

# SQL ЗАПРОСЫ В БАЗАХ ДАННЫХ

ОБЪЕДИНЕНИЕ ТАБЛИЦ, ГРУППИРОВКА ЗАПИСЕЙ



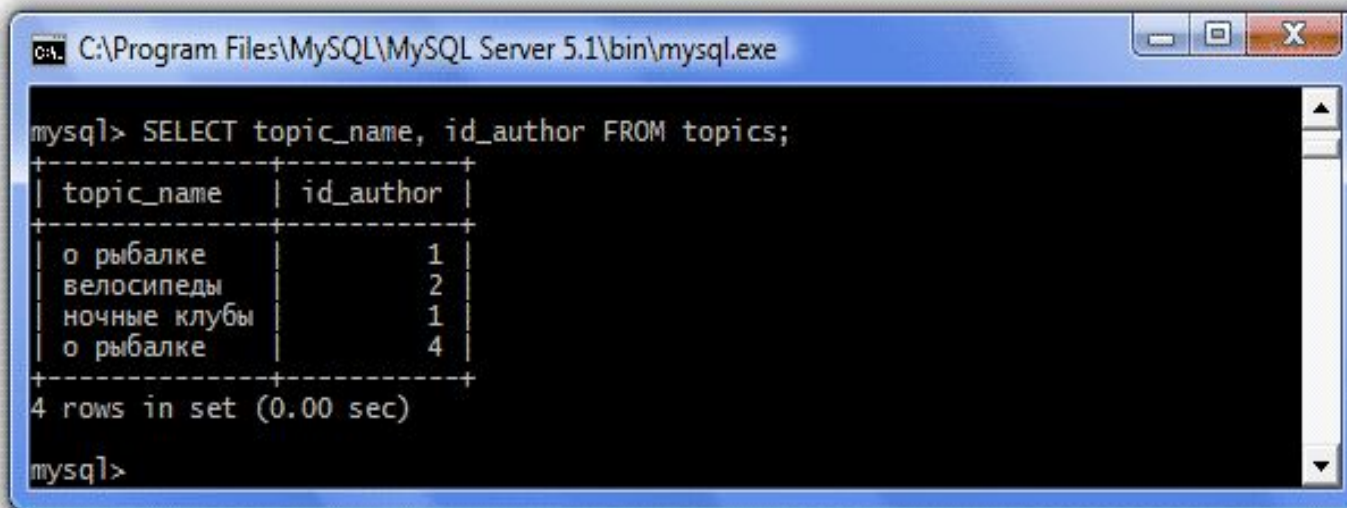
# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНУТРЕННЕЕ ОБЪЕДИНЕНИЕ)

Синтаксис самого простого объединения следующий:

```
SELECT имена_столбцов_таблицы_1, имена_столбцов_таблицы_2  
FROM имя_таблицы_1, имя_таблицы_2;
```

Например, у нас есть две таблицы:

- в первой хранятся идентификаторы авторов и их имена (users);
- во второй темы, созданные авторами и их идентификаторы (topics).



The screenshot shows a MySQL command prompt window with the following content:

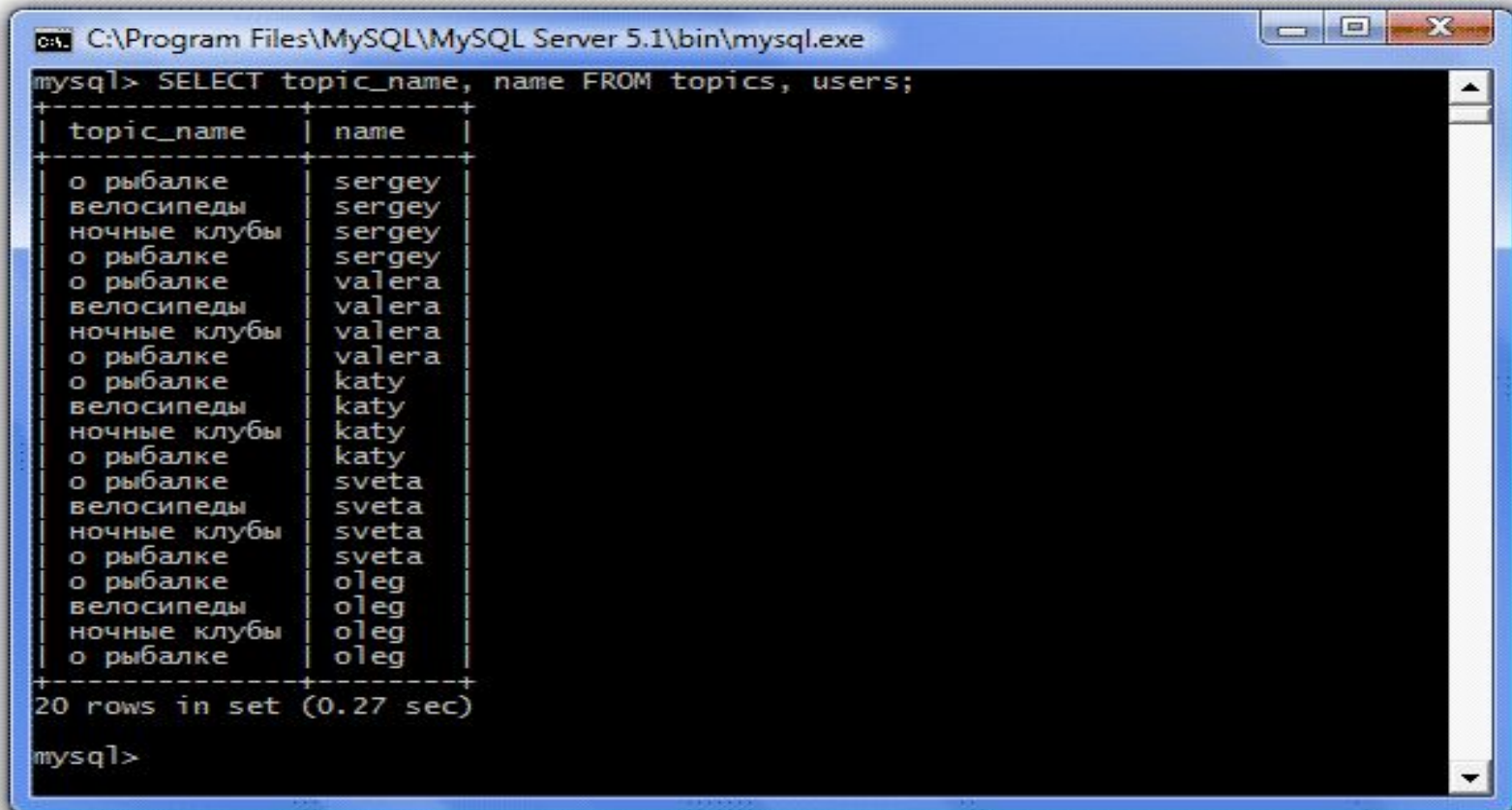
```
C:\Program Files\MySQL\MySQL Server 5.1\bin> mysql.exe  
mysql> SELECT topic_name, id_author FROM topics;  
+-----+-----+  
| topic_name | id_author |  
+-----+-----+  
| о рыбалке  |         1 |  
| велосипеды |         2 |  
| ночные клубы |         1 |  
| о рыбалке  |         4 |  
+-----+-----+  
4 rows in set (0.00 sec)  
mysql>
```

# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНУТРЕННЕЕ ОБЪЕДИНЕНИЕ)

Необходимо создать запрос, чтобы в ответе были не идентификаторы авторов, а их имена.

Создадим простое объединение:

```
SELECT topic_name, name FROM topics, users
```



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe
mysql> SELECT topic_name, name FROM topics, users;
+-----+-----+
| topic_name | name |
+-----+-----+
| о рыбалке | sergey |
| велосипеды | sergey |
| ночные клубы | sergey |
| о рыбалке | sergey |
| о рыбалке | valera |
| велосипеды | valera |
| ночные клубы | valera |
| о рыбалке | valera |
| о рыбалке | katy |
| велосипеды | katy |
| ночные клубы | katy |
| о рыбалке | katy |
| о рыбалке | sveta |
| велосипеды | sveta |
| ночные клубы | sveta |
| о рыбалке | sveta |
| о рыбалке | oleg |
| велосипеды | oleg |
| ночные клубы | oleg |
| о рыбалке | oleg |
+-----+-----+
20 rows in set (0.27 sec)

mysql>
```

# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНУТРЕННЕЕ ОБЪЕДИНЕНИЕ)

Получилось не совсем то, ожидалось.

Такое объединение научно называется декартовым произведением, когда каждой строке первой таблицы ставится в соответствие каждая строка второй таблицы.

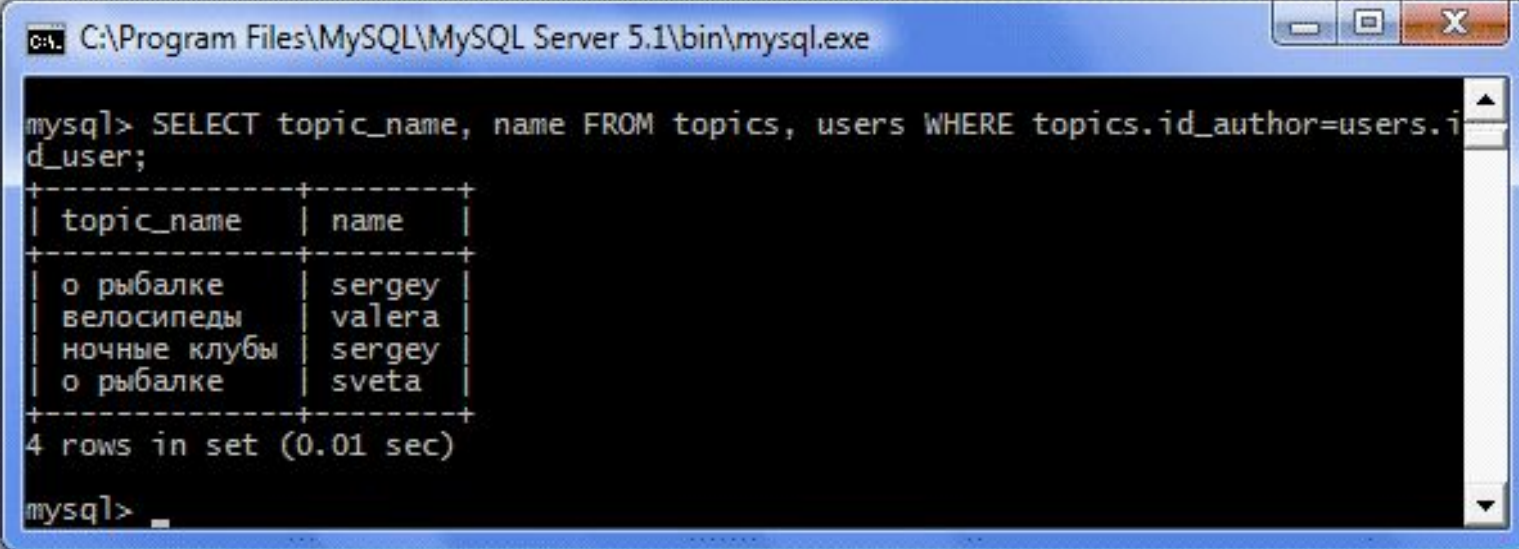
Чтобы результирующая таблица выглядела так, как нам нужно, необходимо указать условие объединения.

Мы связываем наши таблицы по идентификатору автора, это и будет нашим условием.

Т.е. мы укажем в запросе, что необходимо выводить только те строки, в которых значения поля `id_author` таблицы `topics` совпадают со значениями поля `id_user` таблицы `users`:

```
SELECT topic_name, name FROM topics, users  
WHERE topics.id_autor=users.id_user;
```

# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНУТРЕННЕЕ ОБЪЕДИНЕНИЕ)



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe

mysql> SELECT topic_name, name FROM topics, users WHERE topics.id_author=users.i
d_user;

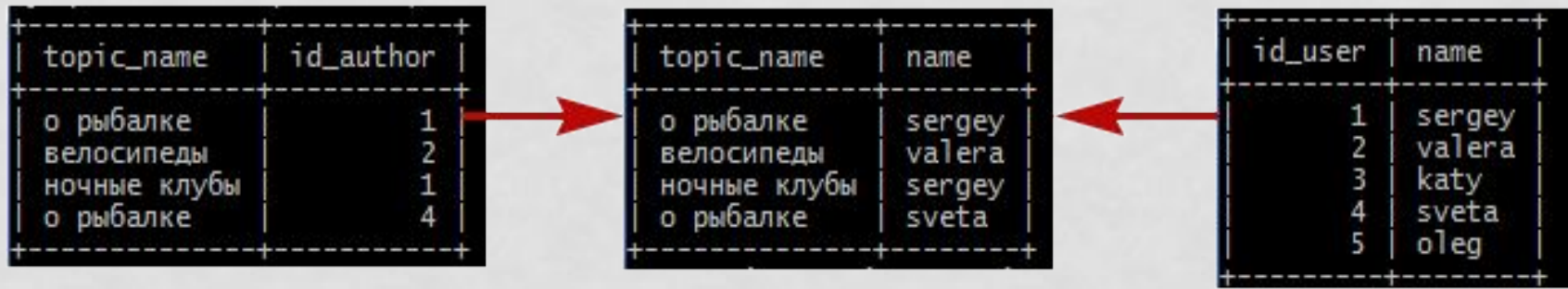
+-----+-----+
| topic_name | name |
+-----+-----+
| о рыбалке | sergey |
| велосипеды | valera |
| ночные клубы | sergey |
| о рыбалке | sveta |
+-----+-----+
4 rows in set (0.01 sec)

mysql> _
```

Т.е. мы в запросе сделали следующее условие: если в обеих таблицах есть одинаковые идентификаторы, то строки с этим идентификатором необходимо объединить в одну результирующую строку.

# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНУТРЕННЕЕ ОБЪЕДИНЕНИЕ)

На схеме это выглядит следующим образом:



Если в одной из объединяемых таблиц есть строка с идентификатором, которого нет в другой объединяемой таблице, то в результирующей таблице строки с таким идентификатором не будет. В нашем примере есть пользователь Oleg (id=5), но он не создавал тем, поэтому в результате запроса его нет.

При указании условия название столбца пишется после названия таблицы, в которой этот столбец находится (через точку). Это сделано во избежание путаницы, так как столбцы в разных таблицах могут иметь одинаковые названия.

# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНУТРЕННЕЕ ОБЪЕДИНЕНИЕ)

Синтаксис объединения с условием:

```
SELECT имя_таблицы_1.имя_столбца1_таблицы_1,  
       имя_таблицы_1.имя_столбца2_таблицы_1,  
       имя_таблицы_2.имя_столбца1_таблицы_2,  
       имя_таблицы_2.имя_столбца2_таблицы_2
```

```
FROM
```

```
    имя_таблицы_1, имя_таблицы_2
```

```
WHERE
```

```
    имя_таблицы_1.имя_столбца_по_которому_объединяем =  
    имя_таблицы_2.имя_столбца_по_которому_объединяем;
```

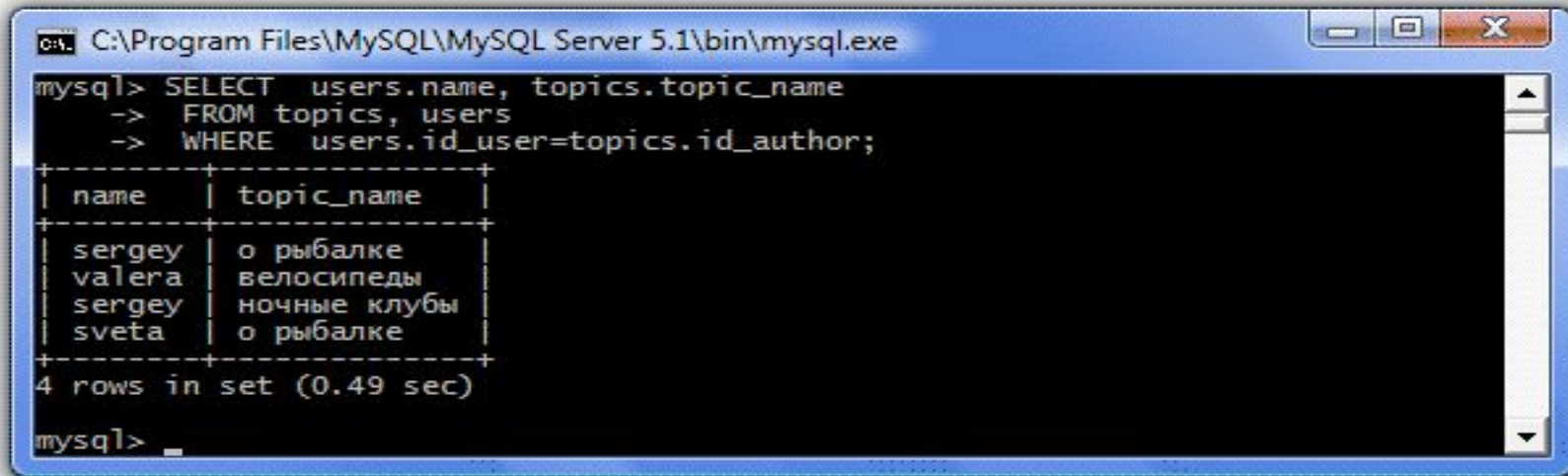
Если имя столбца уникально, то название таблицы можно опустить (как мы делали в примере), но делать это не рекомендуется.

Объединения дают возможность выбирать любую информацию из любых таблиц, причем объединяемых таблиц может быть и три, и четыре, условий для объединения может быть не одно.

# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНЕШНЕЕ ОБЪЕДИНЕНИЕ)

Позволяет выводить все строки одной таблицы и имеющиеся связанные с ними строки из другой таблицы.

Например: нам надо вывести всех пользователей и темы, которые они создавали, если таковые имеются. Если мы воспользуемся внутренним объединением, рассмотренным выше, то получим в итоге следующее:



```
mysql> SELECT users.name, topics.topic_name
-> FROM topics, users
-> WHERE users.id_user=topics.id_author;
+-----+-----+
| name   | topic_name |
+-----+-----+
| sergey | о рыбалке  |
| valera | велосипеды |
| sergey | ночные клубы |
| sveta  | о рыбалке  |
+-----+-----+
4 rows in set (0.49 sec)

mysql>
```

То есть в результирующей таблице есть только те пользователи, которые создавали темы. А нам надо, чтобы выводились все имена.

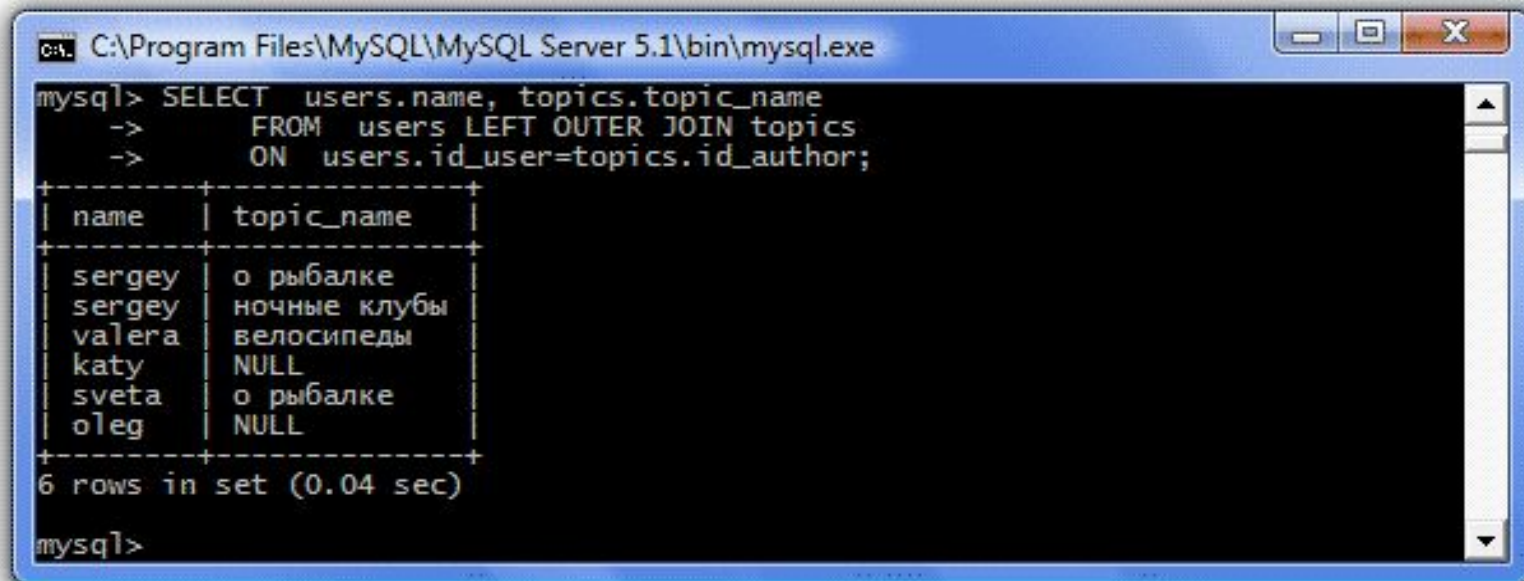


# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНЕШНЕЕ ОБЪЕДИНЕНИЕ)

Немного изменим запрос:

```
SELECT users.name, topics.topic_name  
FROM users LEFT OUTER JOIN topics  
ON users.id_user=topics.id_author;
```

И получим желаемый результат - все пользователи и темы, ими созданные. Если пользователь не создавал тему, но в соответствующем столбце стоит значение NULL.



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe  
mysql> SELECT users.name, topics.topic_name  
-> FROM users LEFT OUTER JOIN topics  
-> ON users.id_user=topics.id_author;  
+-----+-----+  
| name   | topic_name |  
+-----+-----+  
| sergey | о рыбалке  |  
| sergey | ночные клубы |  
| valera  | велосипеды |  
| katy    | NULL       |  
| sveta   | о рыбалке  |  
| oleg    | NULL       |  
+-----+-----+  
6 rows in set (0.04 sec)  
mysql>
```

# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНЕШНЕЕ ОБЪЕДИНЕНИЕ)

В запрос было добавлено ключевое слово - **LEFT OUTER JOIN**, указав тем самым, что из таблицы слева надо взять все строки, и поменяли ключевое слово **WHERE** на **ON**.

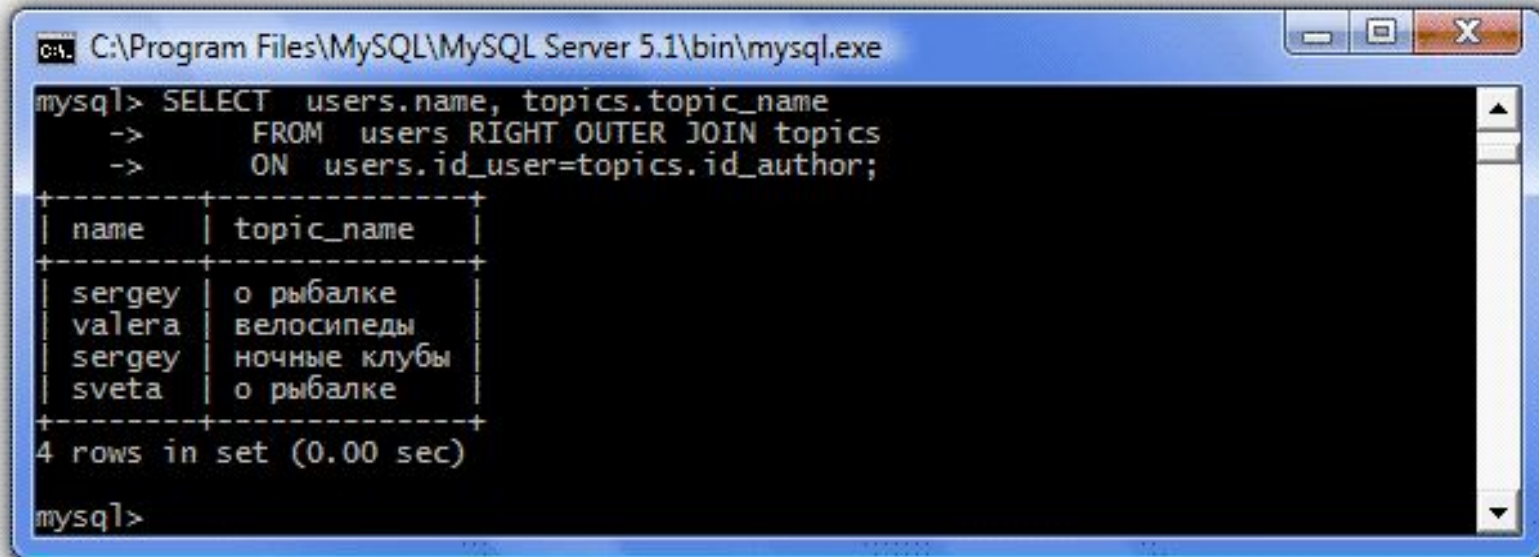
Кроме ключевого слова **LEFT OUTER JOIN** может быть использовано ключевое слово **RIGHT OUTER JOIN**. Тогда будут выбираться все строки из правой таблицы и имеющиеся связанные с ними из левой таблицы. Если написать **FULL OUTER JOIN** произойдет полное внешнее объединение, которое извлечет все строки из обеих таблиц и свяжет между собой те, которые могут быть связаны.

Синтаксис для внешнего объединения следующий:

```
SELECT имя_таблицы_1.имя_столбца, имя_таблицы_2.  
имя_столбца  
    FROM имя_таблицы_1 ТИП ОБЪЕДИНЕНИЯ  
имя_таблицы_2  
    ON условие_объединения;
```

# ОБЪЕДИНЕНИЕ ТАБЛИЦ (ВНЕШНЕЕ ОБЪЕДИНЕНИЕ)

Поменяем в нашем запросе левостороннее объединение на правостороннее



```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe
mysql> SELECT users.name, topics.topic_name
-> FROM users RIGHT OUTER JOIN topics
-> ON users.id_user=topics.id_author;
+-----+-----+
| name  | topic_name |
+-----+-----+
| sergey | о рыбалке  |
| valera | велосипеды |
| sergey | ночные клубы |
| sveta  | о рыбалке  |
+-----+-----+
4 rows in set (0.00 sec)

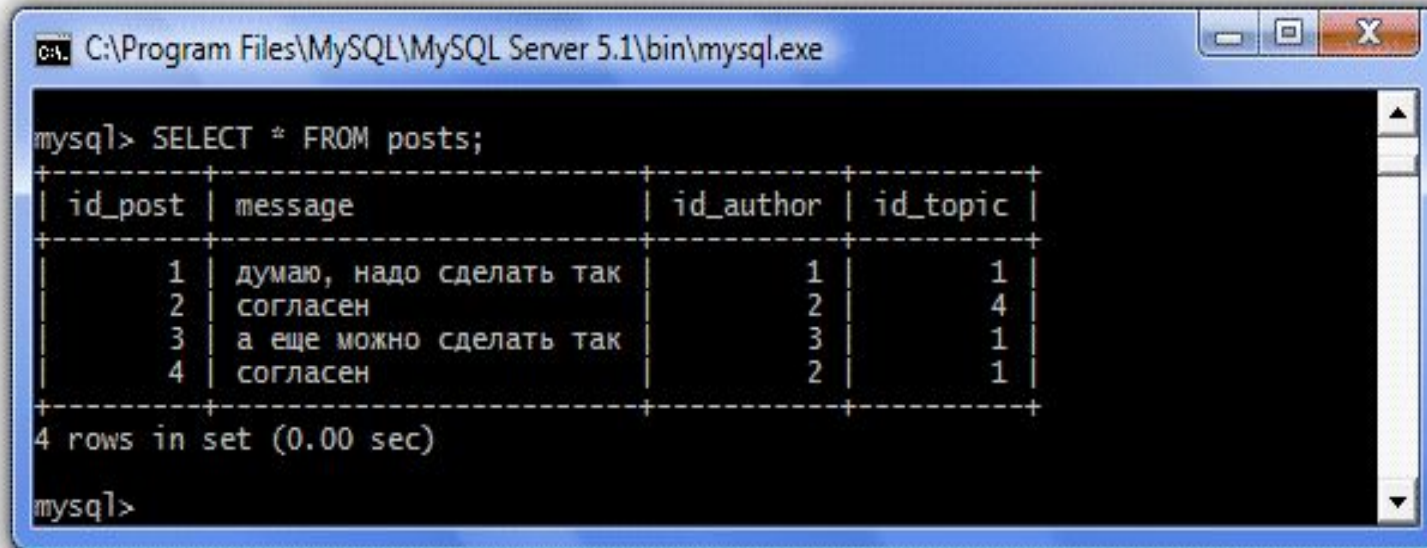
mysql>
```

Теперь есть все темы (все строки из правой таблицы), а пользователи только те, которые темы создавали (т.е. из левой таблицы выбираются только те строки, которые связаны с правой таблицей).

# ГРУППИРОВКА ЗАПИСЕЙ И ФУНКЦИЯ COUNT()

Для того, чтобы посмотреть какие сообщения и в каких темах имеются воспользуемся запросом

```
SELECT * FROM posts;
```



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe
mysql> SELECT * FROM posts;
+----+-----+-----+-----+
| id_post | message                | id_author | id_topic |
+----+-----+-----+-----+
| 1       | думаю, надо сделать так | 1         | 1        |
| 2       | согласен                | 2         | 4        |
| 3       | а еще можно сделать так | 3         | 1        |
| 4       | согласен                | 2         | 1        |
+----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

# ГРУППИРОВКА ЗАПИСЕЙ И ФУНКЦИЯ COUNT()

Для того, чтобы узнать сколько сообщений имеется на форуме можно воспользоваться встроенной функцией COUNT().

Эта функция подсчитывает число строк. Причем, если в качестве аргумента этой функции выступает \*, то подсчитываются все строки таблицы. А если в качестве аргумента указывается имя столбца, то подсчитываются только те строки, которые имеют значение в указанном столбце.

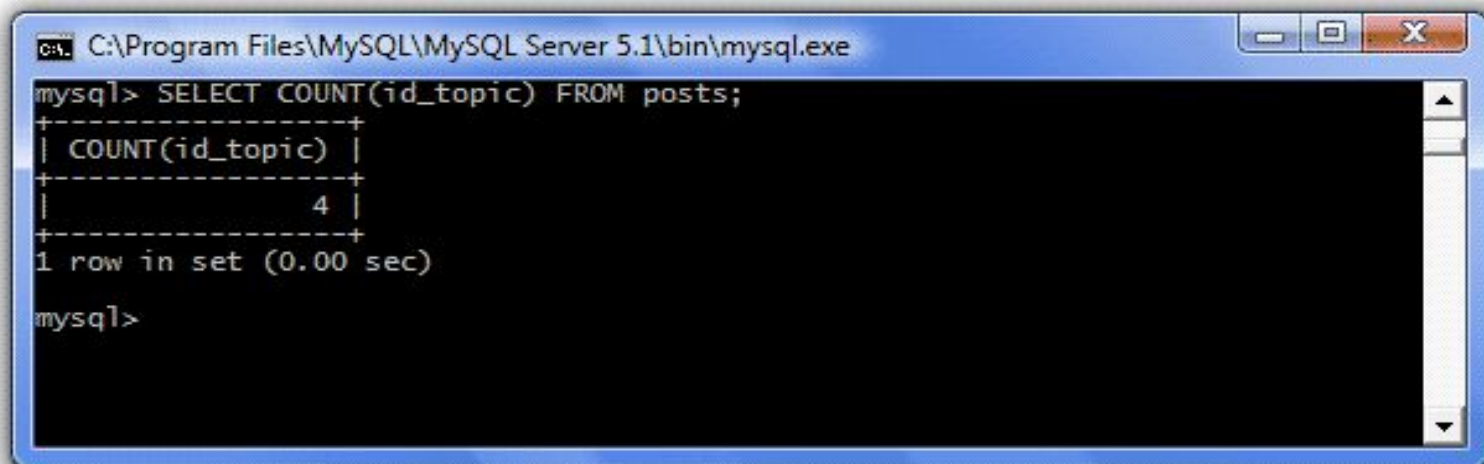
В нашем примере оба аргумента дадут одинаковый результат, т.к. все столбцы таблицы имеют тип NOT NULL.

Напишем запрос, используя в качестве аргумента столбец id\_topic:

```
SELECT COUNT(id_topic) FROM posts;
```

# ГРУППИРОВКА ЗАПИСЕЙ И ФУНКЦИЯ COUNT()

Получим:



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe
mysql> SELECT COUNT(id_topic) FROM posts;
+-----+
| COUNT(id_topic) |
+-----+
|                4 |
+-----+
1 row in set (0.00 sec)

mysql>
```

Т.е., в наших темах имеется 4 сообщения

# ГРУППИРОВКА ЗАПИСЕЙ И ФУНКЦИЯ COUNT()

Для того чтобы узнать сколько сообщений имеется в каждой теме, нужно сгруппировать наши сообщения по темам и вычислить для каждой группы количество сообщений. Для группировки в SQL используется оператор GROUP BY.

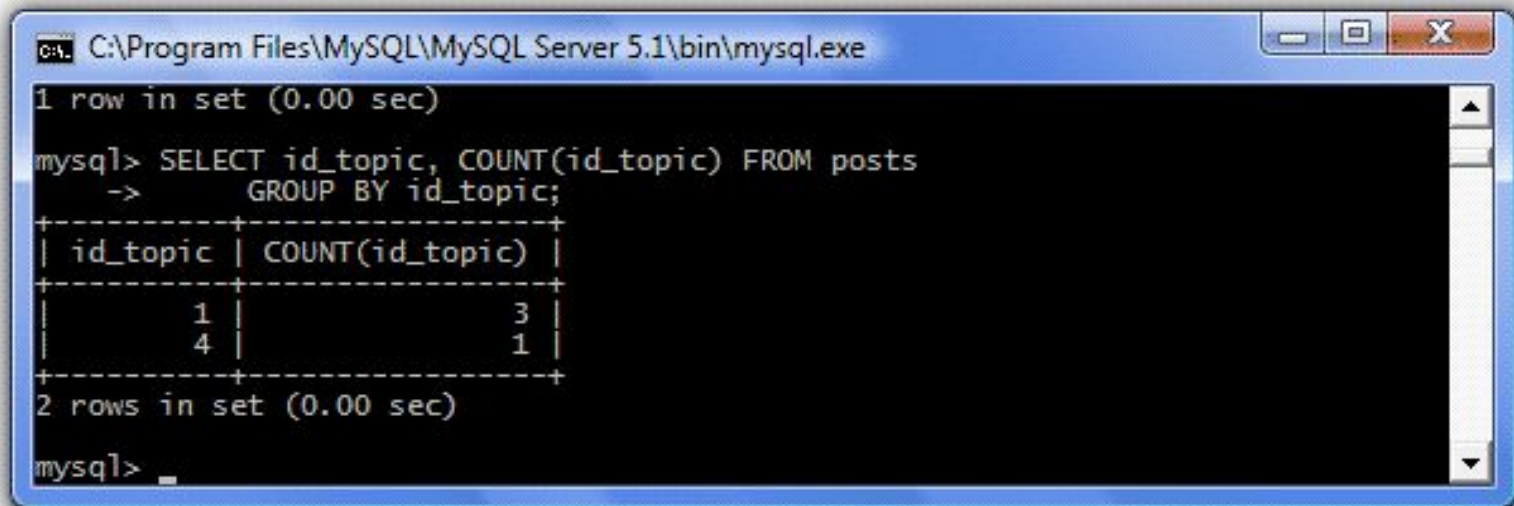
```
SELECT id_topic, COUNT(id_topic) FROM posts  
GROUP BY id_topic;
```

Оператор GROUP BY указывает СУБД сгруппировать данные по столбцу id\_topic (т.е. каждая тема - отдельная группа) и для каждой группы подсчитать количество строк:

Для оператора GROUP BY можно задавать такие же условия, как и для оператора WHERE, только WHERE фильтрует строки, а HAVING - группы.

# ГРУППИРОВКА ЗАПИСЕЙ И ФУНКЦИЯ COUNT()

Результат выполнения запроса:



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe

1 row in set (0.00 sec)

mysql> SELECT id_topic, COUNT(id_topic) FROM posts
        ->     GROUP BY id_topic;
+-----+-----+
| id_topic | COUNT(id_topic) |
+-----+-----+
|         1 |                 3 |
|         4 |                 1 |
+-----+-----+
2 rows in set (0.00 sec)

mysql> _
```

Если бы в поле `id_topic` были возможны отсутствия значений, то такие строки были бы объединены в отдельную группу со значением `NULL`.

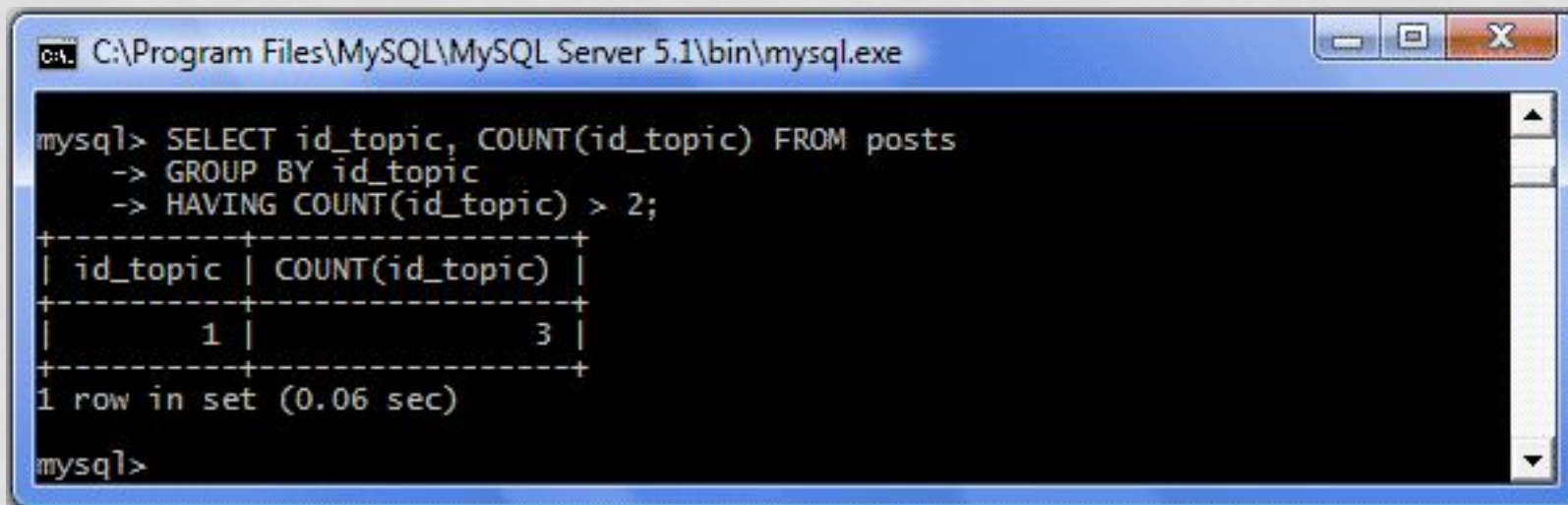


# ГРУППИРОВКА ЗАПИСЕЙ И ФУНКЦИЯ COUNT()

Предположим, что нас интересуют только те группы, в которых больше двух сообщений. В обычном запросе мы указали бы условие с помощью оператора `WHERE`, но этот оператор умеет работать только со строками, а для групп те же функции выполняет оператор **HAVING**:

```
SELECT id_topic, COUNT(id_topic) FROM posts  
GROUP BY id_topic  
HAVING COUNT(id_topic) > 2;
```

Результат:



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe  
mysql> SELECT id_topic, COUNT(id_topic) FROM posts  
-> GROUP BY id_topic  
-> HAVING COUNT(id_topic) > 2;  
+-----+-----+  
| id_topic | COUNT(id_topic) |  
+-----+-----+  
|         1 |                 3 |  
+-----+-----+  
1 row in set (0.06 sec)  
mysql>
```