Примеры задач

1. Сформулируйте SQL запрос для создания таблицы **book**

CREATE TABLE book(

book\_id INT PRIMARY KEY AUTO\_INCREMENT,

title VARCHAR(50),

author VARCHAR(30),

price DECIMAL(8, 2),

amount INT

);

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

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

VALUES (1,"Мастер и Маргарита", "Булгаков М.А.",670.99,3);

SELECT \* FROM book;

1. Занесите три последние записи в таблицу**book**

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

VALUES (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);

select \* from book;

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

SELECT title as Название, author as Автор

FROM book;

1. Для упаковки каждой книги требуется 1 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**

SELECT title, amount,

amount \* 1.65 AS pack

from book;

1. 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.

select title, author, amount,

round((price \* 70/100),2) as new\_price

from book;

1. 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.

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;

1. 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.

select title,author,price, amount

from book

where (price < 500 or price > 600) and amount\*price >= 5000;

1. 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 .

select title,author

from book

where (price between 540.5 and 800) and (amount in (2,3,5,7));

1. 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.

select title,author

from book

where (title like "\_% \_%") and (author like '% С.%');

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

select author, title

from book

where amount between 2 and 14

order by author desc,title asc;

1. Проверить наличие на складе

select title,amount,

(amount>=10,"много","мало") as наличие

from book

where author = "Достоевский Ф.М."

ORDER BY amount;

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

SELECT amount

FROM book

GROUP BY amount;

Или

SELECT DISTINCT amount

FROM book;

1. Посчитать, количество различных книг и количество экземпляров книг каждого автора , хранящихся на складе. Вычисляемые столбцы назвать **Различных\_книг** и **Количество\_экземпляров** соответственно, столбец с фамилиями авторов назвать **Автор**.

SELECT author as Автор,

COUNT(author) as Различных\_книг,

SUM(amount) as Количество\_экземпляров

FROM book

GROUP BY author;

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

select author,MIN(price) AS Минимальная\_цена,MAX(price) AS Максимальная\_цена, AVG(price) as Средняя\_цена

FROM book

GROUP BY author;

1. 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.

select author,

ROUND(SUM(price \* amount),2) 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;

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

select ROUND(MIN(price),2) as Минимальная\_цена,

ROUND(MAX(price),2) as Максимальная\_цена,

ROUND(AVG(price),2) as Средняя\_цена

from book;

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

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;

1. Вычислить среднюю цену и суммарную стоимость тех книг, количество экземпляров которых принадлежит интервалу от 5 до 14, включительно. Столбцы переименовать как показано в образце, значения округлить до 2-х знаков после запятой.

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

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

FROM book

WHERE 5 <= amount AND amount <= 14;

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

select author,

SUM(price\*amount) as Стоимость

from book

where title <> 'Идиот' and title <> 'Белая гвардия'

group by author

having Стоимость > 5000

order by SUM(price\*amount)desc;

1. 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.

select author,title,price

from book

where price <= (select AVG(price) from book)

order by price DESC;

1. 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.

SELECT author,title,price

from book

where ABS(price - (SELECT MIN(price) FROM book)) <= 150

ORDER by price;

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

select author,title,amount

from book

where amount in (

select amount

from book

group by amount

having count(amount) = 1);

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

select author,title,price

from book

where price < ANY(

select MIN(price)

from book

group by author );

1. Посчитать сколько и каких экземпляров книг нужно заказать поставщикам, чтобы на складе было одинаковое количество экземпляров каждой книги, равное значению самого большего количества экземпляров одной книги на складе. Столбцу с количеством заказываемых экземпляров книг присвоить имя **Заказ**.

SELECT title, author, amount,

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

FROM book

where amount < (select MAX(amount) from book);

1. Можно посмотреть, при продаже всех книг, какая книга принесет больше всего выручки, в процентах.

SELECT \*, round((100\*price\*amount/(SELECT SUM(price\*amount) FROM book)), 2) AS income\_percent

FROM book

ORDER BY income\_percent DESC;

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

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;

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

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;

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

UPDATE book

SET price = 0.9 \* price

WHERE amount between 5 and 10 ;

SELECT \* FROM book;

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

UPDATE book

SET price = ROUND(IF(buy = 0, price\*0.9, price), 2), buy = IF(buy > amount, amount, buy);

SELECT \* FROM book

1. 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

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;

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

DELETE FROM supply

WHERE author IN (

SELECT author

FROM book

GROUP BY author

Having SUM(amount) > 10

);

SELECT \* FROM supply;

1. 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**.

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. 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). Информацию вывести в упорядоченном по убыванию длительности поездки, а потом по убыванию названий городов (в обратном алфавитном порядке).Для вычитания двух дат используется функция DATEDIFF(дата\_1, дата\_2), результатом которой является количество дней между дата\_1 и дата\_2. Например,\*/

select name, city

(DATEDIFF(date\_last, date\_first) + 1) as Длительность

from trip

where city NOT IN ( 'Москва' , 'Санкт-Петербург')

order by Длительность;

1. Вывести информацию о командировках сотрудника(ов), которые были самыми короткими по времени.

select name, city, date\_first, date\_last

from trip

where DATEDIFF(date\_last, date\_first) + 1 <= ANY(

select MIN(DATEDIFF(date\_last, date\_first) + 1)

from trip);

1. Вывести информацию о командировках, начало и конец которых относятся к одному месяцу (год может быть любой). Результат отсортировать сначала  в алфавитном порядке по названию города, а затем по фамилии сотрудника .

select name,city,date\_first,date\_last

from trip

where month(date\_first) = month(date\_last)

order by city,name;

1. 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.

select MONTHNAME(date\_first) as Месяц,

count(MONTHNAME(date\_first)) as Количество

from trip

group by Месяц

order by Количество desc,Месяц;

1. 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**.  Информацию отсортировать сначала  в алфавитном порядке по фамилиям сотрудников, а затем по убыванию суммы суточных.

select name,city,date\_first,

(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;

1. 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.

select name,

SUM((DATEDIFF(date\_last, date\_first) + 1) \* per\_diem) as Сумма

from trip

group by name

having count(name) > 3

order by Сумма desc ;

1. 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**.

UPDATE fine AS f, traffic\_violation AS tv

SET f.sum\_fine=IF(f.sum\_fine is Null,

(SELECT sum\_fine FROM traffic\_violation tv WHERE f.violation=tv.violation),

f.sum\_fine);

SELECT \* FROM fine;

1. *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.*

select name, number\_plate, violation

from fine

group by name, number\_plate, violation

having count(\*) > 1

order by name;

SELECT name, number\_plate, violation

FROM fine

GROUP BY name, number\_plate, violation

HAVING COUNT(violation) >= 2

1. В таблице fine увеличить в два раза сумму неоплаченных штрафов для отобранных на предыдущем шаге записей.

Пояснение

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.

Если водитель совершил нарушение на другой машине, ему увеличивать штраф не нужно.

Если несколько повторных нарушений не оплачены, то штраф увеличить для всех.

Этот запрос реализован на предыдущем шаге.

CREATE TABLE query\_in as

SELECT name, number\_plate, violation

FROM fine

GROUP BY name, number\_plate, violation

HAVING COUNT(violation) >= 2

order by name;

select \* from query\_in;

UPDATE fine, query\_in

SET sum\_fine = sum\_fine \* 2

WHERE fine.number\_plate = query\_in.number\_plate and fine.date\_payment is Null ;

SELECT \* FROM fine;

1. 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**.

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;

1. Удалить из таблицы **fine** информацию о нарушениях, совершенных раньше 1 февраля 2020 года.

delete from fine

where date\_violation < '2020-02-01';

select \* from fine;

1. Перепишите запрос на создание таблицы **book** , чтобы ее структура соответствовала структуре, показанной на логической схеме (порядок следования столбцов - как на логической схеме в таблице **book**, **genre\_id**  - внешний ключ) . Для **genre\_id** ограничение о недопустимости пустых значений **не задавать**. В качестве главной таблицы для описания поля  **genre\_id**использовать таблицу **[genre](https://stepik.org/lesson/297508/step/5?unit=279268)** следующей структуры:

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

**Логическая схема** (нужно создать только таблицу **book**):



CREATE TABLE book (

book\_id INT PRIMARY KEY AUTO\_INCREMENT,

title VARCHAR(50),

author\_id INT NOT NULL,

genre\_id INT,

price DECIMAL(8,2),

amount INT,

FOREIGN KEY (author\_id) REFERENCES author (author\_id),

FOREIGN KEY (genre\_id) REFERENCES genre (genre\_id)

);

SHOW COLUMNS FROM book;