**Содержание урока**

В запросах SQL могут участвовать несколько таблиц базы данных. При этом необходимо указать как эти таблицы соединены между собой.

Операция соединения **JOIN** предназначена для обеспечения выборки данных из двух таблиц и включения этих данных в один результирующий набор. При необходимости соединения не двух, а нескольких таблиц, операция соединения применяется несколько раз (последовательно).

Операторы соединения входят в раздел **FROM** SQL запросов.

В данном уроке будут созданы запросы:

* [для двух таблиц, внутреннее соединение INNER JOIN](https://stepik.org/lesson/308886/step/2?unit=291012);
* [для двух таблиц, внешние соединения LEFT JOIN и RIGHT JOIN](https://stepik.org/lesson/308886/step/3?unit=291012);
* [для двух таблиц, перекрестное соединение CROSS JOIN](https://stepik.org/lesson/308886/step/4?unit=291012);
* [выборки данных из нескольких таблиц](https://stepik.org/lesson/308886/step/5?unit=291012);
* [выборки данных из нескольких таблиц с группировкой](https://stepik.org/lesson/308886/step/6?unit=291012);
* [выборки данных с применением вложенных запросов](https://stepik.org/lesson/308886/step/7?unit=291012);
* [вложенные запросы в операторах соединения](https://stepik.org/lesson/308886/step/8?unit=291012),
* [операторы соединения, использование USING](https://stepik.org/lesson/308886/step/9?unit=291012).

**Структура и наполнение таблиц**

**Концептуальная схема базы данных:**



**Логическая схема базы данных:**



Таблица**author**([создание](https://stepik.org/lesson/308885/step/6?unit=291011), [заполнение](https://stepik.org/lesson/308885/step/7?unit=291011)):

|  |  |
| --- | --- |
| **author\_id** | **name\_author** |
| 1 | Булгаков М.А. |
| 2 | Достоевский Ф.М. |
| 3 | Есенин С.А. |
| 4 | Пастернак Б.Л. |
| 5 | Лермонтов М.Ю. |

Таблица**genre**([создание](https://stepik.org/lesson/297508/step/6?unit=279268), [заполнение](https://stepik.org/lesson/297508/step/7?unit=279268), рассмотрено в качестве примеров):

|  |  |
| --- | --- |
| **genre\_id** | **name\_genre** |
| 1 | Роман |
| 2 | Поэзия |
| 3 | Приключения |

Таблица **book** ([создание](https://stepik.org/lesson/308885/step/9?unit=291011), [заполнение](https://stepik.org/lesson/308885/step/11?unit=291011)):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author\_id** | **genre\_id** | **price** | **amount** |
| 1 | Мастер и Маргарита | 1 | 1 | 670.99 | 3 |
| 2 | Белая гвардия | 1 | 1 | 540.50 | 5 |
| 3 | Идиот | 2 | 1 | 460.00 | 10 |
| 4 | Братья Карамазовы | 2 | 1 | 799.01 | 3 |
| 5 | Игрок | 2 | 1 | 480.50 | 10 |
| 6 | Стихотворения и поэмы | 3 | 2 | 650.00 | 15 |
| 7 | Черный человек | 3 | 2 | 570.20 | 6 |
| 8 | Лирика | 4 | 2 | 518.99 | 2 |

**Соединение INNER JOIN**

Оператор внутреннего соединения INNER JOIN соединяет две таблицы. Порядок таблиц для оператора неважен, поскольку оператор является симметричным.

SELECT

...

FROM

таблица\_1 INNER JOIN таблица\_2

ON условие

...

Результат запроса формируется так:

* каждая строка одной таблицы сопоставляется с каждой строкой второй таблицы;
* для полученной «соединённой» строки проверяется условие соединения;
* если условие истинно, в таблицу результата добавляется соответствующая «соединённая» строка;

**Пример**

Вывести название книг и их авторов.

*Запрос:*

SELECT title, name\_author

FROM

author INNER JOIN book

ON author.author\_id = book.author\_id;

Поскольку поля **author\_id** в таблицах **book** и **author** называются одинаково, необходимо в запросах указывать полную ссылку на них (**book.author\_id** и **author.author\_id**).

*Результат:*

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

| title | name\_author |

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

| Мастер и Маргарита | Булгаков М.А. |

| Белая гвардия | Булгаков М.А. |

| Идиот | Достоевский Ф.М. |

| Братья Карамазовы | Достоевский Ф.М. |

| Игрок | Достоевский Ф.М. |

| Стихотворения и поэмы | Есенин С.А. |

| Черный человек | Есенин С.А. |

| Лирика | Пастернак Б.Л. |

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

В данном запросе осуществляется соединение главной таблицы **author** и зависимой таблицы **book** по ключевому столбцу**author.author\_id** и внешнему ключу **book.author\_id**. При этом в результирующую таблицу запроса включаются все строки, в которых значения этих столбцов совпадают. Другими словами строки зависимой таблицы **book** дополняются фамилией и инициалами авторов из таблицы **author**.

**Задание**

Вывести название, жанр и цену тех книг, количество которых больше 8, в отсортированном по убыванию цены виде.

**Логическая схема базы данных:**



***Текст задания****(чтобы не прокручивать страницу):*

*Вывести название, жанр и цену тех книг, количество которых больше 8, в отсортированном по убыванию цены виде.*

**Результат**

**Структура и наполнение таблиц**

Таблица genre:

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

| genre\_id | name\_genre |

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

| 1 | Роман |

| 2 | Поэзия |

| 3 | Приключения |

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

Таблица author:

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

| author\_id | name\_author |

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

| 1 | Булгаков М.А. |

| 2 | Достоевский Ф.М. |

| 3 | Есенин С.А. |

| 4 | Пастернак Б.Л. |

| 5 | Лермонтов М.Ю. |

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

Таблица book:

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

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

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

| 1 | Мастер и Маргарита | 1 | 1 | 670.99 | 3 |

| 2 | Белая гвардия | 1 | 1 | 540.50 | 5 |

| 3 | Идиот | 2 | 1 | 460.00 | 10 |

| 4 | Братья Карамазовы | 2 | 1 | 799.01 | 3 |

| 5 | Игрок | 2 | 1 | 480.50 | 10 |

| 6 | Стихотворения и поэмы | 3 | 2 | 650.00 | 15 |

| 7 | Черный человек | 3 | 2 | 570.20 | 6 |

| 8 | Лирика | 4 | 2 | 518.99 | 2 |

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

Enter an SQL query

Correct answer from **30,244** learners

Total **67%** of tries are correct

 Totally right.

Query result: +-----------------------+------------+--------+ | title | name\_genre | price | +-----------------------+------------+--------+ | Стихотворения и поэмы | Поэзия | 650.00 | | Игрок | Роман | 480.50 | | Идиот | Роман | 460.00 | +-----------------------+------------+--------+ Affected rows: 3

Hide

1

SELECT title, name\_genre, price

2

FROM genre INNER JOIN book

3

   ON genre.genre\_id = book.genre\_id

4

WHERE amount > 8

5

ORDER BY price DESC

6

;

[==================================================================================

**Внешнее соединение LEFT и RIGHT OUTER JOIN**

Оператор внешнего соединения LEFT OUTER JOIN  (можно использовать LEFT JOIN) соединяет две таблицы. Порядок таблиц для оператора важен, поскольку оператор не является симметричным.

SELECT

...

FROM

таблица\_1 LEFT JOIN таблица\_2

ON условие

...

Результат запроса формируется так:

1. в результат включается внутреннее соединение (INNER JOIN) первой и второй таблицы в соответствии с условием;
2. затем в результат добавляются те записи первой таблицы, которые не вошли во внутреннее соединение на шаге 1, для таких записей соответствующие поля второй таблицы заполняются значениями NULL.

Соединение RIGHT JOIN действует аналогично, только в пункте 2 первая таблица меняется на вторую и наоборот.

**Пример**

Вывести название всех книг каждого автора, если книг некоторых авторов в данный момент нет на складе – вместо названия книги указать Null.

*Запрос:*

SELECT name\_author, title

FROM author LEFT JOIN book

ON author.author\_id = book.author\_id

ORDER BY name\_author;

Результат:

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

| name\_author | title |

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

| Булгаков М.А. | Мастер и Маргарита |

| Булгаков М.А. | Белая гвардия |

| Достоевский Ф.М. | Игрок |

| Достоевский Ф.М. | Идиот |

| Достоевский Ф.М. | Братья Карамазовы |

| Есенин С.А. | Стихотворения и поэмы |

| Есенин С.А. | Черный человек |

| Лермонтов М.Ю. | NULL |

| Пастернак Б.Л. | Лирика |

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

Так как в таблице**book** нет книг Лермонтова, напротив этой фамилии стоит Null.

**Задание**

Вывести все жанры, которые не представлены в книгах на складе.

**Логическая схема базы данных:**



**Пояснение**

***Текст задания (чтобы не прокручивать страницу):***

*Вывести все жанры, которые не представлены в книгах на складе.*

**Результат**

**Структура и наполнение таблиц**

Enter an SQL query

Correct answer from **29,370** learners

Total **68%** of tries are correct

 You're right!

Query result: +-------------+ | name\_genre | +-------------+ | Приключения | +-------------+ Affected rows: 1

Hide

1

SELECT name\_genre

2

FROM genre LEFT JOIN book

3

   ON genre.genre\_id = book.genre\_id

4

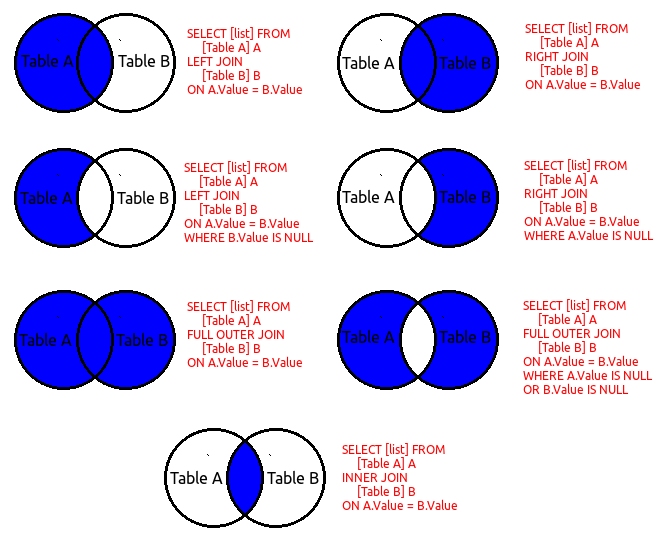
WHERE title IS NULL

5

;

6

​



**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 |

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

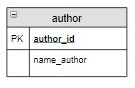
**Задание**

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**

***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**

**Структура и наполнение таблиц**

Enter an SQL query

1

SELECT name\_city, name\_author,

2

    DATE\_ADD('2020-01-01', INTERVAL FLOOR(RAND() \* 365)DAY) AS Дата

3

FROM city, author

4

ORDER BY name\_city ASC, Дата DESC

5

;

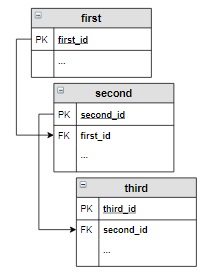
6

​

**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

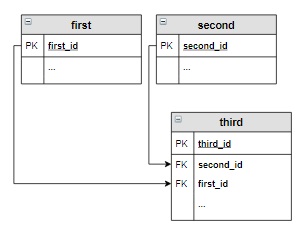
first

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.*

1

SELECT name\_genre, title, name\_author

2

FROM genre

3

       INNER JOIN book ON genre.genre\_id = book.genre\_id

4

       INNER JOIN author ON book.author\_id = author.author\_id

5

WHERE name\_genre = 'роман'

6

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**).

***Текст задания (чтобы не прокручивать страницу):***

*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****.*

**Результат**

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

| name\_author | Количество |

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

| Лермонтов М.Ю. | NULL |

| Пастернак Б.Л. | 2 |

| Булгаков М.А. | 8 |

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

1

SELECT name\_author, SUM(amount) AS 'Количество'

2

FROM author LEFT JOIN book

3

    ON author.author\_id = book.author\_id

4

GROUP BY name\_author

5

HAVING Количество < 10 OR Количество IS NULL

6

ORDER BY Количество

7

;

8

​

9

10

​

11

​

**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).*

1

SELECT name\_author

2

FROM genre

3

       INNER JOIN book ON genre.genre\_id = book.genre\_id

4

       INNER JOIN author ON book.author\_id = author.author\_id

5

GROUP BY name\_author

6

HAVING COUNT( DISTINCT(name\_genre))=1

7

ORDER BY name\_author

8

;

**Вложенные запросы в операторах соединения**

Вложенные запросы могут использоваться в операторах соединения JOIN.  При этом им необходимо присваивать имя, которое записывается сразу после закрывающей скобки вложенного запроса.

SELECT

...

FROM

таблица ... JOIN

(

SELECT ...

) имя\_вложенного\_запроса

ON условие

...

Вложенный запрос может стоять как справа, так и слева от оператора JOIN. Допускается использование двух запросов в операторах соединения.

**Пример**

Вывести авторов, пишущих книги в самом популярном жанре. Указать этот жанр.

Самым популярным считать жанр, общее количество экземпляров книг которого на складе максимально. Таких жанров может быть несколько, если они имеют одинаковое максимальное значение общего количества экземпляров. Только для этого шага изменена запись в таблице **book**.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author\_id** | **genre\_id** | **price** | **amount** |
| 8 | Лирика | 4 | 2 | 518.9910 | 10 |

А также добавлены новые записи:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **book\_id** | **title** | **author\_id** | **genre\_id** | **price** | **amount** |
| 9 | Герой нашего времени | 5 | 3 | 570.59 | 2 |
| 10 | Доктор Живаго | 4 | 3 | 740.50 | 5 |

Рассмотрим реализацию этого запроса по шагам.

**Шаг 1.** Найдем общее количество книг по каждому жанру, отсортируем его по убыванию и ограничим вывод одной строкой. Рекомендуется, если запрос будет использоваться в качестве вложенного (особенно в операциях соединения), вычисляемым полям запроса давать собственное имя.

*Запрос:*

SELECT genre\_id, SUM(amount) AS sum\_amount

FROM book

GROUP BY genre\_id

ORDER BY sum\_amount DESC

LIMIT 1

*Результат:*

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

| genre\_id | sum\_amount |

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

| 1 | 31 |

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

Кажется, что, уже используя этот запрос, можно получить**id** самого популярного жанра. Но это не так, поскольку несколько жанров могут иметь одинаковую популярность. Поэтому нам необходим запрос, который отберет ВСЕ жанры, суммарное количество книг которых равно **sum\_amount**.

**Шаг 2.** Используя запрос с предыдущего шага, найдем **id** самых популярных жанров.

*Запрос:*

SELECT query\_in\_1.genre\_id

FROM

(/\* выбираем код жанра и количество произведений, относящихся к нему \*/

SELECT genre\_id, SUM(amount) AS sum\_amount

FROM book

GROUP BY genre\_id

)query\_in\_1

INNER JOIN

(/\* выбираем запись, в которой указан код жанр с максимальным количеством книг \*/

SELECT genre\_id, SUM(amount) AS sum\_amount

FROM book

GROUP BY genre\_id

ORDER BY sum\_amount DESC

LIMIT 1

) query\_in\_2

ON query\_in\_1.sum\_amount= query\_in\_2.sum\_amount

*Результат:*

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

| genre\_id |

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

| 1 |

| 2 |

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

**Шаг 3.** Используя запрос с шага 2, выведем фамилии авторов, которые пишут в самых популярных жанрах, и названия этих жанров. В этом запросе обязательно выполнить группировку по фамилиям авторов и **id** жанров, так как без этого фамилии авторов будут повторяться, поскольку в таблице**book** есть разные книги, написанные автором в одном жанре.

*Запрос:*

SELECT name\_author, name\_genre

FROM

author

INNER JOIN book ON author.author\_id = book.author\_id

INNER JOIN genre ON book.genre\_id = genre.genre\_id

GROUP BY name\_author,name\_genre, genre.genre\_id

HAVING genre.genre\_id IN

(/\* выбираем автора, если он пишет книги в самых популярных жанрах\*/

SELECT query\_in\_1.genre\_id

FROM

( /\* выбираем код жанра и количество произведений, относящихся к нему \*/

SELECT genre\_id, SUM(amount) AS sum\_amount

FROM book

GROUP BY genre\_id

)query\_in\_1

INNER JOIN

( /\* выбираем запись, в которой указан код жанр с максимальным количеством книг \*/

SELECT genre\_id, SUM(amount) AS sum\_amount

FROM book

GROUP BY genre\_id

ORDER BY sum\_amount DESC

LIMIT 1

) query\_in\_2

ON query\_in\_1.sum\_amount= query\_in\_2.sum\_amount

);

**Важно!**

1. Обратите внимание, что в группировку включен столбец**genre\_id**, который используется в HAVING. Это связано с тем, что в HAVING можно использовать либо столбцы, перечисленные в GROUP BY, либо вычисляемые с помощью групповых функций столбцы. Добавление столбца **genre\_id**не влияет на группировку, так как между названием жанра и его **id** - взаимно-однозначное соответствие.
2. Название столбца**genre\_id**задается с указанием имени таблицы (**genre.genre\_id**), так как этот столбец входит в структуру двух таблиц **book** и **genre**.  Для этого запроса можно было бы указать и **book.genre\_id**, так как эти таблицы связаны внутренним соединением INNER JOIN и имеют одинаковые значения в полях **genre.genre\_id** и **book.genre\_id**.

*Результат:*

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

| name\_author | name\_genre |

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

| Достоевский Ф.М. | Роман |

| Булгаков М.А. | Роман |

| Пастернак Б.Л. | Поэзия |

| Есенин С.А. | Поэзия |

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

**Задание**

Вывести информацию о книгах (название книги, фамилию и инициалы автора, название жанра, цену и количество экземпляров книги), написанных в самых популярных жанрах, в отсортированном в алфавитном порядке по названию книг виде. Самым популярным считать жанр, общее количество экземпляров книг которого на складе максимально.

**Логическая схема базы данных:**



***Текст задания (чтобы не прокручивать страницу):***

*Вывести информацию о книгах (название книги, фамилию и инициалы автора, название жанра, цену и количество экземпляров книг), написанных в самых популярных жанрах, в отсортированном в алфавитном порядке по названию книг виде. Самым популярным считать жанр, общее количество экземпляров книг которого на складе максимально.*

1

SELECT title, name\_author, name\_genre, price, amount

2

FROM

3

  author

4

   INNER JOIN book ON author.author\_id = book.author\_id

5

   INNER JOIN genre ON book.genre\_id = genre.genre\_id

6

GROUP BY title, name\_author, name\_genre, price, amount, genre.genre\_id

7

HAVING genre.genre\_id IN

8

        (/\* выбираем автора, если он пишет книги в самых популярных жанрах\*/

9

         SELECT query\_in\_1.genre\_id

10

         FROM

11

             ( /\* выбираем код жанра и количество произведений, относящихся к нему \*/

12

               SELECT genre\_id, SUM(amount) AS sum\_amount

13

               FROM book

14

               GROUP BY genre\_id

15

              )query\_in\_1

16

         INNER JOIN

17

             ( /\* выбираем запись, в которой указан код жанр с максимальным количеством книг \*/

18

               SELECT genre\_id, SUM(amount) AS sum\_amount

19

               FROM book

20

               GROUP BY genre\_id

21

               ORDER BY sum\_amount DESC

22

               LIMIT 1

23

              ) query\_in\_2

24

         ON query\_in\_1.sum\_amount= query\_in\_2.sum\_amount

25

        )

26

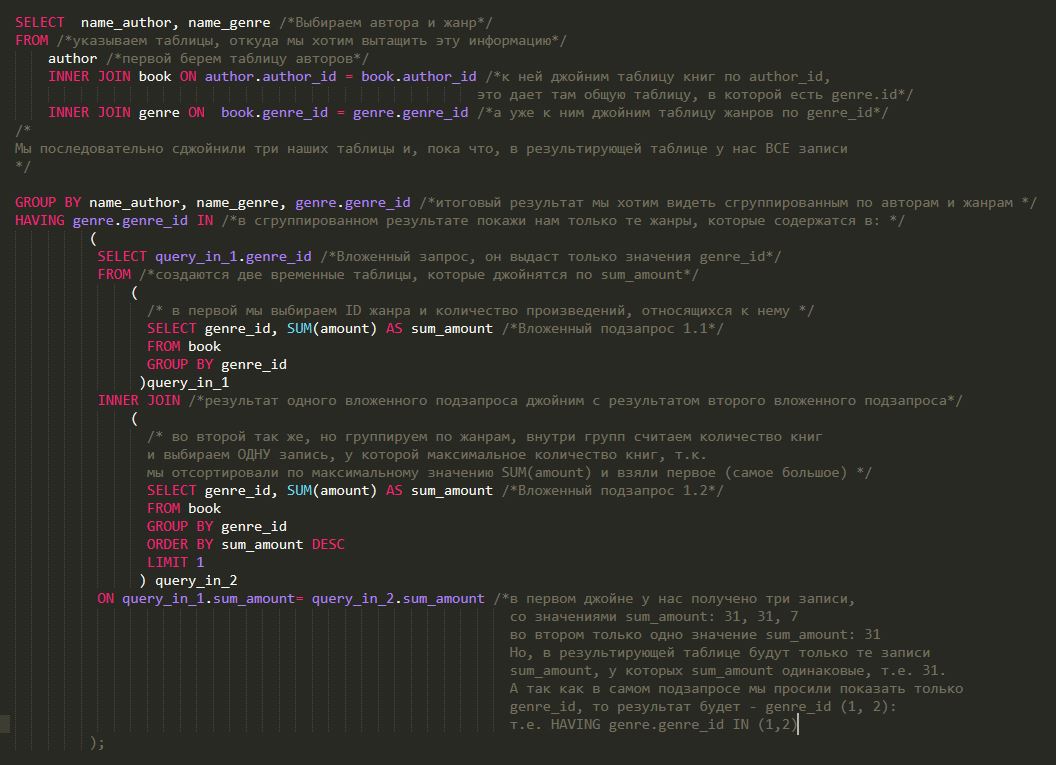
ORDER BY title;

27

​

28

​



**Операция соединение, использование USING()**

*Данный шаг добавлен по предложениям пользователей*([Валерий Родькин](https://stepik.org/users/60542727), [Todor Illia](https://stepik.org/users/33149099)  и другие)*.*

При описании соединения таблиц с помощью **JOIN** в некоторых случаях вместо **ON**и следующего за ним условия можно использовать оператор**USING()**.

**USING** позволяет указать набор столбцов, которые есть в обеих объединяемых таблицах. Если база данных хорошо спроектирована, а каждый внешний ключ имеет такое же имя, как и соответствующий первичный ключ (например, **genre.genre\_id = book.genre\_id**), тогда можно использовать предложение**USING** для реализации операции **JOIN**.

При этом после **SELECT**,при использовании столбцов из **USING()**, необязательно указывать, из какой именно таблицы берется столбец.

**Пример**

Вывести название книг, фамилии и **id** их авторов.

*Запрос:*

Вариант с **ON**

SELECT title, name\_author, author.author\_id /\* явно указать таблицу - обязательно \*/

FROM

author INNER JOIN book

ON author.author\_id = book.author\_id;

Вариант с **USING**

SELECT title, name\_author, author\_id /\* имя таблицы, из которой берется author\_id, указывать не обязательно\*/

FROM

author INNER JOIN book

USING(author\_id);

*Результат (одинаковый для обоих запросов):*

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

| title | name\_author | author\_id |

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

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

| Белая гвардия | Булгаков М.А. | 1 |

| Идиот | Достоевский Ф.М. | 2 |

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

| Игрок | Достоевский Ф.М. | 2 |

| Стихотворения и поэмы | Есенин С.А. | 3 |

| Черный человек | Есенин С.А. | 3 |

| Лирика | Пастернак Б.Л. | 4 |

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

Запись условия соединения с **ON**является более общим случаем, так как

* позволяет задавать соединение не только по одноименным полям;
* позволяет использовать произвольное условие на соединение таблиц, при этом в условие может включаться произвольное выражение, например, можно указать связь двух таблиц по двум и более столбцам.

**Пример**

В таблице **supply**занесена информация о книгах, поступивших на склад.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **supply\_id** | **title** | **author** | **price** | **amount** |
| 1 | Доктор Живаго | Пастернак Б.Л. | 618.99 | 3 |
| 2 | Черный человек | Есенин С.А. | 570.20 | 6 |
| 3 | Евгений Онегин | Пушкин А.С. | 440.80 | 5 |
| 4 | Идиот | Достоевский Ф.М. | 360.80 | 3 |

Если в таблицах **supply**  и **book** есть одинаковые книги,  вывести их название и автора. При этом учесть, что у нескольких авторов могут быть книги с одинаковым названием.

**Важно.**В данном примере для соединения **book** и **supply** использовать **USING** нельзя, так как:

* в таблице **book** фамилий авторов вообще нет (их необходимо получить из таблицы**author**, столбец**name\_author**),  а в таблице **supply** фамилии занесены в столбец **author**;
* для однозначной идентификации книги нужно указать, что совпадают не только названия, но и авторы книг.

*Запрос:*

SELECT book.title, name\_author

FROM

author

INNER JOIN book USING (author\_id)

INNER JOIN supply ON book.title = supply.title

and author.name\_author = supply.author;

*Результат:*

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

| title | name\_author |

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

| Идиот | Достоевский Ф.М. |

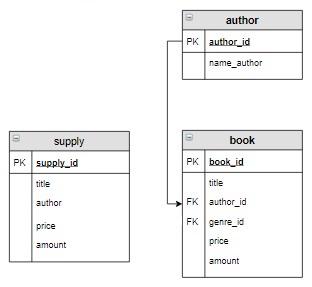
| Черный человек | Есенин С.А. |

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

**Задание**

Если в таблицах **supply**  и **book** есть одинаковые книги, которые имеют равную цену,  вывести их название и автора, а также посчитать общее количество экземпляров книг в таблицах **supply** и **book**,  столбцы назвать **Название, Автор**  и **Количество.**

**Схема данных:**



***Текст задания (чтобы не прокручивать страницу):***

*Если в таблицах****supply****и****book****есть одинаковые книги, которые имеют равную цену,  вывести их название и автора, а также посчитать общее количество экземпляров книг в таблицах****supply****и****book****,  столбцы назвать****Название, Автор****и****Количество.***

1

SELECT book.title AS 'Название', name\_author AS 'Автор', book.amount + supply.amount

2

       /\*SUM(book.amount) + SUM(supply.amount)\*/ /\*SUM(book.amount)\*2\*/ AS 'Количество'

3

FROM

4

  author

5

   INNER JOIN book USING (author\_id)

6

   INNER JOIN supply ON book.title = supply.title

7

                        and author.name\_author = supply.author

8

                        AND book.price = supply.price

9

GROUP BY name\_author, book.title, book.amount, supply.amount;

10

​

11

​

12

​