN 5 AND 10;

**Запросы на обновление нескольких столбцов**

Запросом UPDATE можно обновлять значения нескольких столбцов одновременно. В этом случае простейший запрос будет выглядеть так:

UPDATE таблица SET поле1 = выражение1, поле2 = выражение2

На складе, кроме хранения и получения книг, выполняется их оптовая продажа. Для реализации этого действия включим дополнительный столбец **buy**  в таблицу **book**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author** | **price** | **amount** | **buy** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(50)** | **VARCHAR(30)** | **DECIMAL(8,2)** | **INT** | int |
| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 | 0 |
| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 | 3 |
| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 | 8 |
| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 | 0 |
| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 | 18 |

**Пример**

В столбце **buy** покупатель указывает количество книг, которые он хочет приобрести. Для каждой книги, выбранной покупателем, необходимо уменьшить ее количество на складе на указанное в столбце**buy** количество, а в столбец **buy** занести 0.

*Запрос:*

UPDATE book

SET amount = amount - buy,

buy = 0;

SELECT \* FROM book;

*Результат:*

Affected rows: 3

Query result:

+---------+-----------------------+------------------+--------+--------+-----+

| book\_id | title | author | price | amount | buy |

+---------+-----------------------+------------------+--------+--------+-----+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 | 0 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 2 | 0 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 2 | 0 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 | 0 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | -3 | 0 |

+---------+-----------------------+------------------+--------+--------+-----+

Как видно из таблицы, без проверки данных, которые занесены в столбец,  нельзя запускать запрос на обновление (может получиться отрицательное значение количества).

**Задание**

В таблице **book** необходимо скорректировать значение для покупателя в столбце **buy**таким образом, чтобы оно не превышало количество экземпляров книг, указанных в столбце **amount**. А цену тех книг, которые покупатель не заказывал, снизить на 10%.

**Результат**

Affected rows: 3

Query result:

+---------+-----------------------+------------------+--------+--------+-----+

| book\_id | title | author | price | amount | buy |

+---------+-----------------------+------------------+--------+--------+-----+

| 1 | Мастер и Маргарита | Булгаков М.А. | 603.89 | 3 | 0 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 | 3 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 | 8 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 719.11 | 2 | 0 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 | 15 |

+---------+-----------------------+------------------+--------+--------+-----+

**Пояснение**

Запрос на обновление количества книг должен корректировать значения в столбце **buy**  таблицы **book** следующим образом:

* если покупатель заказал количество книг больше, чем есть на складе, то заменить значение **buy** на имеющееся на складе количество **amount**;
* если покупатель хочет купить количество книг меньшее или равное количеству книг на складе, то значение **buy** изменять не надо.

Для реализации этого запроса можно  использовать функцию [**if()**](https://stepik.org/lesson/297509/step/7?unit=279269).  Синтаксис раздела SET при использовании функции **if()**следующий:

SET столбец = IF(условие, выражение\_1, выражение\_2)

Выполняется этот оператор так:

* сначала вычисляется **условие**;
* если условие ИСТИНА, то вычисляется **выражение\_1**, в противном случае (если условие ЛОЖНО) вычисляется **выражение\_2**;
* в столбец заносится результат выполнения функции (либо значение **выражения\_1**, либо значение **выражения\_2** в зависимости от условия).

Например, для увеличения на 10% только цен книг Булгакова используется запрос:

UPDATE book

SET price = IF(author = "Булгаков М.А.", price \* 1.1, price);

**Структура и наполнение таблицы book**

**Введите SQL запрос**

UPDATE book

SET buy = IF(buy >= amount, amount, buy),

price = IF(buy = 0, 0.9 \* price, price);

SELECT \* FROM book;

**Запросы на обновление нескольких таблиц**

В запросах на обновление можно использовать несколько таблиц, но тогда

* для столбцов, имеющих одинаковые имена, необходимо указывать имя таблицы, к которой они относятся, например, **book.price** – столбец **price** из таблицы **book**, **supply.price** – столбец **price** из таблицы **supply**;
* все таблицы, используемые в запросе, нужно перечислить после ключевого слова UPDATE;
* в запросе обязательно условие WHERE, в котором указывается условие при котором обновляются данные.

**Пример**

Если в таблице **supply**  есть те же книги, что и в таблице **book**, добавлять эти книги в таблицу **book** не имеет смысла. Необходимо увеличить их количество на значение столбца **amount**таблицы **supply**.

*Запрос:*

UPDATE book, supply

SET book.amount = book.amount + supply.amount

WHERE book.title = supply.title AND book.author = supply.author;

SELECT \* FROM book;

*Результат:*

Affected rows: 2

Query result:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 12 |

| 3 | Идиот | Достоевский Ф.М. | 460.00 | 13 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

+---------+-----------------------+------------------+--------+--------+

В этом запросе увеличилось количество двух книг: «Белая гвардия», которая в **supply** имеет ту же цену, и «Идиот», но цена этой книги в таблицах **book** и **supply** отличается. Для этой книги нужно пересчитать цену.

**Задание**

Для тех книг в таблице **book** , которые есть в таблице **supply,**не только увеличить их количество в таблице **book** ( увеличить их количество на значение столбца **amount**таблицы **supply**), но и пересчитать их цену (для каждой книги найти сумму цен из таблиц **book** и **supply** и разделить на 2).

**Результат**

Affected rows: 2

Query result:

+---------+-----------------------+------------------+--------+--------+

| book\_id | title | author | price | amount |

+---------+-----------------------+------------------+--------+--------+

| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |

| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 12 |

| 3 | Идиот | Достоевский Ф.М. | 410.40 | 13 |

| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |

| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

+---------+-----------------------+------------------+--------+--------+

**Пояснение**

Пересчет для книг с одинаковым названием и ценой не повлияет на результат, поэтому в запросе не обязательно рассматривать два случая: когда цена у одинаковых книг равна и когда нет.

**Структура и наполнение таблиц book и supply**

**Введите SQL запрос**

UPDATE book, supply

SET book.amount = book.amount + supply.amount,

book.price = (book.price + supply.price)/2

WHERE book.title = supply.title AND

book.author = supply.author;

SELECT \* FROM book;

## Запросы на удаление

Запросы корректировки данных позволяют удалить одну или несколько записей из  таблицы. Простейший запрос на удаление имеет вид:

DELETE FROM таблица;

Этот запрос удаляет все записи из указанной после FROM таблицы.

**Пример**

После того, как информация о книгах из таблицы **supply** перенесена в **book** , необходимо очистить таблицу  **supply**.

Запрос:

DELETE FROM supply;

SELECT \* FROM supply;

Результат:

Affected rows: 4

Affected rows: 0

Из таблицы удалены все записи. Запрос на выборку отобрал 0 записей.

Запрос на удаления позволяет удалить не все записи таблицы, а только те, которые удовлетворяют условию, указанному после ключевого слова WHERE:

DELETE FROM таблица

WHERE условие;

**Пример**

Удалить из таблицы **supply** все книги, названия которых есть в таблице **book**.

Запрос:

DELETE FROM supply

WHERE title IN (

SELECT title

FROM book

);

SELECT \* FROM supply;

Результат:

Affected rows: 2

Query result:

+-----------+--------------------------+------------------+--------+--------+

| supply\_id | title | author | price | amount |

+-----------+--------------------------+------------------+--------+--------+

| 1 | Лирика | Пастернак Б.Л. | 518.99 | 2 |

| 2 | Черный человек | Есенин С.А. | 570.20 | 6 |

+-----------+--------------------------+------------------+--------+--------+

Из таблицы **supply** удалены две записи о книгах «Белая гвардия» и «Идиот».

## ****Задание****

Удалить из таблицы **supply** книги тех авторов, общее количество экземпляров книг которых в таблице **book** превышает 10.

**Результат**

Affected rows: 2

Query result:

+-----------+---------------+----------------+--------+--------+

| supply\_id | title | author | price | amount |

+-----------+---------------+----------------+--------+--------+

| 1 | Лирика | Пастернак Б.Л. | 518.99 | 2 |

| 3 | Белая гвардия | Булгаков М.А. | 540.50 | 7 |

+-----------+---------------+----------------+--------+--------+

**Структура и наполнение таблиц book и supply**

### Введите SQL запрос

DELETE FROM supply

WHERE author IN (

SELECT author

FROM book

GROUP by author

HAVING SUM(amount) > 10

);

SELECT \* FROM supply;

## Запросы на создание таблицы

Новая таблица может быть создана на основе данных из другой таблицы. Для этого используется запрос SELECT, результирующая таблица которого и будет новой таблицей базы данных. При этом имена столбцов запроса становятся именами столбцов новой таблицы. Запрос на создание новой таблицы имеет вид:

CREATE TABLE имя\_таблицы AS

SELECT ...

**Пример**

Создать таблицу заказ (**ordering**), куда включить авторов и названия тех книг, количество экземпляров которых в таблице **book** меньше 4. Для всех книг указать одинаковое количество экземпляров 5.

Запрос:

CREATE TABLE ordering AS

SELECT author, title, 5 AS amount

FROM book

WHERE amount < 4;

SELECT \* FROM ordering;

Результат:

Affected rows: 2

Query result:

+------------------+--------------------+--------+

| author | title | amount |

+------------------+--------------------+--------+

| Булгаков М.А. | Мастер и Маргарита | 5 |

| Достоевский Ф.М. | Братья Карамазовы | 5 |

+------------------+--------------------+--------+

При создании таблицы можно использовать вложенные запросы как после SELECT, так и после WHERE.

**Пример**

Создать таблицу заказ (**ordering**), куда включить авторов и названия тех книг, количество экземпляров которых в таблице **book** меньше 4. Для всех книг указать одинаковое значение - среднее количество экземпляров книг в таблице **book**.

Запрос:

CREATE TABLE ordering AS

SELECT author, title,

(

SELECT ROUND(AVG(amount))

FROM book

) AS amount

FROM book

WHERE amount < 4;

SELECT \* FROM ordering;

Результат:

Affected rows: 2

Query result:

+------------------+--------------------+--------+

| author | title | amount |

+------------------+--------------------+--------+

| Булгаков М.А. | Мастер и Маргарита | 7 |

| Достоевский Ф.М. | Братья Карамазовы | 7 |

+------------------+--------------------+--------+

## Задание

Создать таблицу заказ (**ordering**), куда включить авторов и названия тех книг, количество экземпляров которых в таблице **book** меньше среднего количества экземпляров книг в таблице **book**. В таблицу включить столбец   **amount,** в котором для всех книг указать одинаковое значение - среднее количество экземпляров книг в таблице **book**.

**Результат** **Структура и наполнение таблицы book**

### Введите SQL запрос

CREATE TABLE ordering AS

SELECT author, title, (

SELECT ROUND(AVG(amount))

FROM book

) AS amount

FROM book

WHERE amount < (SELECT ROUND(AVG(amount)) FROM book);

SELECT \* FROM ordering;

## 1.6 Таблица "Командировки", запросы на выборку

## Содержание урока

В этом уроке на каждом шаге нужно реализовать запросы на выборку для таблицы, в которой представлена информация о командировках сотрудников некоторой организации:

* [Вывести информацию о командировках тех сотрудников, фамилия которых заканчивается на букву «а»](https://stepik.org/lesson/297510/step/2?unit=279270).
* [Вывести в алфавитном порядке фамилии, имена и отчества тех сотрудников, которые были в командировке в Москве](https://stepik.org/lesson/297510/step/3?unit=279270).
* [Для каждого города посчитать, сколько раз сотрудники в нем были](https://stepik.org/lesson/297510/step/4?unit=279270).
* [Вывести два города, в которых чаще всего были в командировках сотрудники](https://stepik.org/lesson/297510/step/5?unit=279270).
* [Вывести информацию о длительности командировок сотрудников](https://stepik.org/lesson/297510/step/6?unit=279270).
* [Вывести информацию о командировках сотрудника(ов), которые были самыми короткими по времени](https://stepik.org/lesson/297510/step/7?unit=279270).
* [Вывести информацию о командировках, начало и конец которых относятся к одному месяцу](https://stepik.org/lesson/297510/step/8?unit=279270).
* [Вывести номер месяца и количество командировок, первый день которых приходился на этот месяц](https://stepik.org/lesson/297510/step/9?unit=279270).
* [Вывести сумму суточных  для командировок сотрудников](https://stepik.org/lesson/297510/step/10?unit=279270).
* [Вывести фамилию с инициалами и общую сумму суточных, полученных за все командировки для тех сотрудников, которые были в командировках больше чем 3 раза](https://stepik.org/lesson/297510/step/11?unit=279270).

## Структура и наполнение таблицы

Таблица **trip**, в которой представлена информация о командировках сотрудников некоторой организации (фамилия сотрудника, город, куда он ездил, размер суточных, даты первого и последнего дня командировки) :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **trip\_id** | **name** | **city** | **per\_diem** | **date\_first** | **date\_last** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(30)** | **VARCHAR(25)** | **DECIMAL(8,2)** | **DATЕ** | **DATE** |
| 1 | Баранов П.Е. | Москва | 700 | 2020-01-12 | 2020-01-17 |
| 2 | Абрамова К.А. | Владивосток | 450 | 2020-01-14 | 2020-01-27 |
| 3 | Семенов И.В. | Москва | 700 | 2020-01-23 | 2020-01-31 |
| 4 | Ильиных Г.Р. | Владивосток | 450 | 2020-01-12 | 2020-02-02 |
| 5 | Колесов С.П. | Москва | 700 | 2020-02-01 | 2020-02-06 |
| 6 | Баранов П.Е. | Москва | 700 | 2020-02-14 | 2020-02-22 |
| 7 | Абрамова К.А. | Москва | 700 | 2020-02-23 | 2020-03-01 |
| 8 | Лебедев Т.К. | Москва | 700 | 2020-03-03 | 2020-03-06 |
| 9 | Колесов С.П. | Новосибирск | 450 | 2020-02-27 | 2020-03-12 |
| 10 | Семенов И.В. | Санкт-Петербург | 700 | 2020-03-29 | 2020-04-05 |
| 11 | Абрамова К.А. | Москва | 700 | 2020-04-06 | 2020-04-14 |
| 12 | Баранов П.Е. | Новосибирск | 450 | 2020-04-18 | 2020-05-04 |
| 13 | Лебедев Т.К. | Томск | 450 | 2020-05-20 | 2020-05-31 |
| 14 | Семенов И.В. | Санкт-Петербург | 700 | 2020-06-01 | 2020-06-03 |
| 15 | Абрамова К.А. | Санкт-Петербург | 700 | 2020-05-28 | 2020-06-04 |
| 16 | Федорова А.Ю. | Новосибирск | 450 | 2020-05-25 | 2020-06-04 |
| 17 | Колесов С.П. | Новосибирск | 450 | 2020-06-03 | 2020-06-12 |
| 18 | Федорова А.Ю. | Томск | 450 | 2020-06-20 | 2020-06-26 |
| 19 | Абрамова К.А. | Владивосток | 450 | 2020-07-02 | 2020-07-13 |
| 20 | Баранов П.Е. | Воронеж | 450 | 2020-07-19 | 2020-07-25 |

**Пояснение**

Тип **DATE** – позволяет описать дату в формате ГГГГ-ММ-ДД, например, 2020-02-02. При вставке данных в таблицу с помощью INSERT INTO ... VALUES значение даты заключается в кавычки.

## Задание

Вывести из таблицы **trip** информацию о командировках тех сотрудников, фамилия которых заканчивается на букву «а», в отсортированном по убыванию даты последнего дня командировки виде. В результат включить столбцы **name, city, per\_diem, date\_first, date\_last.**

**Результат**

+---------------+-----------------+----------+------------+------------+

| name | city | per\_diem | date\_first | date\_last |

+---------------+-----------------+----------+------------+------------+

| Абрамова К.А. | Владивосток | 450.00 | 2020-07-02 | 2020-07-13 |

| Федорова А.Ю. | Томск | 450.00 | 2020-06-20 | 2020-06-26 |

| Абрамова К.А. | Санкт-Петербург | 700.00 | 2020-05-28 | 2020-06-04 |

| Федорова А.Ю. | Новосибирск | 450.00 | 2020-05-25 | 2020-06-04 |

| Абрамова К.А. | Москва | 700.00 | 2020-04-06 | 2020-04-14 |

| Абрамова К.А. | Москва | 700.00 | 2020-02-23 | 2020-03-01 |

| Абрамова К.А. | Владивосток | 450.00 | 2020-01-14 | 2020-01-27 |

+---------------+-----------------+----------+------------+------------+

**Связанные шаги**

* [выборка столбцов](https://stepik.org/lesson/297509/step/3?unit=279269);
* [условие отбора](https://stepik.org/lesson/297509/step/12?unit=279269);
* [сортировка](https://stepik.org/lesson/297509/step/11?unit=279269).

**Структура и наполнение таблицы trip**

### Введите SQL запрос

SELECT name, city, per\_diem, date\_first, date\_last

FROM trip

WHERE name LIKE '%а %'

ORDER BY date\_last DESC

;

## Задание

Вывести в алфавитном порядке фамилии и инициалы тех сотрудников, которые были в командировке в Москве.

**Результат**

+---------------+

| name |

+---------------+

| Абрамова К.А. |

| Баранов П.Е. |

| Колесов С.П. |

| Лебедев Т.К. |

| Семенов И.В. |

+---------------+

**Связанные шаги**

* [выборка столбцов](https://stepik.org/lesson/297509/step/3?unit=279269);
* [уникальные записи](https://stepik.org/lesson/297515/step/2?unit=279275);
* [условие отбора](https://stepik.org/lesson/297509/step/8?unit=279269);
* [сортировка](https://stepik.org/lesson/297509/step/11?unit=279269).

**Структура и наполнение таблицы trip**

### Введите SQL запрос

SELECT DISTINCT name

FROM trip

WHERE city = 'Москва'

ORDER by name

;

## Задание

Для каждого города посчитать, сколько раз сотрудники в нем были.  Информацию вывести в отсортированном в алфавитном порядке по названию городов. Вычисляемый столбец назвать **Количество**.

**Результат**

+-----------------+------------+

| city | Количество |

+-----------------+------------+

| Владивосток | 3 |

| Воронеж | 1 |

| Москва | 7 |

| Новосибирск | 4 |

| Санкт-Петербург | 3 |

| Томск | 2 |

+-----------------+------------+

**Связанные шаги**

* [выборка столбцов и их именование](https://stepik.org/lesson/297509/step/4?unit=279269);
* [вычисления для сгруппированных данных](https://stepik.org/lesson/297515/step/3?unit=279275);
* [сортировка](https://stepik.org/lesson/297509/step/11?unit=279269).

### Введите SQL запрос

SELECT city, COUNT(city) AS Количество

FROM trip

GROUP BY city

ORDER BY city;

## ****Оператор LIMIT****

Для ограничения вывода записей в SQL используется оператор LIMIT , после которого указывается количество строк.  Результирующая таблица будет иметь количество строк не более указанного после LIMIT. LIMIT размещается после раздела ORDER BY.

Как правило, этот оператор используется, чтобы отобрать заданное количество отсортированных строк результата запроса.

**Пример**

Вывести информацию о первой  командировке из таблицы **trip**. "Первой" считать командировку с самой ранней датой начала.

Запрос:

SELECT \*

FROM trip

ORDER BY  date\_first

LIMIT 1;

Результат:

+---------+--------------+--------+----------+------------+------------+

| trip\_id | name | city | per\_diem | date\_first | date\_last |

+---------+--------------+--------+----------+------------+------------+

| 1 | Баранов П.Е. | Москва | 700.00 | 2020-01-12 | 2020-01-17 |

+---------+--------------+--------+----------+------------+------------+

**Важно.** Оператор LIMIT нужно использовать очень осторожно. Например, если бы в таблице **trip** было несколько командировок с одинаковой датой начала, этот запрос работал бы НЕВЕРНО. Это связано с тем, что заранее не известно точное значение таких командировок.

## Задание

Вывести два города, в которых чаще всего были в командировках сотрудники. Вычисляемый столбец назвать **Количество**.

**Результат**

+-------------+------------+

| city | Количество |

+-------------+------------+

| Москва | 7 |

| Новосибирск | 4 |

+-------------+------------+

**Связанные шаги**

* [выборка столбцов и их именование](https://stepik.org/lesson/297509/step/4?unit=279269);
* [вычисления для сгруппированных данных](https://stepik.org/lesson/297515/step/3?unit=279275);
* [сортировка](https://stepik.org/lesson/297509/step/11?unit=279269).

**Структура и наполнение таблицы trip**

### Введите SQL запрос

SELECT city, COUNT(city) AS Количество

FROM trip

GROUP BY city

ORDER BY Количество DESC

LIMIT 2;

## Задание

Вывести информацию о командировках во все города кроме Москвы и Санкт-Петербурга (фамилии и инициалы сотрудников, город ,  длительность командировки в днях, при этом первый и последний день относится к периоду командировки). Последний столбец назвать **Длительность**. Информацию вывести в упорядоченном по убыванию длительности поездки, а потом по убыванию названий городов (в обратном алфавитном порядке).

#### Немного теории

Для вычитания двух дат используется функция **DATEDIFF(дата\_1, дата\_2)**, результатом которой является количество дней между **дата\_1** и **дата\_2**. Например,

DATEDIFF('2020-04-01', '2020-03-28')=4

DATEDIFF('2020-05-09','2020-05-01')=8

DATEDIFF(date\_last, date\_first)

**Пояснение**

Увеличьте разницу на 1, чтобы включить первый день командировки.

**Текст задания** (чтобы не прокручивать страницу):

Вывести информацию о командировках во все города кроме Москвы и Санкт-Петербурга (фамилии и инициалы сотрудников, город ,  длительность командировки в днях, при этом первый и последний день относится к периоду командировки). Последний столбец назвать **Длительность**. Информацию вывести в упорядоченном по убыванию длительности поездки, а потом по убыванию названий городов (в обратном алфавитном порядке).

**Результат**

+---------------+-------------+--------------+

| name | city | Длительность |

+---------------+-------------+--------------+

| Ильиных Г.Р. | Владивосток | 22 |

| Баранов П.Е. | Новосибирск | 17 |

| Колесов С.П. | Новосибирск | 15 |

| Абрамова К.А. | Владивосток | 14 |

| Лебедев Т.К. | Томск | 12 |

| Абрамова К.А. | Владивосток | 12 |

| Федорова А.Ю. | Новосибирск | 11 |

| Колесов С.П. | Новосибирск | 10 |

| Федорова А.Ю. | Томск | 7 |

| Баранов П.Е. | Воронеж | 7 |

+---------------+-------------+--------------+

**Связанные шаги**

* [выборка столбцов и их именование](https://stepik.org/lesson/297509/step/4?unit=279269);
* условие отбора ([шаг](https://stepik.org/lesson/297509/step/9?unit=279269), [шаг](https://stepik.org/lesson/297509/step/10?unit=279269));
* [сортировка](https://stepik.org/lesson/297509/step/11?unit=279269).

**Структура и наполнение таблицы trip**

### Введите SQL запрос

SELECT name, city,

DATEDIFF(date\_last, date\_first) + 1 AS Длительность

FROM trip

WHERE city NOT IN ('Москва', 'Санкт-Петербург')

ORDER BY Длительность DESC, city DESC;

## Задание

Вывести информацию о командировках сотрудника(ов), которые были самыми короткими по времени. В результат включить столбцы **name**, **city**, **date\_first**, **date\_last**.

**Пояснение**

Используйте вложенный запрос, чтобы найти длительность самой короткой командировки.

**Результат**

+--------------+-----------------+------------+------------+

| name | city | date\_first | date\_last |

+--------------+-----------------+------------+------------+

| Семенов И.В. | Санкт-Петербург | 2020-06-01 | 2020-06-03 |

+--------------+-----------------+------------+------------+

**Связанные шаги**

* [выборка столбцов](https://stepik.org/lesson/297509/step/3?unit=279269);
* [вложенный запрос в WHERE](https://stepik.org/lesson/297514/step/3?unit=279274);
* [вычитание дат](https://stepik.org/lesson/297510/step/6?unit=279270);
* [групповые вычисления по всей таблице](https://stepik.org/lesson/297515/step/6?unit=279275);
* [групповые функции](https://stepik.org/lesson/297515/step/4?unit=279275).

**Структура и наполнение таблицы trip**

### Введите SQL запрос

SELECT name, city, date\_first, date\_last

FROM trip

WHERE DATEDIFF(date\_last, date\_first) =

(SELECT MIN(DATEDIFF(date\_last, date\_first))

FROM trip);

## Задание

Вывести информацию о командировках, начало и конец которых относятся к одному месяцу (год может быть любой). В результат включить столбцы **name**, **city**, **date\_first**, **date\_last**. Строки отсортировать сначала  в алфавитном порядке по названию города, а затем по фамилии сотрудника .

**Немного теории**

Для того, чтобы выделить номер месяца из даты используется функция **MONTH(дата)**.

Например, **MONTH('2020-04-12') = 4**.

Если определяется месяц для  значений столбца **date\_first**, то используется запись **MONTH(date\_first)**

**Результат**

+---------------+-----------------+------------+------------+

| name | city | date\_first | date\_last |

+---------------+-----------------+------------+------------+

| Абрамова К.А. | Владивосток | 2020-01-14 | 2020-01-27 |

| Абрамова К.А. | Владивосток | 2020-07-02 | 2020-07-13 |

| Баранов П.Е. | Воронеж | 2020-07-19 | 2020-07-25 |

| Абрамова К.А. | Москва | 2020-04-06 | 2020-04-14 |

| Баранов П.Е. | Москва | 2020-01-12 | 2020-01-17 |

| Баранов П.Е. | Москва | 2020-02-14 | 2020-02-22 |

| Колесов С.П. | Москва | 2020-02-01 | 2020-02-06 |

| Лебедев Т.К. | Москва | 2020-03-03 | 2020-03-06 |

| Семенов И.В. | Москва | 2020-01-23 | 2020-01-31 |

| Колесов С.П. | Новосибирск | 2020-06-03 | 2020-06-12 |

| Семенов И.В. | Санкт-Петербург | 2020-06-01 | 2020-06-03 |

| Лебедев Т.К. | Томск | 2020-05-20 | 2020-05-31 |

| Федорова А.Ю. | Томск | 2020-06-20 | 2020-06-26 |

+---------------+-----------------+------------+------------+

**Связанные шаги**

* [выборка столбцов](https://stepik.org/lesson/297509/step/3?unit=279269);
* [условие отбора](https://stepik.org/lesson/297509/step/12?unit=279269);
* [сортировка](https://stepik.org/lesson/297509/step/11?unit=279269).

**Структура и наполнение таблицы trip**

### Введите SQL запрос

SELECT name, city, date\_first, date\_last

FROM trip

WHERE MONTH(date\_first) = MONTH(date\_last)

ORDER BY city, name;

## Задание

Вывести название месяца и количество командировок для каждого месяца. Считаем, что командировка относится к некоторому месяцу, если она началась в этом месяце. Информацию вывести сначала в отсортированном по убыванию количества, а потом в алфавитном порядке по названию месяца виде. Название столбцов – **Месяц** и **Количество**.

**Немного теории**

1. Для того, чтобы выделить название месяца из даты используется функция **MONTHNAME(дата)**, которая возвращает название месяца на английском языке для указанной даты. Например, **MONTHNAME('2020-04-12')='April'**.
2. Если группировка осуществляется по вычисляемому столбцу (в данном случае «вычисляется» название месяца), то после GROUP BYможно указать как вычисляемое выражение, так и имя столбца, заданное с помощью AS. Важно отметить, что последний вариант (указать имя столбца)  нарушает стандарт по порядку выполнения запросов, но иногда может встречаться на реальных платформах.

**Результат**

+----------+------------+

| Месяц | Количество |

+----------+------------+

| February | 4 |

| January | 4 |

| June | 3 |

| May | 3 |

| April | 2 |

| July | 2 |

| March | 2 |

+----------+------------+

**Связанные шаги**

* [выборка столбцов и их именование](https://stepik.org/lesson/297509/step/4?unit=279269);
* [вычисления для сгруппированных данных](https://stepik.org/lesson/297515/step/3?unit=279275);
* [сортировка](https://stepik.org/lesson/297509/step/11?unit=279269).

**Структура и наполнение таблицы trip**

### Введите SQL запрос

SELECT MONTHNAME(date\_first) AS Месяц,

COUNT(MONTHNAME(date\_first)) AS Количество

FROM trip

GROUP BY Месяц

ORDER BY Количество DESC, Месяц;

## Задание

Вывести сумму суточных (произведение количества дней командировки и размера суточных) для командировок, первый день которых пришелся на февраль или март 2020 года. Значение суточных для каждой командировки занесено в столбец **per\_diem**. Вывести фамилию и инициалы сотрудника, город, первый день командировки и сумму суточных. Последний столбец назвать **Сумма**. Информацию отсортировать сначала  в алфавитном порядке по фамилиям сотрудников, а затем по убыванию суммы суточных.

**Пояснение**

**1.** В SQL есть функции, которые позволяют выделить часть даты: день(**DAY()**), месяц (**MONTH()**), год(**YEAR()**) . Например:

DAY('2020-02-01') = 1

MONTH('2020-02-01') = 2

YEAR('2020-02-01') = 2020

2. Количество дней командировки вычисляется как разница между датами последнего и первого дня командировки плюс 1.

**Результат**

+---------------+-----------------+------------+---------+

| name | city | date\_first | Сумма |

+---------------+-----------------+------------+---------+

| Абрамова К.А. | Москва | 2020-02-23 | 5600.00 |

| Баранов П.Е. | Москва | 2020-02-14 | 6300.00 |

| Колесов С.П. | Новосибирск | 2020-02-27 | 6750.00 |

| Колесов С.П. | Москва | 2020-02-01 | 4200.00 |

| Лебедев Т.К. | Москва | 2020-03-03 | 2800.00 |

| Семенов И.В. | Санкт-Петербург | 2020-03-29 | 5600.00 |

+---------------+-----------------+------------+---------+

**Связанные шаги**

* [выборка столбцов и их именование](https://stepik.org/lesson/297509/step/4?unit=279269);
* [вычисления в SELECT](https://stepik.org/lesson/297509/step/5?unit=279269);
* [вычитание дат](https://stepik.org/lesson/297510/step/6?unit=279270);
* [выделение номера месяца из даты](https://stepik.org/lesson/297510/step/8?unit=279270);
* [условие отбора](https://stepik.org/lesson/297509/step/12?unit=279269);
* [сортировка](https://stepik.org/lesson/297509/step/11?unit=279269).

**Структура и наполнение таблицы trip**

### Введите SQL запрос

SELECT name, city, date\_first,

(SELECT (DATEDIFF(date\_last, date\_first) + 1) \* per\_diem) AS Сумма

FROM trip

WHERE MONTH(date\_first) = 2 OR MONTH(date\_first) = 3

ORDER BY name, Сумма DESC;

## Задание

Вывести фамилию с инициалами и общую сумму суточных, полученных за все командировки для тех сотрудников, которые были в командировках больше чем 3 раза, в отсортированном по убыванию сумм суточных виде. Последний столбец назвать **Сумма**.

Только для этого задания изменена строка таблицы **trip**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4 | Ильиных Г.Р. | Владивосток | 450 | 2020-01-12 | 2020-03-02 |

**Результат**

+---------------+----------+

| name | Сумма |

+---------------+----------+

| Абрамова К.А. | 29200.00 |

| Баранов П.Е. | 21300.00 |

+---------------+----------+

**Связанные шаги**

* [выборка столбцов](https://stepik.org/lesson/297509/step/3?unit=279269);
* [вычисления в SELECT](https://stepik.org/lesson/297509/step/5?unit=279269);
* [вычисления над сгруппированными данными](https://stepik.org/lesson/297515/step/3?unit=279275);
* [условие отбора в запросах группировки](https://stepik.org/lesson/297515/step/7?unit=279275);
* [вложенные запросы](https://stepik.org/lesson/297514/step/4?unit=279274);
* [сортировка](https://stepik.org/lesson/297509/step/11?unit=279269).

**Структура и наполнение таблицы trip**

### Введите SQL запрос

SELECT name,

SUM((DATEDIFF(date\_last, date\_first) + 1) \* per\_diem) AS Сумма

FROM trip

GROUP BY name

HAVING COUNT(date\_first) > 3

ORDER BY Сумма DESC;

Когда не научился пользоваться HAVING COUNT

SELECT name, SUM((DATEDIFF(date\_last,date\_first)+1)\*per\_diem) AS Сумма  
FROM trip  
WHERE name IN (SELECT IF(COUNT(name)>3,name,0) from trip group by name)  
GROUP BY name  
ORDER BY name

## 1.7 Таблица "Нарушения ПДД", запросы корректировки

## Содержание урока

В этом уроке на каждом шаге используется таблица,  в которой представлена информация о начисленных водителям штрафах за нарушения правил дорожного движения (ПДД). С помощью запросов корректировки необходимо выполнить следующие действия:

* [создать таблицу с информацией о штрафах](https://stepik.org/lesson/305762/step/2?unit=287773) ;
* [заполнить ее](https://stepik.org/lesson/305762/step/3?unit=287773);
* [занести сумму штрафа за каждое новое нарушение ПДД](https://stepik.org/lesson/305762/step/4?unit=287773);
* если водитель на определенной машине совершил повторное нарушение, то сумму его штрафа за данное нарушение нужно увеличить в два раза ([часть 1](https://stepik.org/lesson/305762/step/5?unit=287773), [часть 2](https://stepik.org/lesson/305762/step/6?unit=287773)) ;
* [если водитель оплатил свой штраф в течение 20 дней со дня нарушения, то значение его штрафа уменьшить в два раза](https://stepik.org/lesson/305762/step/7?unit=287773);
* [создать новую таблицу,  в которую включить информацию о всех неоплаченных штрафах](https://stepik.org/lesson/305762/step/8?unit=287773);
* [удалить  информацию о нарушениях, совершенных раньше некоторой даты](https://stepik.org/lesson/305762/step/9?unit=287773).

На [четвертом шаге](https://stepik.org/lesson/305762/step/4?unit=287773) рассматривается временное именование таблиц (алиасы).

## Структура и наполнение таблиц

В таблице **fine** представлена информация о начисленных водителям штрафах за нарушения правил дорожного движения (ПДД) (фамилия водителя, номер машины, описание нарушения, сумма штрафа, дата совершения нарушения и дата оплаты штрафа):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **fine\_id** | **name** | **number\_plate** | **violation** | **sum\_fine** | **date\_violation** | **date\_payment** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(30)** | **VARCHAR(6)** | **VARCHAR(50)** | **DECIMAL(8,2)** | **DATЕ** | **DATE** |
| 1 | Баранов П.Е. | Р523ВТ | Превышение скорости (от 40 до 60) | 500.00 | 2020-01-12 | 2020-01-17 |
| 2 | Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал | 1000.00 | 2020-01-14 | 2020-02-27 |
| 3 | Яковлев Г.Р. | Т330ТТ | Превышение скорости (от 20 до 40) | 500.00 | 2020-01-23 | 2020-02-23 |
| 4 | Яковлев Г.Р. | М701АА | Превышение скорости (от 20 до 40) |  | 2020-01-12 |  |
| 5 | Колесов С.П. | К892АХ | Превышение скорости (от 20 до 40) |  | 2020-02-01 |  |
| 6 | Баранов П.Е. | Р523ВТ | Превышение скорости (от 40 до 60) |  | 2020-02-14 |  |
| 7 | Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал |  | 2020-02-23 |  |
| 8 | Яковлев Г.Р. | Т330ТТ | Проезд на запрещающий сигнал |  | 2020-03-03 |  |

В таблицу  **traffic\_violation** занесены нарушения ПДД и соответствующие штрафы (в рублях):

|  |  |  |
| --- | --- | --- |
| **violation\_id** | **violation** | **sum\_fine** |
| **INT PRIMARY KEY AUTO\_INCREMENT** | **VARCHAR(50)** | **DECIMAL(8,2)** |
| 1 | Превышение скорости (от 20 до 40) | 500.00 |
| 2 | Превышение скорости (от 40 до 60) | 1000.00 |
| 3 | Проезд на запрещающий сигнал | 1000.00 |

## Задание

Создать таблицу **fine** следующей структуры:

|  |  |
| --- | --- |
| **Поле** | **Описание** |
| **fine\_id** | ключевой столбец целого типа с автоматическим увеличением значения ключа на 1 |
| **name** | строка длиной 30 |
| **number\_plate** | строка длиной 6 |
| **violation** | строка длиной 50 |
| **sum\_fine** | вещественное число, максимальная длина 8, количество знаков после запятой 2 |
| **date\_violation** | дата |
| **date\_payment** | дата |

**Результат** **Связанные шаги**

### Введите SQL запрос

CREATE TABLE fine (

fine\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(30),

number\_plate VARCHAR(6),

violation VARCHAR(50),

sum\_fine DECIMAL(8,2),

date\_violation DATE,

date\_payment DATE

);

**Задание**

В таблицу **fine** первые 5 строк уже занесены. Добавить в таблицу записи с ключевыми значениями 6, 7, 8.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **fine\_id** | **name** | **number\_plate** | **violation** | **sum\_fine** | **date\_violation** | **date\_payment** |
| 1 | Баранов П.Е. | Р523ВТ | Превышение скорости (от 40 до 60) | 500.00 | 2020-01-12 | 2020-01-17 |
| 2 | Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал | 1000.00 | 2020-01-14 | 2020-02-27 |
| 3 | Яковлев Г.Р. | Т330ТТ | Превышение скорости (от 20 до 40) | 500.00 | 2020-01-23 | 2020-02-23 |
| 4 | Яковлев Г.Р. | М701АА | Превышение скорости (от 20 до 40) |  | 2020-01-12 |  |
| 5 | Колесов С.П. | К892АХ | Превышение скорости (от 20 до 40) |  | 2020-02-01 |  |
| 6 | Баранов П.Е. | Р523ВТ | Превышение скорости (от 40 до 60) |  | 2020-02-14 |  |
| 7 | Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал |  | 2020-02-23 |  |
| 8 | Яковлев Г.Р. | Т330ТТ | Проезд на запрещающий сигнал |  | 2020-03-03 |  |

**Пояснение**

1. Между формулировкой нарушения и открывающей скобкой ПРОБЕЛА НЕТ. Например, ***Превышение скорости(от 40 до 60)***.
2. Для занесения пустых значений в поля используется оператор **Null**.
3. Номера машин записывать РУССКИМИ буквами.

**Результат** **Важно**

**Текст задания** (чтобы не прокручивать страницу):

 В таблицу **fine** первые 5 строк уже занесены. Добавить в таблицу записи с ключевыми значениями 6, 7, 8.

**Введите SQL запрос**

INSERT INTO fine (name, number\_plate, violation, sum\_fine, date\_violation, date\_payment) VALUES

('Баранов П.Е.', 'Р523ВТ', 'Превышение скорости(от 40 до 60)', NULL, '2020-02-14', NULL),

('Абрамова К.А.', 'О111АВ', 'Проезд на запрещающий сигнал', NULL, '2020-02-23', NULL),

('Яковлев Г.Р.', 'Т330ТТ', 'Проезд на запрещающий сигнал', NULL, '2020-03-03', NULL);

## Использование временного имени таблицы (алиаса)

Теоретический материал для этого шага подготовлен [Михаилом Захаров](https://stepik.org/users/15955624)ым . Большое ему спасибо!

Чтобы не писать название таблицы каждый раз, удобно использовать алиасы.

Алиас, это псевдоним, который мы присваивали столбцам после ключевого слова **AS**([шаг](https://stepik.org/lesson/297509/step/4?unit=279269)).  Алиасы так же можно использовать и для таблиц. Это становится актуальным, при увеличении числа используемых таблиц, их иногда может быть и 5 и 10 и более. Псевдонимы помогают сделать запрос чище и читабельнее.

Для присваивания псевдонима существует 2 варианта:

* с использованием ключевого слова **AS**

FROM fine AS f, traffic\_violation AS tv

* а так же и без него

FROM fine f, traffic\_violation tv

После присвоения таблице алиаса, он используется во всех разделах запроса, в котором алиас задан. Например:

WHERE f.violation = tv.violation

**Пример**

Для тех, кто уже оплатил штраф, вывести информацию о том, изменялась ли стандартная сумма штрафа.

Запрос:

SELECT f.name, f.number\_plate, f.violation,

if(

f.sum\_fine = tv.sum\_fine, "Стандартная сумма штрафа",

if(

f.sum\_fine < tv.sum\_fine, "Уменьшенная сумма штрафа", "Увеличенная сумма штрафа"

)

) AS description

FROM fine f, traffic\_violation tv

WHERE tv.violation = f.violation and f.sum\_fine IS NOT Null;

Результат:

+---------------+--------------+----------------------------------+--------------------------+

| name | number\_plate | violation | description |

+---------------+--------------+----------------------------------+--------------------------+

| Баранов П.Е. | Р523ВТ | Превышение скорости(от 40 до 60) | Уменьшенная сумма штрафа |

| Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал | Стандартная сумма штрафа |

| Яковлев Г.Р. | Т330ТТ | Превышение скорости(от 20 до 40) | Стандартная сумма штрафа |

+---------------+--------------+----------------------------------+--------------------------+

## Задание

Занести в таблицу **fine** суммы штрафов, которые должен оплатить водитель, в соответствии с данными из таблицы **traffic\_violation**. При этом суммы заносить только в пустые поля столбца **sum\_fine**.

Таблица **traffic\_violation**создана и заполнена.

**Важно!** Сравнение значения столбца с пустым значением осуществляется с помощью оператора **IS NULL**.

**Пояснение**

1. После ключевого слова UPDATE кроме обновляемой таблицы**fine** укажите таблицу **traffic\_violation,**для того чтобы запрос видел таблицы источники.  Сначала перечисляем **все источники,** потом выполняем необходимые действия.

2.Обновляйте только те записи таблицы**fine**, у которых значение столбца **violation**совпадает со значением соответствующего столбца таблицы **traffic\_violation**, а также значение столбца **sum\_fine** пусто.

**Результат**

Примечание: для сокращения записи ключевой столбец не показан.

Affected rows: 5

Query result:

+---------------+--------+------------------------------+----------+----------------+--------------+

| name | number | violation | sum\_fine | date\_violation | date\_payment |

| | \_plate | | | | |

+---------------+--------+------------------------------+----------+----------------+--------------+

| Баранов П.Е. | Р523ВТ | Превышение скорости(от 40... | 500.00 | 2020-01-12 | 2020-01-17 |

| Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал | 1000.00 | 2020-01-14 | 2020-02-27 |

| Яковлев Г.Р. | Т330ТТ | Превышение скорости(от 20... | 500.00 | 2020-01-23 | 2020-02-23 |

| Яковлев Г.Р. | М701АА | Превышение скорости(от 20... | 500.00 | 2020-01-12 | NULL |

| Колесов С.П. | К892АХ | Превышение скорости(от 20... | 500.00 | 2020-02-01 | NULL |

| Баранов П.Е. | Р523ВТ | Превышение скорости(от 40... | 1000.00 | 2020-02-14 | NULL |

| Абрамова К.А. | О111АВ | Проезд на запрещающий сигн...| 1000.00 | 2020-02-23 | NULL |

| Яковлев Г.Р. | Т330ТТ | Проезд на запрещающий сигн...| 1000.00 | 2020-03-03 | NULL |

+---------------+--------+------------------------------+----------+----------------+--------------+

**Связанные шаги** **Структура и наполнение исходной таблицы fine (без ключевого столбца) и таблицыtraffic\_violation**

### Введите SQL запрос

UPDATE fine AS f, traffic\_violation AS tv

SET f.sum\_fine = tv.sum\_fine

WHERE f.sum\_fine IS NULL AND

f.violation = tv.violation;

SELECT \* FROM fine;

## Группировка данных по нескольким столбцам

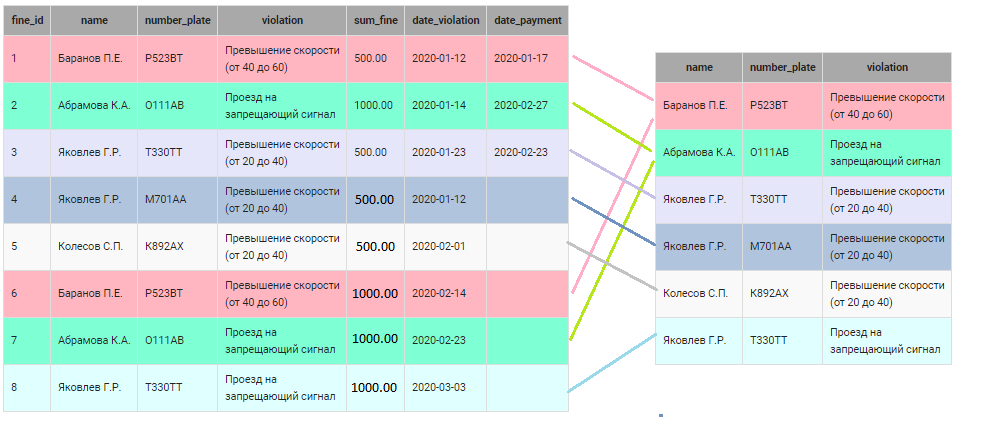
В разделе GROUP BY можно указывать несколько столбцов, разделяя их запятыми. Тогда к одной группе будут относиться записи, у которых значения столбцов, входящих в группу, равны. Рассмотрим группировку по нескольким столбцам на примере следующего запроса:

SELECT name, number\_plate, violation, count(\*)

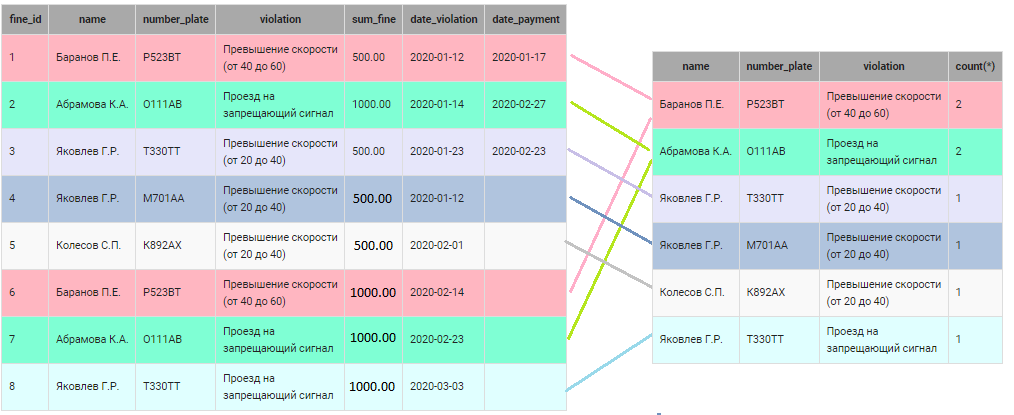
FROM fine

GROUP BY name, number\_plate, violation;

1. Сначала записи таблицы  **fine** разделяются на группы. В каждую группу включаются строки, у которых равны значения в столбцах **name**, **number\_plate** и **violation**  соответственно. Получается 6 групп.



2. Затем вычисляется функция **count(\*)**, которая определяет количество записей в каждой группе. Получается, что к первым двум группам относятся по две записи, ко всем остальным - по одной.



**Важно!** В разделе **GROUP BY** нужно перечислять все НЕАГРЕГИРОВАННЫЕ столбцы (к которым не применяются групповые функции) из **SELECT**.

## Задание

Вывести фамилию, номер машины и нарушение только для тех водителей, которые на одной машине нарушили одно и то же правило   два и более раз. При этом учитывать все нарушения, независимо от того оплачены они или нет. Информацию отсортировать в алфавитном порядке, сначала по фамилии водителя, потом по номеру машины и, наконец, по нарушению.

**Пояснение** **Результат** **Связанные шаги** **Структура и наполнение таблицы fine (без ключевого столбца) перед выполнением этого шага**

### Введите SQL запрос

SELECT name, number\_plate, violation

FROM fine

GROUP BY name, number\_plate, violation

HAVING COUNT(\*) >= 2

ORDER BY name, number\_plate, violation;

## Задание

В таблице **fine** увеличить в два раза сумму неоплаченных штрафов для отобранных на предыдущем шаге записей.

**Пояснение !!! если не получается запрос или валидатор выдает ошибки, раскройте это пояснение!!!**

1. Для всех нарушений, по которым штраф еще не оплачен, (тех, у которых **date\_payment** имеет пустое значение **Null**), необходимо проверить, является ли данное нарушение для водителя и машины повторным, если да –  увеличить штраф в два раза.
2. Если водитель совершил нарушение на другой машине, ему увеличивать штраф не нужно.
3. Если несколько повторных нарушений не оплачены, то штраф увеличить для всех.
4. Этот запрос реализован на предыдущем [шаге](https://stepik.org/lesson/305762/step/5?unit=287773).

При реализации можно использовать вложенный запрос как отдельную таблицу, записанную после ключевого слова UPDATE, при этом вложенному запросу необходимо присвоить имя, например **query\_in**:

UPDATE fine,

(

SELECT ...

) query\_in

SET ...

WHERE указать, что совпадают нарушение, фамилия водителя и номер машины в таблицах fine и вложенном запросе query\_in соответственно, а также дата оплаты в таблице fine пуста

Другим способом решения является использование двух запросов: сначала создать временную таблицу, например **query\_in**, в которую включить информацию о тех штрафах, сумму которых нужно увеличить в два раза, а затем уже обновлять информацию в таблице **fine**:

CREATE TABLE query\_in ...;

UPDATE fine, query\_in

SET ...

WHERE ...;

После ключевого слова WHERE  указывается условие, при котором нужно обновлять данные. В нашем случае  данные обновляются, если и фамилия, и государственный номер, и нарушение совпадают в таблице **fine** и в результирующей таблице запроса **query\_in**. Например, для связи по фамилии используется запись **fine.name = query\_in.name**. Также в условии нужно учесть, что данные обновляются только для тех записей, у которых в столбце **date\_payment** пусто.

**Важно!** Если в запросе используется несколько таблиц или запросов, включающих одинаковые поля, то применяется полное имя столбца, включающего название таблицы через символ «**.**». Например, **fine.name**  и  **query\_in.name**.

**Результат** Affected rows: 2

Query result:

+---------------+--------------+----------------------------------+----------+------------------+--------------+

| name | number\_plate | violation | sum\_fine | date\_violation | date\_payment |

+---------------+--------------+----------------------------------+----------+------------------+--------------+

| Баранов П.Е. | Р523ВТ | Превышение скорости(от 40 до 60) | 500.00 | 2020-01-12 | 2020-01-17 |

| Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал | 1000.00 | 2020-01-14 | 2020-02-27 |

| Яковлев Г.Р. | Т330ТТ | Превышение скорости(от 20 до 40) | 500.00 | 2020-01-23 | 2020-02-23 |

| Яковлев Г.Р. | М701АА | Превышение скорости(от 20 до 40) | 500.00 | 2020-01-12 | NULL |

| Колесов С.П. | К892АХ | Превышение скорости(от 20 до 40) | 500.00 | 2020-02-01 | NULL |

| Баранов П.Е. | Р523ВТ | Превышение скорости(от 40 до 60) | 2000.00 | 2020-02-14 | NULL |

| Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал | 2000.00 | 2020-02-23 | NULL |

| Яковлев Г.Р. | Т330ТТ | Проезд на запрещающий сигнал | 1000.00 | 2020-03-03 | NULL |

+---------------+--------------+----------------------------------+----------+------------------+--------------+

Введите SQL запрос

UPDATE fine,

(

SELECT name, number\_plate, violation

FROM fine

GROUP BY name, number\_plate, violation

HAVING COUNT(\*) >= 2

ORDER BY name, number\_plate, violation

) query\_in

SET sum\_fine = sum\_fine \* 2

WHERE date\_payment IS NULL AND

fine.name=query\_in.name AND

fine.number\_plate=query\_in.number\_plate

AND fine.violation=query\_in.violation;

**Задание**

Водители оплачивают свои штрафы. В таблице **payment** занесены даты их оплаты:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **payment\_id** | **name** | **number\_plate** | **violation** | **date\_violation** | **date\_payment** |
| 1 | Яковлев Г.Р. | М701АА | Превышение скорости (от 20 до 40) | 2020-01-12 | 2020-01-22 |
| 2 | Баранов П.Е. | Р523ВТ | Превышение скорости (от 40 до 60) | 2020-02-14 | 2020-03-06 |
| 3 | Яковлев Г.Р. | Т330ТТ | Проезд на запрещающий сигнал | 2020-03-03 | 2020-03-23 |

Необходимо:

* в таблицу **fine**занести дату оплаты соответствующего штрафа из таблицы **payment;**
* уменьшить начисленный штраф в таблице **fine** в два раза  (только для тех штрафов, информация о которых занесена в таблицу **payment**) , если оплата произведена не позднее 20 дней со дня нарушения.

**Пояснение к заданию**

1. Для уменьшения суммы штрафа в два раза в зависимости от условия можно  использовать функцию [**if()**](https://stepik.org/lesson/297509/step/7?unit=279269).  Синтаксис раздела SET при использовании функции **if()** следующий:

SET столбец = IF(условие, выражение\_1, выражение\_2)

Выполняется этот оператор так:

* сначала вычисляется **условие**;
* если условие ИСТИНА, то вычисляется **выражение\_1**, в противном случае (если условие ЛОЖНО) вычисляется **выражение\_2**;
* в столбец заносится результат выполнения функции (либо значение **выражения\_1**, либо значение **выражения\_2** в зависимости от условия).

Например, чтобы обнулить штрафы, меньшие или равные 500 рублей, а остальные оставить без изменения, используется запрос:

UPDATE fine

SET sum\_fine = IF(sum\_fine <= 500, 0, sum\_fine)

2. Количество дней между датой нарушения и датой оплаты считается по формуле:

**количество\_дней = дата\_оплаты - дата\_нарушения**

**Пояснение к решению  !!! раскройте пояснение, если не получается решить или ошибка в запросе !!!**

Шаблон запроса:

UPDATE

fine, payment

SET

fine.date\_payment = дата оплаты из payment,

fine.sum\_fine = сравнить разницу между датой нарушения и датой оплаты (из payment)

и при необходимости уменьшить размер штрафа (использовать IF)

WHERE

указать условие, что совпадают и номера машин, и нарушения, и

фамилия водителя в таблицах fine и payment, а также, что

дата оплаты в fine должна быть пуста

**Результат**

Affected rows: 3

Query result:

+---------------+--------+----------------------------------+----------+---------------------------+

| name | number\_| violation | sum\_fine | date\_ | date\_payment |

| | plate | | | violation | |

+---------------+--------+----------------------------------+----------+------------+--------------+

| Баранов П.Е. | Р523ВТ | Превышение скорости(от 40 до 60) | 500.00 | 2020-01-12 | 2020-01-17 |

| Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал | 1000.00 | 2020-01-14 | 2020-02-27 |

| Яковлев Г.Р. | Т330ТТ | Превышение скорости(от 20 до 40) | 500.00 | 2020-01-23 | 2020-02-23 |

| Яковлев Г.Р. | М701АА | Превышение скорости(от 20 до 40) | 250.00 | 2020-01-12 | 2020-01-22 |

| Колесов С.П. | К892АХ | Превышение скорости(от 20 до 40) | 500.00 | 2020-02-01 | NULL |

| Баранов П.Е. | Р523ВТ | Превышение скорости(от 40 до 60) | 2000.00 | 2020-02-14 | 2020-03-06 |

| Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал | 2000.00 | 2020-02-23 | NULL |

| Яковлев Г.Р. | Т330ТТ | Проезд на запрещающий сигнал | 500.00 | 2020-03-03 | 2020-03-23 |

+---------------+--------+----------------------------------+----------+------------+--------------+

**Связанные шаги** **Структура и наполнение таблицы fine (без ключевого столбца) перед выполнением этого шага**

**Введите SQL запрос**

UPDATE fine, payment

SET fine.date\_payment = payment.date\_payment,

fine.sum\_fine = IF((fine.date\_payment - fine.date\_violation) <= 20, fine.sum\_fine \* 0.5, fine.sum\_fine)

WHERE fine.name = payment.name AND

fine.number\_plate = payment.number\_plate AND

fine.violation = payment.violation AND

fine.date\_payment IS NULL;

SELECT name, number\_plate, violation, sum\_fine, date\_payment from fine;

## Задание

Создать новую таблицу **back\_payment**, куда внести информацию о неоплаченных штрафах (Фамилию и инициалы водителя, номер машины, нарушение, сумму штрафа  и  дату нарушения) из таблицы **fine**.

**Пояснение**

Для неоплаченных штрафов столбец **date\_payment** имеет пустое значение.

**Важно.** На этом шаге необходимо создать таблицу на основе запроса! Не нужно одним запросом создавать таблицу, а вторым в нее добавлять строки.

**Результат**

Affected rows: 2

Query result:

+---------------+--------------+----------------------------------+----------+------------------+

| name | number\_plate | violation | sum\_fine | date\_violation |

+---------------+--------------+----------------------------------+----------+------------------+

| Колесов С.П. | К892АХ | Превышение скорости(от 20 до 40) | 500.00 | 2020-02-01 |

| Абрамова К.А. | О111АВ | Проезд на запрещающий сигнал | 2000.00 | 2020-02-23 |

+---------------+--------------+----------------------------------+----------+------------------+

**Связанные шаги** **Структура и наполнение таблицы fine (без ключевого столбца) перед выполнением этого шага**

### Введите SQL запрос

CREATE TABLE back\_payment AS

SELECT name, number\_plate, violation, sum\_fine, date\_violation

FROM fine

WHERE date\_payment IS NULL;

SELECT \* from back\_payment;

## Задание

Удалить из таблицы **fine** информацию о нарушениях, совершенных раньше 1 февраля 2020 года.

**Результат**

Affected rows: 4

Query result (это выборка из таблицы fine после удаления записей):

+---------------+--------+------------------------------+----------+----------------+--------------+

| name | number | violation | sum\_fine | date\_violation | date\_payment |

| | \_plate | | | | |

+---------------+--------+------------------------------+----------+----------------+--------------+

| Колесов С.П. | К892АХ | Превышение скорости(от 20... | 500.00 | 2020-02-01 | NULL |

| Баранов П.Е. | Р523ВТ | Превышение скорости(от 40... | 2000.00 | 2020-02-14 | 2020-03-05 |

| Абрамова К.А. | О111АВ | Проезд на запрещающий сигн...| 2000.00 | 2020-02-23 | NULL |

| Яковлев Г.Р. | Т330ТТ | Проезд на запрещающий сигн...| 500.00 | 2020-03-03 | 2020-03-22 |

+---------------+--------+------------------------------+----------+----------------+--------------+

**Связанные шаги** **Структура и наполнение таблицы fine (без ключевого столбца) перед выполнением этого шага**

### Введите SQL запрос

DELETE

FROM fine

WHERE date\_violation < '2020-02-01';

SELECT name, number\_plate, violation, sum\_fine, date\_violation FROM fine;

## 2.1 Связи между таблицами

## Содержание урока

Средствами SQL запросов можно выбирать и обрабатывать данные не только из одной таблицы, но из нескольких связанных таблиц. В данном уроке мы рассмотрим способы соединения таблиц:

* [связь между таблицами «один ко многим»](https://stepik.org/lesson/308885/step/2?unit=291011);
* [связь между таблицами «многие ко многим»](https://stepik.org/lesson/308885/step/3?unit=291011);
* [создание таблицы с внешними ключами](https://stepik.org/lesson/308885/step/8?unit=291011);
* [действия при удалении записи главной таблицы](https://stepik.org/lesson/308885/step/9?unit=291011);
* [заполнение таблицы с внешними ключами](https://stepik.org/lesson/308885/step/10?unit=291011);
* [добавление данных в таблицу с внешними ключами](https://stepik.org/lesson/308885/step/11?unit=291011).

## Связь «один ко многим»

Рассмотрим таблицу **book**(в ней столбец **author** переименован в **name\_author**):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **book\_id** | **title** | **name\_author** | **price** | **amount** |
| 1 | Мастер и Маргарита | Булгаков М.А. | 670.99 | 3 |
| 2 | Белая гвардия | Булгаков М.А. | 540.50 | 5 |
| 3 | Идиот | Достоевский Ф.М. | 460.00 | 10 |
| 4 | Братья Карамазовы | Достоевский Ф.М. | 799.01 | 2 |
| 5 | Стихотворения и поэмы | Есенин С.А. | 650.00 | 15 |

В этой таблице фамилии авторов повторяются для нескольких книг. А что, если придется вместо инициалов для каждого автора хранить его полное имя и отчество? Тогда, если в таблице содержится информация о 50 книгах Достоевского, придется 50 раз исправлять «Ф.М.» на «Федор Михайлович». При этом, если в некоторых записях использовать «Фёдор Михайлович» (c буквой ё), то мы вообще получим двух разных авторов...

Чтобы устранить эту проблему в реляционных базах данных создается новая таблица **author**,  в которой перечисляются все различные авторы, а затем эта таблица связывается с таблицей **book**. При этом такая связь называется «**один ко многим**», таблица **author** называется главной, таблица **book** – связанной или подчиненной.

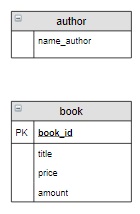
Связь «**один ко многим**» имеет место, когда одной записи главной таблицы соответствует несколько записей связанной таблицы, а каждой записи связанной таблицы соответствует только одна запись главной таблицы. Обозначается это так:



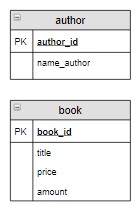
**Этапы реализации связи** «**один ко многим**» на следующем примере:

Один автор может написать несколько книг, а одна книга написана только одним автором. Для каждой книги известны ее количество и цена.

1. Создать таблицу **author**,  в которую включить уникальных авторов книг, хранящихся на складе:



2. Обе таблицы должны содержать первичный ключ, в таблице  **book** он уже есть,  в таблицу **author** добавим ключ **author\_id**:



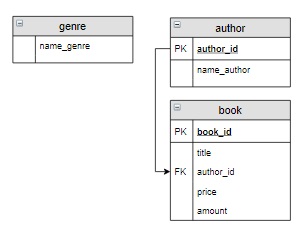
3. Включить в таблицу **book** связанный столбец (внешний ключ, **FOREIGN KEY**), соответствующий по имени и типу ключевому столбцу главной таблицы (в нашем случае это столбец **author\_id**). Для наглядности связь на схеме обозначается стрелкой от ключевого столбца главной таблицы к внешнему ключу связной таблицы:

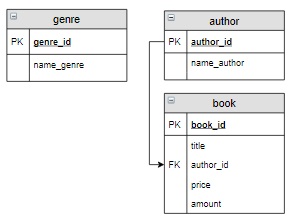


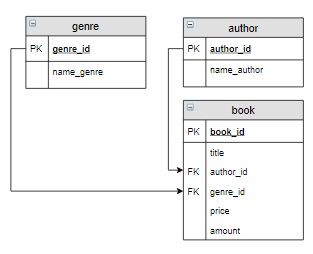
## Задание

Добавить новую характеристику книги – ее жанр, если считать, что каждая книга относится к одному жанру, то есть между ними определена связь «**один ко многим**». Расположите в правильном порядке этапы связывания таблицы с жанрами (**genre**) и таблицы **book**.

### Расположите элементы списка в правильном порядке





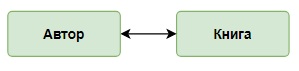


## Связь «многие ко многим»

На предыдущем шаге мы реализовали связь «**один ко многим**» для книг и авторов. Она означает, что каждый автор написал несколько книг, но каждую книгу написал только один автор. На самом деле, это не совсем верное утверждение. Например, книга «12 стульев» написана двумя авторами Ильфом И.А. и Петровым Е.П. С другой стороны, эти авторы написали и другие книги, например «Золотой теленок».

Для соединения таких таблиц используется связь «**многие ко многим**».

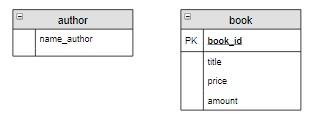
Связь «**многие ко многим**» имеет место когда каждой записи одной таблицы соответствует несколько записей во второй, и наоборот, каждой записи второй таблицы соответствует несколько записей в первой. Обозначается это так:



**Этапы реализации связи** «**многие ко многим**»  на следующем примере:

Один автор может написать несколько книг, а одна книга может быть написана несколькими авторами. Для каждой книги известны ее количество и цена.

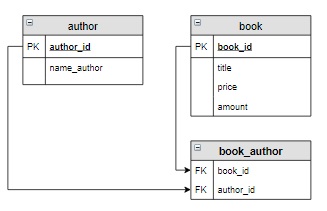
1. Создать таблицу **author**,  в которую включить уникальных авторов книг, хранящихся на складе:



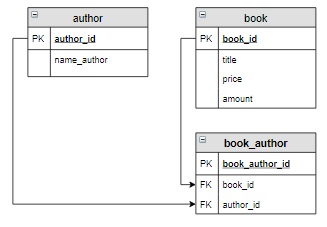
2. В обеих таблицах необходимо определить первичный ключ, в нашем случае в таблице **book** он уже есть, поэтому достаточно включить первичный ключ **author\_id**в таблицу **author**:



3. Создать новую таблицу-связку, состоящую из двух столбцов, соответствующих по имени и типу ключевым столбцам исходных таблиц. Каждый из этих столбцов является внешним ключом (**FOREIGN KEY**)  и связан с ключевым столбцом каждой таблицы. Для наглядности связи на схеме обозначаются стрелкой от ключевого столбца исходной таблицы к внешнему ключу связной таблицы.



4. Дальше необходимо определиться с первичным ключом таблицы-связки. Можно сделать два ключевых столбца, тогда все записи в этой таблице должны быть уникальными, то есть не повторяться. Для связи автор-книга этот вариант подходит. Но в некоторых случаях записи в таблице-связке могут повторяться, например, если мы будем продавать книги покупателям (один человек может купить несколько книг, а одну и ту же книгу могут купить несколько человек). Тогда в таблицу-связку включают дополнительные столбцы для идентификации записей, например, дату продажи,  также в таблицу-связку добавляют первичный ключ. Мы воспользуемся вторым способом:

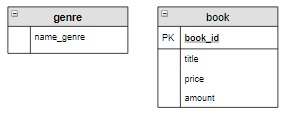


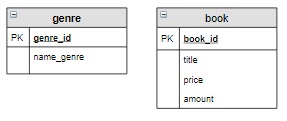
## Задание

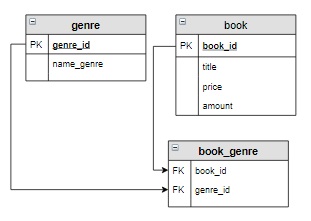
Добавить новую характеристику книги – ее жанр, если считать, что каждая книга может относиться к нескольким жанрам, а каждый жанр включает несколько книг, то есть между ними определена связь «**многие ко многим**».

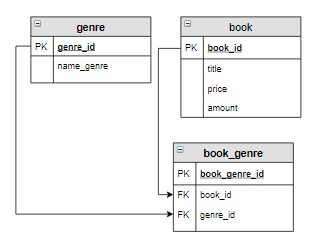
Расположите в правильном порядке этапы связывания таблиц **genre** и **book**.

### Расположите элементы списка в правильном порядке









## Задание

Выберите тип связи, который подходит для описания пар информационных объектов.

### Заполните пропуски

В **группе** учатся **студенты**, каждый студент может учиться только в одной группе:

Студент <--- группа

**Студенты** изучают несколько **дисциплин**, одну и ту же дисциплину могут изучать несколько студентов:  
Студент <---> дисциплина

В библиотеке **студент** может взять несколько **книг**, одну и ту же книгу могут взять несколько студентов (в разное время):

Студент <---> книга  
  
Каждый **студент** родился в определенном **городе**, в одном городе родились несколько студентов:

Студент <--- город  
  
Каждый **город** относится к определенной **стране**, в каждой стране есть несколько городов:

страна ---> город

## Задание

Дана таблица **trip**. Выберите одну или несколько схем, которые позволяют правильно представить информацию из этой таблицы в виде нескольких связанных таблиц.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **trip\_id** | **name** | **city** | **per\_diem** | **date\_first** | **date\_last** |
| 1 | Баранов П.Е. | Москва | 700 | 2020-01-12 | 2020-01-17 |
| 2 | Абрамова К.А. | Владивосток | 450 | 2020-01-14 | 2020-01-27 |
| 3 | Семенов И.В. | Москва | 700 | 2020-01-23 | 2020-01-31 |
| 4 | Семенов И.В. | Владивосток | 450 | 2020-02-12 | 2020-02-22 |

**Пояснение**

Для решения этой задачи необходимо:

1. Проанализировать информацию и выделить повторяющиеся данные в отдельные таблицы. Это будут Сотрудники и Города.

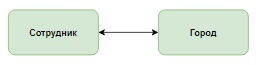
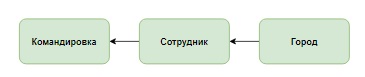
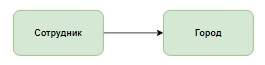
2. Определить тип связи между таблицами (многие ко многим или один ко многим), который позволит описать командировки сотрудников - выбрать верный вариант из предложенных. При этом связь один ко многим обозначается стрелкой (например,  для Дома и Квартиры определена связь один ко многим, которая обозначается как **Дом --> Квартира)**.

3. Определить, как данная связь реализуется в реляционной модели - выбрать верный вариант.

**Пояснение от пользователя @Дмитрий\_Ефремов**

Здесь нужно выбрать не полностью готовую схему БД, а все адекватные связи, которые могли бы быть между этими таблицами (среди них 1-полная схема из 3 таблиц). Обращайте внимание на направление стрелочек.

### Выберите все подходящие ответы из списка



## ****Перекрестное соединение CROSS JOIN****

Оператор перекрёстного соединения, или декартова произведения CROSS JOIN (в запросе вместо ключевых слов можно поставить запятую между таблицами) соединяет две таблицы. Порядок таблиц для оператора неважен, поскольку оператор является симметричным. Его структура:

SELECT

...

FROM

таблица\_1 CROSS JOIN таблица\_2

...

или

SELECT

...

FROM

таблица\_1, таблица\_2

...

Результат запроса формируется так: каждая строка одной таблицы соединяется с каждой строкой другой таблицы, формируя  в результате все возможные сочетания строк двух таблиц.

Например, запрос:

SELECT name\_author, name\_genre

FROM

author, genre;

каждому автору из таблицы **author** поставит в соответствие все возможные жанры из таблицы **genre**:

+------------------+-------------+

| name\_author | name\_genre |

+------------------+-------------+

| Булгаков М.А. | Роман |

| Булгаков М.А. | Поэзия |

| Булгаков М.А. | Приключения |

| Достоевский Ф.М. | Роман |

| Достоевский Ф.М. | Поэзия |

| Достоевский Ф.М. | Приключения |

| Есенин С.А. | Роман |

| Есенин С.А. | Поэзия |

| Есенин С.А. | Приключения |

| Пастернак Б.Л. | Роман |

| Пастернак Б.Л. | Поэзия |

| Пастернак Б.Л. | Приключения |

| Лермонтов М.Ю. | Роман |

| Лермонтов М.Ю. | Поэзия |

| Лермонтов М.Ю. | Приключения |

+------------------+-------------+

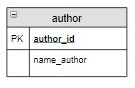
## ****Задание****

Есть список городов, хранящийся в таблице **city**:

|  |  |
| --- | --- |
| **city\_id** | **name\_city** |
| 1 | Москва |
| 2 | Санкт-Петербург |
| 3 | Владивосток |

Необходимо в каждом городе провести выставку книг каждого автора в течение 2020 года. Дату проведения выставки выбрать случайным образом. Создать запрос, который выведет город, автора и дату проведения выставки. Последний столбец назвать **Дата**. Информацию вывести, отсортировав сначала в алфавитном порядке по названиям городов, а потом по убыванию дат проведения выставок.

**Структура таблицы:**



**Пояснение**

1. Для генерации случайной даты можно к первому числу года ('2020-01-01') прибавить целое случайное число в интервале от 0 до 365.

Генерации случайных чисел в интервале от 0 до 1 (не включительно) осуществляется с помощью функции RAND(). Если эту функцию умножить на 365, то она будет генерировать вещественные числа от 0 до 365 (не включительно). Осталось только отбросить дробную часть. Это можно сделать с помощью функции FLOOR(), которая возвращает наибольшее целое число, меньшее или равное указанному числовому значению. Таким образом, случайное число от 0 до 365 можно получить с помощью выражения:

FLOOR(RAND() \* 365)

**Важно!** Даты должны быть за 2020 год, первое число года - 1 января 2020 года.

2. Для сложения  даты с числом используется функция:

DATE\_ADD(дата, INTERVAL число единица\_измерения),

где

единица\_измерения (использовать прописные буквы) – это день (DAY), месяц(MONTH), неделя(WEEK) и пр.,

число – целое число,

дата – значение даты или даты и времени.

Функция к **дате**  прибавляет указанное **число**, выраженное в днях, месяцах и пр. , в зависимости от заданного интервала, и возвращает новую дату.

Например:

DATE\_ADD('2020-02-02', INTERVAL 45 DAY) возвращает 18 марта 2020 года

DATE\_ADD('2020-02-02', INTERVAL 6 MONTH) возвращает 2 августа 2020 года

***Текст задания (чтобы не прокручивать страницу):***

 Есть список городов, хранящийся в таблице **city**:

|  |  |
| --- | --- |
| **city\_id** | **name\_city** |
| 1 | Москва |
| 2 | Санкт-Петербург |
| 3 | Владивосток |

Необходимо в каждом городе провести выставку книг каждого автора в течение 2020 года. Дату проведения выставки выбрать случайным образом. Создать запрос, который выведет город, автора и дату проведения выставки. Последний столбец назвать **Дата**. Информацию вывести, отсортировав сначала в алфавитном порядке по названиям городов, а потом по убыванию дат проведения выставок.

**Результат**

Примечание: даты при каждом запуске получаются разными, и не должны совпадать с приведенными значениями.

+-----------------+------------------+------------+

| name\_city | name\_author | Дата |

+-----------------+------------------+------------+

| Владивосток | Достоевский Ф.М. | 2020-12-04 |

| Владивосток | Лермонтов М.Ю. | 2020-10-21 |

| Владивосток | Пастернак Б.Л. | 2020-08-23 |

| Владивосток | Есенин С.А. | 2020-08-14 |

| Владивосток | Булгаков М.А. | 2020-01-08 |

| Москва | Лермонтов М.Ю. | 2020-09-30 |

| Москва | Достоевский Ф.М. | 2020-07-21 |

| Москва | Есенин С.А. | 2020-06-23 |

| Москва | Булгаков М.А. | 2020-05-28 |

| Москва | Пастернак Б.Л. | 2020-04-08 |

| Санкт-Петербург | Булгаков М.А. | 2020-11-05 |

| Санкт-Петербург | Лермонтов М.Ю. | 2020-10-22 |

| Санкт-Петербург | Достоевский Ф.М. | 2020-09-19 |

| Санкт-Петербург | Есенин С.А. | 2020-08-11 |

| Санкт-Петербург | Пастернак Б.Л. | 2020-06-28 |

+-----------------+------------------+------------+

**Структура и наполнение таблиц**

### Введите SQL запрос

SELECT name\_city, name\_author, DATE\_ADD('2020-01-01', INTERVAL FLOOR(RAND() \* 365) DAY) AS Дата

FROM

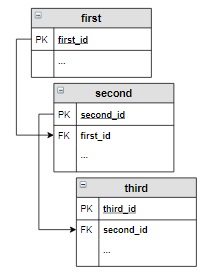
city CROSS JOIN author

ORDER BY name\_city, Дата DESC;

## Запросы на выборку из нескольких таблиц

Запрос на выборку может выбирать данные из двух и более таблиц базы данных. При этом таблицы должны быть логически связаны между собой. Для каждой пары таблиц, включаемых в запрос, необходимо указать свой оператор соединения. Наиболее распространенным является внутреннее соединение INNER JOIN, поэтому в примерах будем использовать его.

Пусть таблицы связаны между собой следующим образом:



тогда запрос на выборку для этих таблиц будет иметь вид:

SELECT

...

FROM

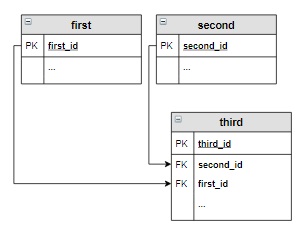
first

INNER JOIN second ON first.first\_id = second.first\_id

INNER JOIN third ON second.second\_id = third.second\_id

...

Если же таблицы связаны так:



то запрос на выборку выглядит следующим образом:

SELECT

...

FROM

first

INNER JOIN third ON first.first\_id = third.first\_id

INNER JOIN second ON third.second\_id = second.second\_id

...

В этом случае рекомендуется соединение таблиц записывать последовательно, «по кругу»: **first → third → second**.

**Пример**

Вывести информацию о тех книгах, их авторах и жанрах, цена которых принадлежит интервалу от 500  до 700 рублей  включительно.

Запрос:

SELECT title, name\_author, name\_genre, price, amount

FROM

author

INNER JOIN book ON author.author\_id = book.author\_id

INNER JOIN genre ON genre.genre\_id = book.genre\_id

WHERE price BETWEEN 500 AND 700;

Результат:

+-----------------------+----------------+------------+--------+--------+

| title | name\_author | name\_genre | price | amount |

+-----------------------+----------------+------------+--------+--------+

| Мастер и Маргарита | Булгаков М.А. | Роман | 670.99 | 3 |

| Белая гвардия | Булгаков М.А. | Роман | 540.50 | 5 |

| Стихотворения и поэмы | Есенин С.А. | Поэзия | 650.00 | 15 |

| Черный человек | Есенин С.А. | Поэзия | 570.20 | 6 |

| Лирика | Пастернак Б.Л. | Поэзия | 518.99 | 2 |

+-----------------------+----------------+------------+--------+--------+

## Задание

 Вывести информацию о книгах (жанр, книга, автор), относящихся к жанру, включающему слово «роман» в отсортированном по названиям книг виде.

**Логическая схема базы данных:**



***Текст задания (чтобы не прокручивать страницу):***

 Вывести информацию о книгах (жанр, книга, автор), относящихся к жанру, включающему слово «роман» в отсортированном по названиям книг виде.

**Результат**

+------------+--------------------+------------------+

| name\_genre | title | name\_author |

+------------+--------------------+------------------+

| Роман | Белая гвардия | Булгаков М.А. |

| Роман | Братья Карамазовы | Достоевский Ф.М. |

| Роман | Игрок | Достоевский Ф.М. |

| Роман | Идиот | Достоевский Ф.М. |

| Роман | Мастер и Маргарита | Булгаков М.А. |

+------------+--------------------+------------------+

**Структура и наполнение таблиц**

### Введите SQL запрос

SELECT name\_genre, title, name\_author

FROM genre

INNER JOIN book

ON genre.genre\_id = book.genre\_id

INNER JOIN author

ON author.author\_id = book.author\_id

WHERE name\_genre LIKE '%роман%'

ORDER BY title;

## Запросы для нескольких таблиц с группировкой

В запросах с групповыми функциями могут использоваться несколько таблиц, между которыми используются различные типы соединений.

**Пример**

Вывести количество различных книг каждого автора. Информацию отсортировать в алфавитном порядке по фамилиям  авторов.

Запрос:

SELECT name\_author, count(title) AS Количество

FROM

author INNER JOIN book

on author.author\_id = book.author\_id

GROUP BY name\_author

ORDER BY name\_author;

Результат

+------------------+------------+

| name\_author | Количество |

+------------------+------------+

| Булгаков М.А. | 2 |

| Достоевский Ф.М. | 3 |

| Есенин С.А. | 2 |

| Пастернак Б.Л. | 1 |

+------------------+------------+

При использовании соединения INNER JOIN мы не можем узнать, что книг Лермонтова на складе нет, но предполагается, что они могут быть.  Чтобы автор Лермонтов был включен в результат, нужно изменить соединение таблиц.

Запрос:

SELECT name\_author, count(title) AS Количество

FROM

author LEFT JOIN book

on author.author\_id = book.author\_id

GROUP BY name\_author

ORDER BY name\_author;

Результат:

+------------------+------------+

| name\_author | Количество |

+------------------+------------+

| Булгаков М.А. | 2 |

| Достоевский Ф.М. | 3 |

| Есенин С.А. | 2 |

| Лермонтов М.Ю. | 0 |

| Пастернак Б.Л. | 1 |

+------------------+------------+

## ****Задание****

Посчитать количество экземпляров книг каждого автора из таблицы **author**.  Вывести тех авторов, количество книг которых меньше 10, в отсортированном по возрастанию количества виде. Последний столбец назвать **Количество**.

**Логическая схема базы данных:**



**Пояснение**

Чтобы в результат были включены авторы, книг которых на складе нет, необходимо в условии отбора, кроме того, что общее количество книг каждого автора меньше 10, учесть, что у автора вообще может не быть книг (то есть **COUNT(title) = 0**).

**Результат**

+----------------+------------+

| name\_author | Количество |

+----------------+------------+

| Лермонтов М.Ю. | NULL |

| Пастернак Б.Л. | 2 |

| Булгаков М.А. | 8 |

+----------------+------------+

**Структура и наполнение таблиц**

### Введите SQL запрос

SELECT name\_author, SUM(amount) AS Количество

FROM

author LEFT JOIN book

on author.author\_id = book.author\_id

GROUP BY name\_author

HAVING Количество < 10 OR Количество IS NULL

ORDER BY Количество;

## Запросы для нескольких таблиц со вложенными запросами

В запросах, построенных на нескольких таблицах, можно использовать вложенные запросы. Вложенный запрос может быть включен:  после ключевого слова SELECT,  после FROM и в условие отбора после WHERE (HAVING).

**Пример**

Вывести авторов, общее количество книг которых на складе максимально.

Это достаточно сложный запрос, поэтому будем решать его по шагам (реализуя каждый запрос по отдельности), а потом объединим все запросы в один.

**Шаг 1.** Найдем суммарное количество книг на складе по каждому автору. Поскольку фамилии автора в этой таблице нет, то группировку будем осуществлять по **author\_id**.

Запрос:

SELECT author\_id, SUM(amount) AS sum\_amount FROM book GROUP BY author\_id

Результат:

+-----------+------------+

| author\_id | sum\_amount |

+-----------+------------+

| 1 | 8 |

| 2 | 23 |

| 3 | 21 |

| 4 | 2 |

+-----------+------------+

**Шаг 2**. В результирующей таблице предыдущего запроса необходимо найти максимальное значение, то есть 23. Для этого запросу, созданному на шаге 1, необходимо присвоить имя (например, **query\_in**) и использовать его в качестве таблицы-источника после FROM. Затем уже находить максимум по столбцу **sum\_amount**.

Запрос:

SELECT MAX(sum\_amount) AS max\_sum\_amount

FROM

(

SELECT author\_id, SUM(amount) AS sum\_amount

FROM book

GROUP BY author\_id

) query\_in

Результат:

+----------------+

| max\_sum\_amount |

+----------------+

| 23 |

+----------------+

**Шаг 3**. Выведем фамилию автора и общее количество книг для него.

Запрос:

SELECT name\_author, SUM(amount) as Количество

FROM

author INNER JOIN book

on author.author\_id = book.author\_id

GROUP BY name\_author

Результат:

+------------------+------------+

| name\_author | Количество |

+------------------+------------+

| Булгаков М.А. | 8 |

| Достоевский Ф.М. | 23 |

| Есенин С.А. | 21 |

| Пастернак Б.Л. | 2 |

+------------------+------------+

**Шаг 4**.  Включим запрос с шага 2 в условие отбора запроса с шага 3. И получим всех авторов, общее количество книг которых максимально.

 Запрос:

SELECT name\_author, SUM(amount) as Количество

FROM

author INNER JOIN book

on author.author\_id = book.author\_id

GROUP BY name\_author

HAVING SUM(amount) =

(/\* вычисляем максимальное из общего количества книг каждого автора \*/

SELECT MAX(sum\_amount) AS max\_sum\_amount

FROM

(/\* считаем количество книг каждого автора \*/

SELECT author\_id, SUM(amount) AS sum\_amount

FROM book GROUP BY author\_id

) query\_in

);

Результат:

+------------------+------------+

| name\_author | Количество |

+------------------+------------+

| Достоевский Ф.М. | 23 |

+------------------+------------+

## ****Задание****

Вывести в алфавитном порядке всех авторов, которые пишут только в одном жанре. Поскольку у нас в таблицах так занесены данные, что у каждого автора книги только в одном жанре,  для этого запроса внесем изменения в таблицу **book**. Пусть у нас  книга Есенина «Черный человек» относится к жанру «Роман», а книга Булгакова «Белая гвардия» к «Приключениям» (эти изменения в таблицы уже внесены).

**Логическая схема базы данных:**



***Текст задания (чтобы не прокручивать страницу):***

 Вывести в алфавитном порядке всех авторов, которые пишут только в одном жанре. Поскольку у нас в таблицах так занесены данные, что у каждого автора книги только в одном жанре,  для этого запроса внесем изменения в таблицу **book**. Пусть у нас  книга Есенина «Черный человек» относится к жанру «Роман», а книга Булгакова «Белая гвардия» к «Приключениям» (эти изменения в таблицы уже внесены).

**Результат**

+------------------+

| name\_author |

+------------------+

| Достоевский Ф.М. |

| Пастернак Б.Л. |

+------------------+

**Структура и наполнение таблиц**

### Введите SQL запрос

SELECT name\_author

FROM author INNER JOIN

(SELECT author\_id, COUNT(genre\_id) AS genre\_count

FROM (SELECT DISTINCT author\_id, genre\_id FROM book) q1

GROUP BY author\_id

HAVING genre\_count = 1) q2

on author.author\_id = q2.author\_id

/\*1. Отобрать из book только DISTINCT записи author\_id, genre\_id

SELECT DISTINCT author\_id, genre\_id FROM book

2. Запросом по предыдыдущему запросу для каждого автора считаем количество жанров и оставляем только авторов с одим жанром

SELECT author\_id, COUNT(genre\_id) AS genre\_count

FROM (SELECT DISTINCT author\_id, genre\_id FROM book) q1

GROUP BY author\_id

HAVING genre\_count = 1

3. Присоединяем таблицу author для получения имен авторов. Итого:

SELECT name\_author

FROM author INNER JOIN

(SELECT author\_id, COUNT(genre\_id) AS genre\_count

FROM (SELECT DISTINCT author\_id, genre\_id FROM book) q1

GROUP BY author\_id

HAVING genre\_count = 1) q2

on author.author\_id = q2.author\_id \*/