

ЗАДАНИЕ ДЛЯ САМОСТОЯТЕЛЬНОЙ ПРАКТИЧЕСКОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ «СИСТЕМЫ ОБРАБОТКИ И ХРАНЕНИЯ ДАННЫХ»

Изучить и создать выборку и сортировку данных. Изучить и применить операторы для изменения данных в таблицах.

Результат работы в виде отчета должен содержать:

- снимки экрана (скриншоты) выборки данных по различным параметрам (по каждому оператору);
- снимок экрана (скриншоты) сортировки данных;
- снимки экрана (скриншоты) применения операторов изменения данных в таблицах Вашей базы данных;
- добавить все практические работы по SQL в итоговых отчет.

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ.

Выборка и сортировка данных

Продолжаем изучать базовые знания SQL на пример БД forum есть три таблицы: users (пользователи), topics (темы) и posts (сообщения).

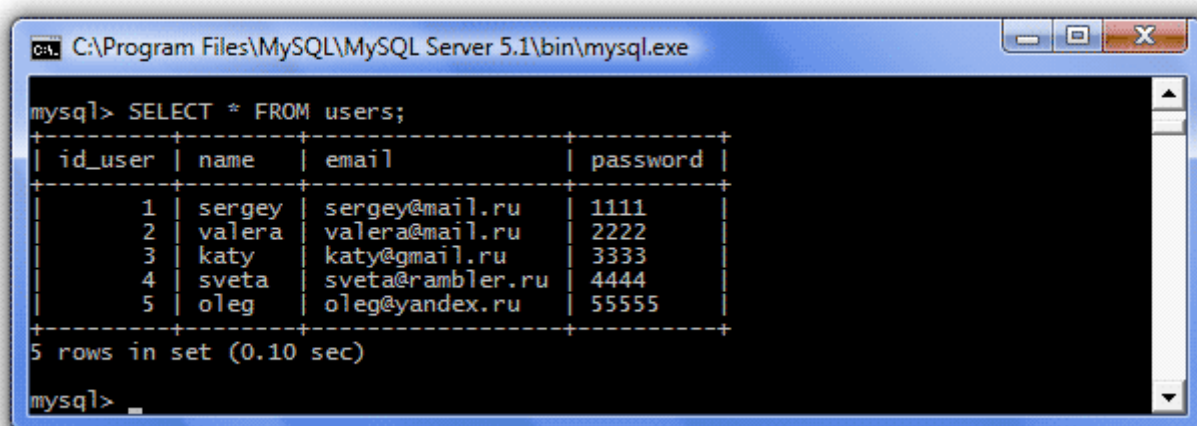
Если необходимо посмотреть, какие данные в них содержатся, то для этого в SQL существует оператор `SELECT`. Синтаксис его использования следующий:

1. `SELECT что_выбрать FROM откуда_выбрать;`

Вместо «что_выбрать» нужно указать либо имя столбца, значения которого хотим увидеть, либо имена нескольких столбцов через запятую, либо символ звездочки (*), означающий выбор всех столбцов таблицы. Вместо «откуда_выбрать» следует указать имя таблицы.

Давайте сначала посмотрим все столбцы из таблицы users:

1. `SELECT * FROM users;`



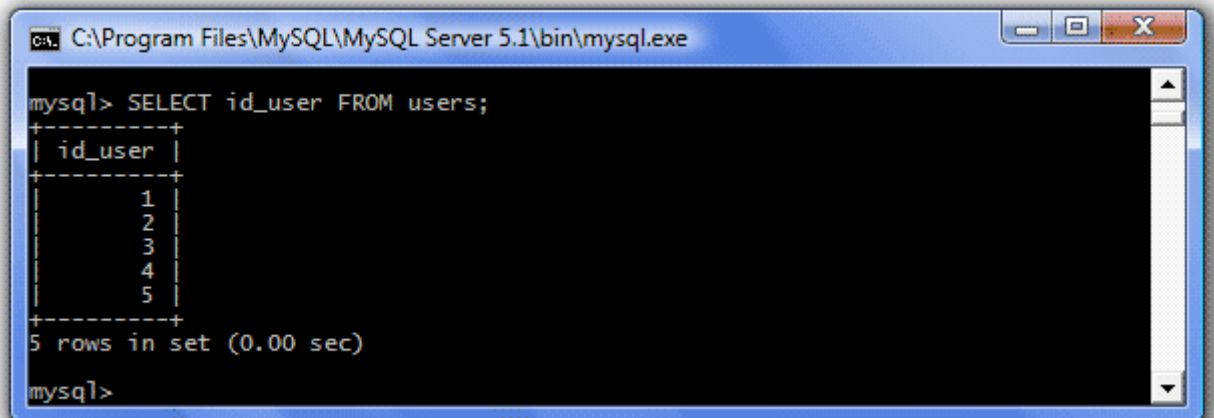
```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe
mysql> SELECT * FROM users;
+----+-----+-----+-----+
| id_user | name  | email          | password |
+----+-----+-----+-----+
| 1       | sergey | sergey@mail.ru | 1111     |
| 2       | valera | valera@mail.ru | 2222     |
| 3       | katy   | katy@gmail.ru  | 3333     |
| 4       | sveta  | sveta@rambler.ru | 4444     |
| 5       | oleg   | oleg@yandex.ru | 5555     |
+----+-----+-----+-----+
5 rows in set (0.10 sec)

mysql> _
```

Вот и все данные, которые были внесены в эту таблицу. Но предположим, что нужно посмотреть только столбец `id_user` (например, на предыдущей практической нам надо было для заполнения таблицы topics (темы) знать,

какие `id_user` есть в таблице `users`). Для этого в запросе укажем имя этого столбца:

```
1. SELECT id_user FROM users;
```



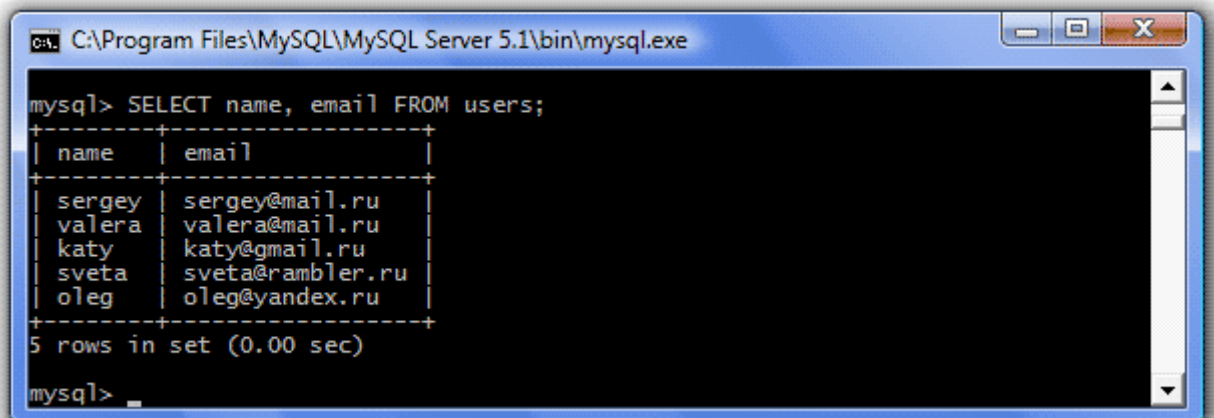
```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> SELECT id_user FROM users;
+----+
| id_user |
+----+
| 1       |
| 2       |
| 3       |
| 4       |
| 5       |
+----+
5 rows in set (0.00 sec)

mysql>
```

Если нужно посмотреть, например, имена и e-mail наших пользователей, то перечислим интересующие столбцы через запятую:

```
1. SELECT name, email FROM users;
```



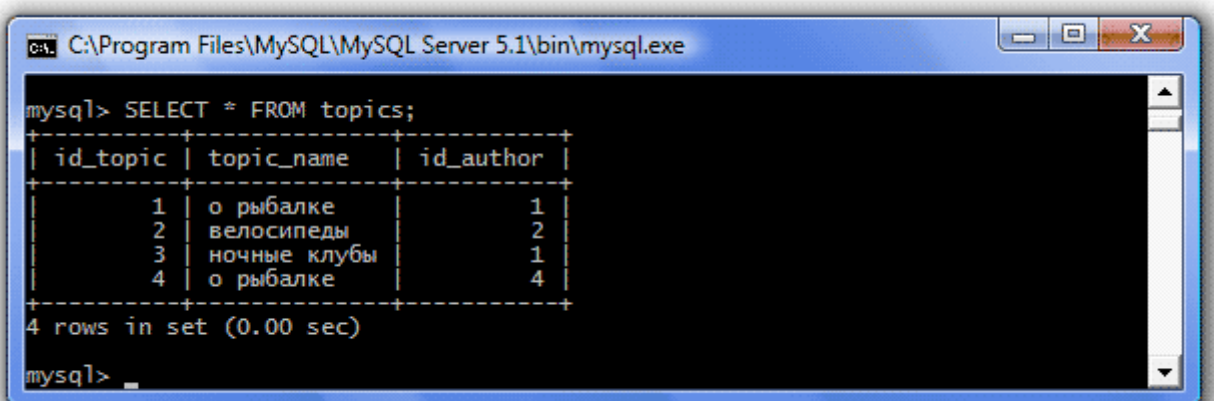
```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> SELECT name, email FROM users;
+-----+-----+
| name  | email          |
+-----+-----+
| sergey | sergey@mail.ru |
| valera | valera@mail.ru |
| katy   | katy@gmail.ru  |
| sveta  | sveta@rambler.ru |
| oleg   | oleg@yandex.ru |
+-----+-----+
5 rows in set (0.00 sec)

mysql> _
```

Аналогично, можно посмотреть, какие данные содержат и другие наши таблицы. Давайте посмотрим, какие существуют темы:

```
1. SELECT * FROM topics;
```



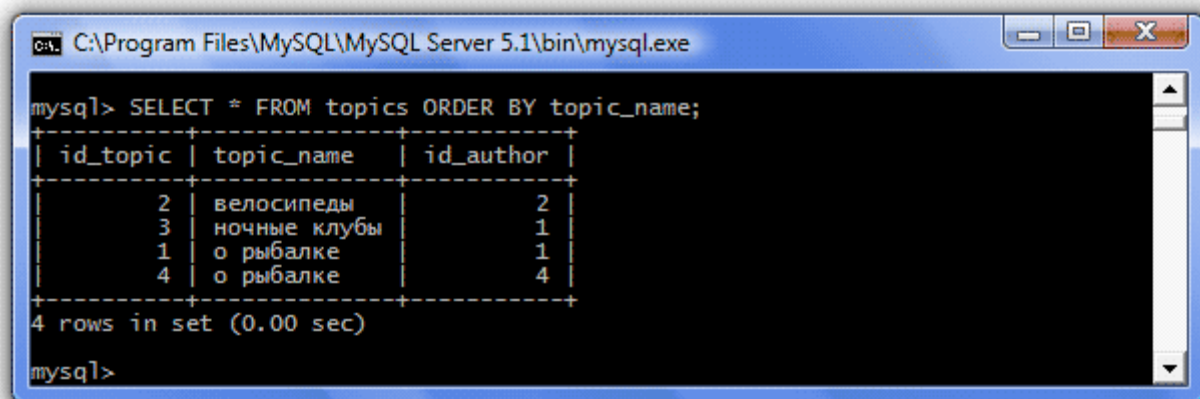
```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> SELECT * FROM topics;
+----+-----+-----+
| id_topic | topic_name | id_author |
+----+-----+-----+
| 1       | о рыбалке  | 1         |
| 2       | велосипеды | 2         |
| 3       | ночные клубы | 1         |
| 4       | о рыбалке  | 4         |
+----+-----+-----+
4 rows in set (0.00 sec)

mysql> _
```

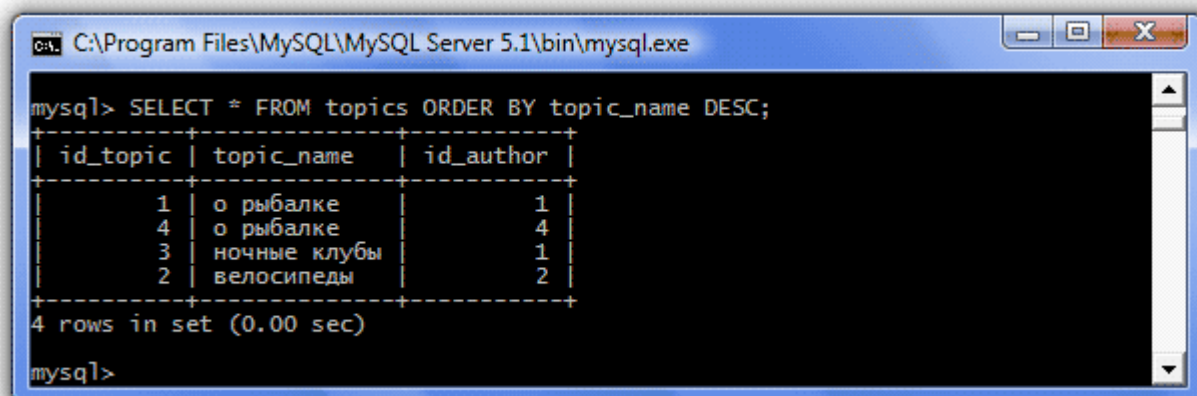
Сейчас в таблице всего 4 темы, а если их будет 100? Хотелось бы, чтобы они выводились, например, по алфавиту. Для этого в SQL существует ключевое слово **ORDER BY** после которого указывается имя столбца по которому будет происходить сортировка. Синтаксис следующий:

```
1. SELECT имя_столбца FROM имя_таблицы ORDER BY  
   имя_столбца_сортировки;
```



```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe  
mysql> SELECT * FROM topics ORDER BY topic_name;  
+-----+-----+-----+  
| id_topic | topic_name | id_author |  
+-----+-----+-----+  
| 2 | велосипеды | 2 |  
| 3 | ночные клубы | 1 |  
| 1 | о рыбалке | 1 |  
| 4 | о рыбалке | 4 |  
+-----+-----+-----+  
4 rows in set (0.00 sec)  
mysql>
```

По умолчанию сортировка идет по возрастанию, но это можно изменить, добавив ключевое слово **DESC**.



```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe  
mysql> SELECT * FROM topics ORDER BY topic_name DESC;  
+-----+-----+-----+  
| id_topic | topic_name | id_author |  
+-----+-----+-----+  
| 1 | о рыбалке | 1 |  
| 4 | о рыбалке | 4 |  
| 3 | ночные клубы | 1 |  
| 2 | велосипеды | 2 |  
+-----+-----+-----+  
4 rows in set (0.00 sec)  
mysql>
```

Теперь данные отсортированы в порядке по убыванию.

Сортировку можно производить сразу по нескольким столбцам. Например, следующий запрос отсортирует данные по столбцу `topic_name`, и если в этом столбце будет несколько одинаковых строк, то в столбце `id_author` будет осуществлена сортировка по убыванию:

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

mysql> SELECT * FROM topics ORDER BY topic_name DESC, id_author DESC;
+-----+-----+-----+
| id_topic | topic_name | id_author |
+-----+-----+-----+
| 4       | о рыбалке | 4         |
| 1       | о рыбалке | 1         |
| 3       | ночные клубы | 1       |
| 2       | велосипеды | 2         |
+-----+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

Сравните результат с результатом предыдущего запроса.

Очень часто бывает, что все информация из таблицы не нужна. Например, необходимо узнать, какие темы были созданы пользователем sveta (id=4). Для этого в SQL есть ключевое слово WHERE, синтаксис у такого запроса следующий:

1. SELECT имя_столбца FROM имя_таблицы WHERE условие;

Для нашего примера условием является идентификатор пользователя, т.е. нужны только те строки, в столбце id_author которых стоит 4 (идентификатор пользователя sveta):

1. SELECT * FROM topics WHERE id_author=4;

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

mysql> SELECT * FROM topics WHERE id_author=4;
+-----+-----+-----+
| id_topic | topic_name | id_author |
+-----+-----+-----+
| 4       | о рыбалке | 4         |
+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

Или нужно узнать, кто создал тему «велосипеды»:

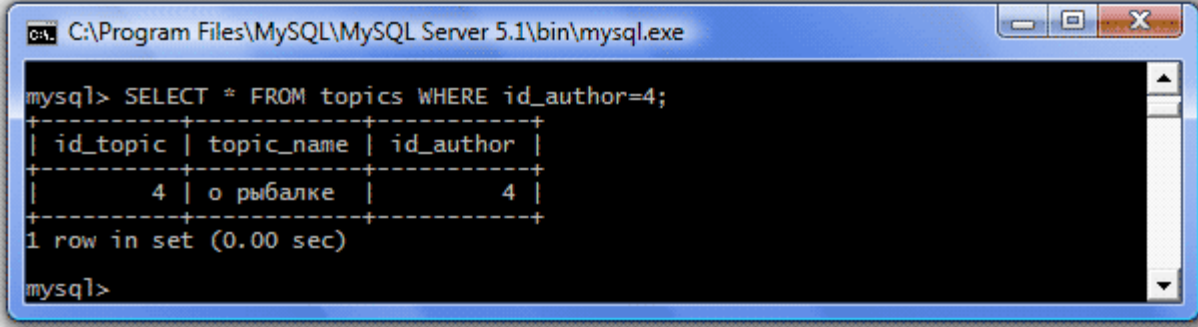
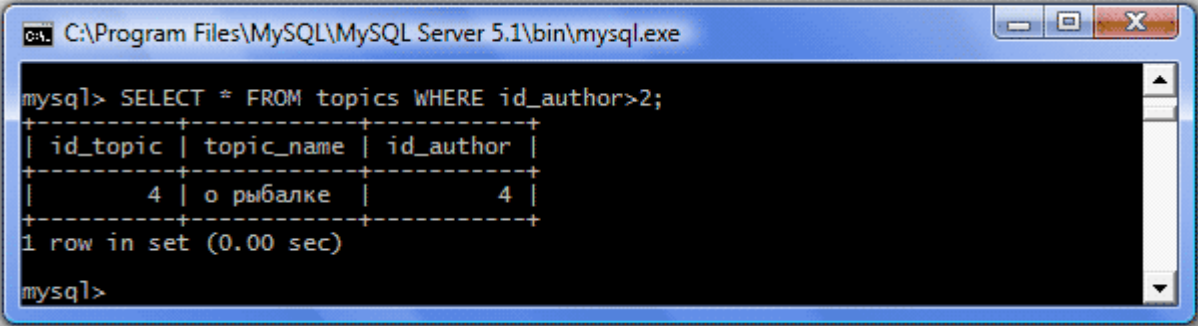
```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> SELECT * FROM topics WHERE topic_name='велосипеды';
+-----+-----+-----+
| id_topic | topic_name | id_author |
+-----+-----+-----+
| 2       | велосипеды | 2         |
+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

Конечно, было бы удобнее, чтобы вместо id автора, выводилось его имя, но имена хранятся в другой таблице. В последующих практических узнаем, как

выбирать данные из нескольких таблиц. А пока узнаем, какие условия можно задавать, используя ключевое слово `WHERE`.

Оператор	Описание
<p>= (равно)</p>	<p>Отбираются значения равные указанному Пример: SELECT * FROM topics WHERE id_author=4; Результат:</p>  <p>The screenshot shows a MySQL command window with the following text:</p> <pre> C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe mysql> SELECT * FROM topics WHERE id_author=4; +-----+-----+-----+ id_topic topic_name id_author +-----+-----+-----+ 4 о рыбалке 4 +-----+-----+-----+ 1 row in set (0.00 sec) mysql> </pre>
<p>> (больше)</p>	<p>Отбираются значения больше указанного Пример: SELECT * FROM topics WHERE id_author>2; Результат:</p>  <p>The screenshot shows a MySQL command window with the following text:</p> <pre> C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe mysql> SELECT * FROM topics WHERE id_author>2; +-----+-----+-----+ id_topic topic_name id_author +-----+-----+-----+ 4 о рыбалке 4 +-----+-----+-----+ 1 row in set (0.00 sec) mysql> </pre>

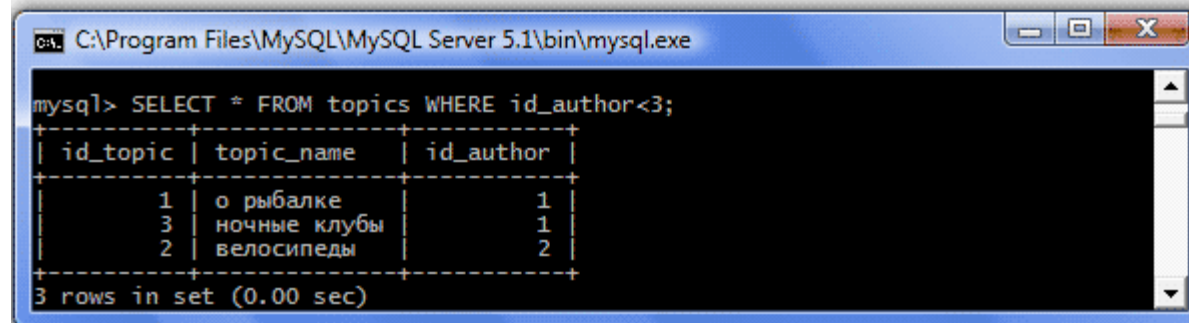
< (меньше)

Отбираются значения меньше указанного

Пример:

```
SELECT * FROM topics WHERE id_author<3;
```

Результат:



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

mysql> SELECT * FROM topics WHERE id_author<3;
+----+-----+-----+
| id_topic | topic_name | id_author |
+----+-----+-----+
| 1       | о рыбалке  | 1         |
| 3       | ночные клубы | 1         |
| 2       | велосипеды | 2         |
+----+-----+-----+
3 rows in set (0.00 sec)
```

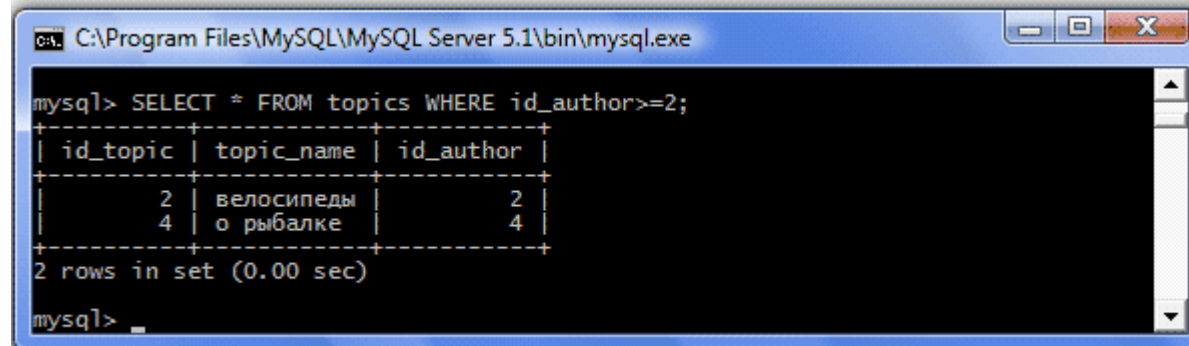
>= (больше или равно)

Отбираются значения большие и равные указанному

Пример:

```
SELECT * FROM topics WHERE id_author>=2;
```

Результат:



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

mysql> SELECT * FROM topics WHERE id_author>=2;
+----+-----+-----+
| id_topic | topic_name | id_author |
+----+-----+-----+
| 2       | велосипеды | 2         |
| 4       | о рыбалке  | 4         |
+----+-----+-----+
2 rows in set (0.00 sec)

mysql> _
```

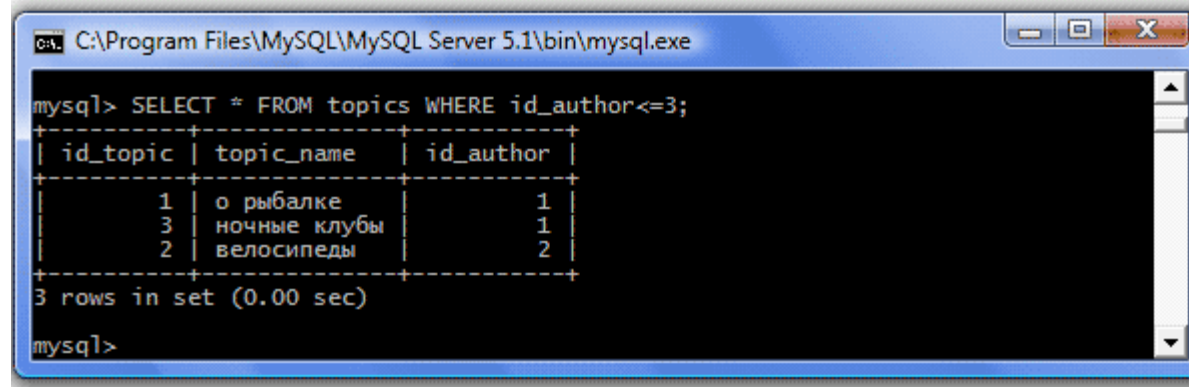
<= (меньше или
равно)

Отбираются значения меньше и равные указанному

Пример:

SELECT * FROM topics WHERE id_author<=3;

Результат:



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

mysql> SELECT * FROM topics WHERE id_author<=3;
+----+-----+-----+
| id_topic | topic_name | id_author |
+----+-----+-----+
| 1 | о рыбалке | 1 |
| 3 | ночные клубы | 1 |
| 2 | велосипеды | 2 |
+----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

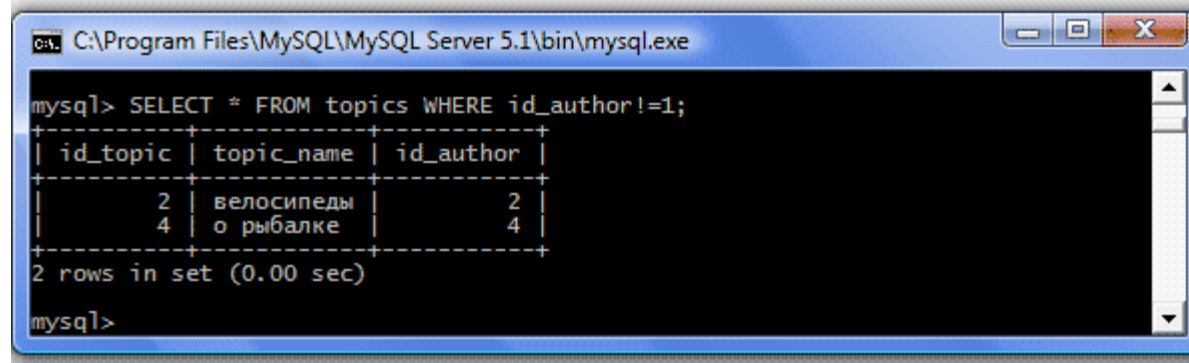
!= (не равно)

Отбираются значения не равные указанному

Пример:

SELECT * FROM topics WHERE id_author!=1;

Результат:



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

mysql> SELECT * FROM topics WHERE id_author!=1;
+----+-----+-----+
| id_topic | topic_name | id_author |
+----+-----+-----+
| 2 | велосипеды | 2 |
| 4 | о рыбалке | 4 |
+----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

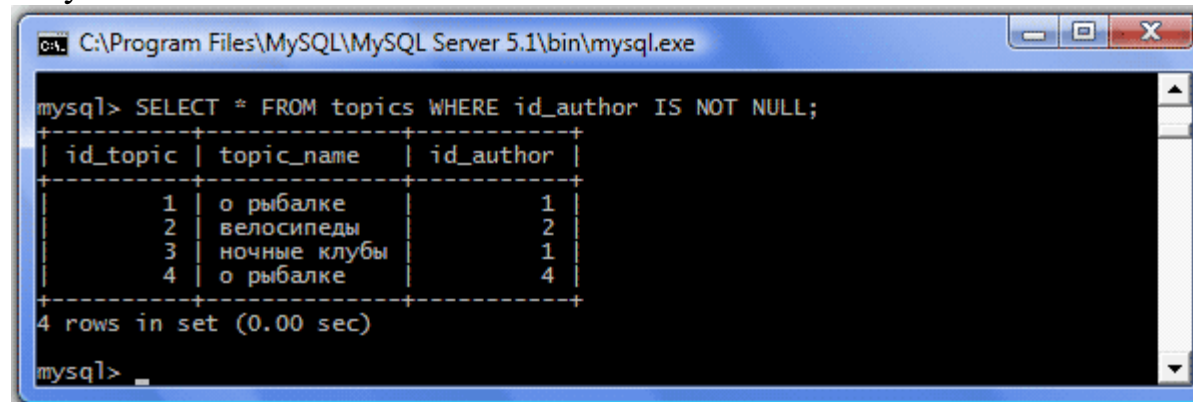

IS NOT NULL

Отбираются строки, имеющие значения в указанном поле

Пример:

```
SELECT * FROM topics WHERE id_author IS NOT NULL;
```

Результат:



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

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

mysql> SELECT * FROM topics WHERE id_author IS NOT NULL;
+----+-----+-----+
| id_topic | topic_name | id_author |
+----+-----+-----+
| 1       | о рыбалке  | 1         |
| 2       | велосипеды | 2         |
| 3       | ночные клубы | 1        |
| 4       | о рыбалке  | 4         |
+----+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

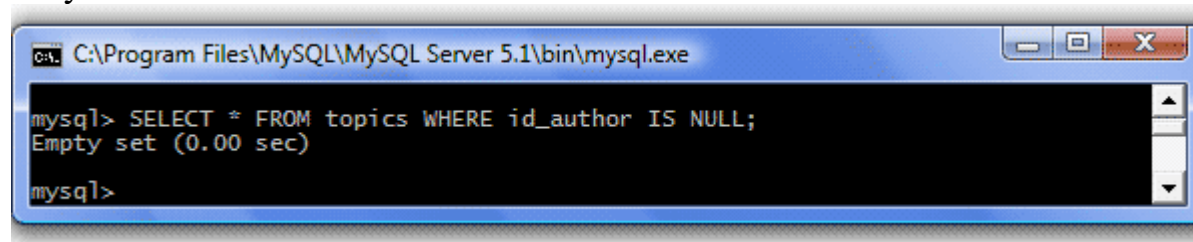
IS NULL

Отбираются строки, не имеющие значения в указанном поле

Пример:

```
SELECT * FROM topics WHERE id_author IS NULL;
```

Результат:



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

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

mysql> SELECT * FROM topics WHERE id_author IS NULL;
Empty set (0.00 sec)

mysql>
```

Empty set — нет таких строк.

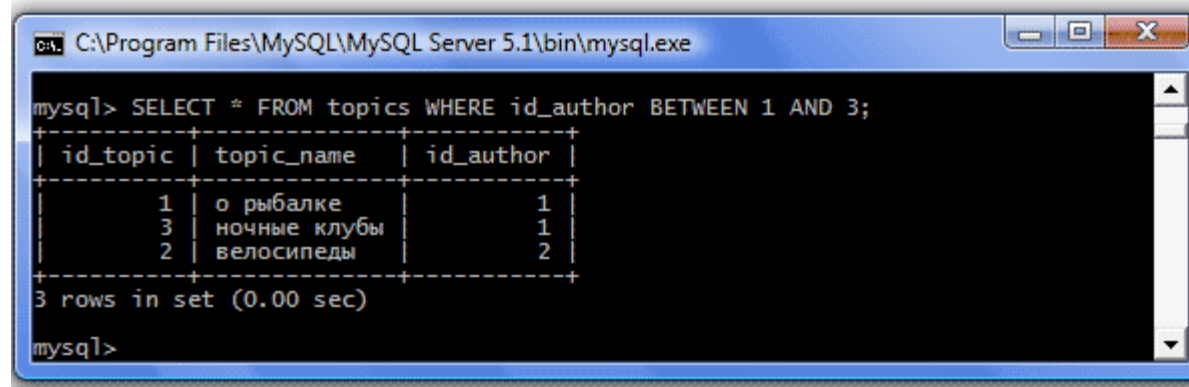
BETWEEN
(между)

Отбираются значения, находящиеся между указанными

Пример:

```
SELECT * FROM topics WHERE id_author BETWEEN 1 AND 3;
```

Результат:



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

mysql> SELECT * FROM topics WHERE id_author BETWEEN 1 AND 3;
+----+-----+-----+
| id_topic | topic_name | id_author |
+----+-----+-----+
| 1       | о рыбалке  | 1         |
| 3       | ночные клубы | 1         |
| 2       | велосипеды | 2         |
+----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

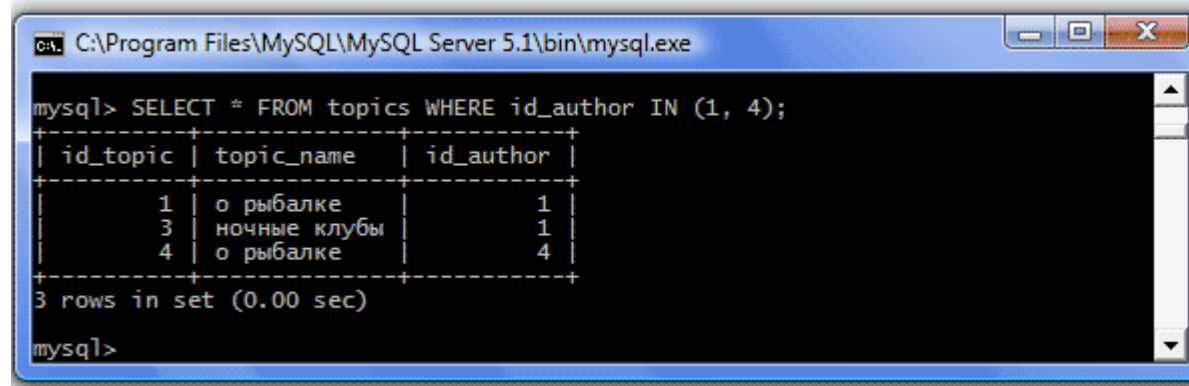
IN (значение
содержится)

Отбираются значения, соответствующие указанным

Пример:

```
SELECT * FROM topics WHERE id_author IN (1, 4);
```

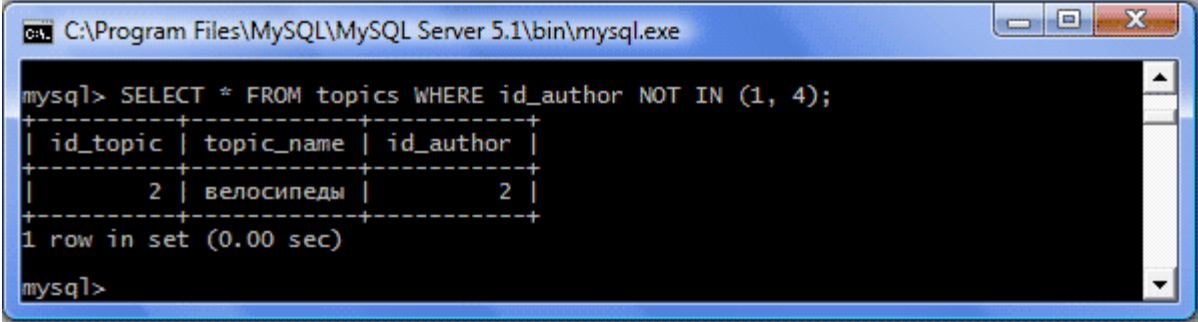
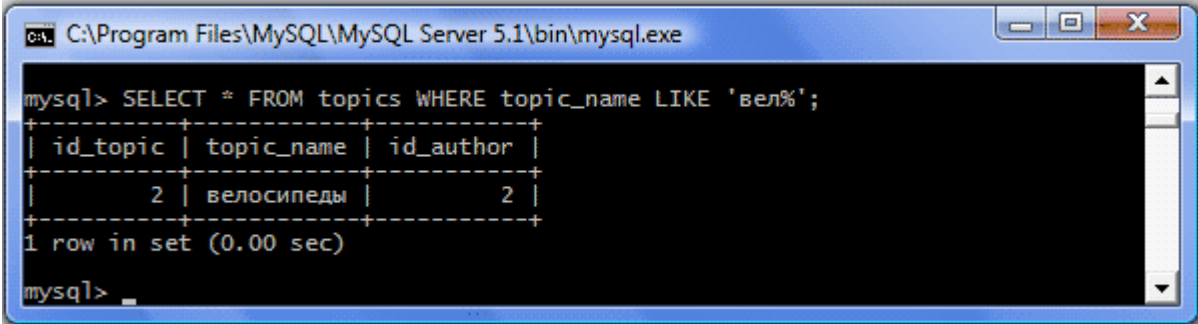
Результат:



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

mysql> SELECT * FROM topics WHERE id_author IN (1, 4);
+----+-----+-----+
| id_topic | topic_name | id_author |
+----+-----+-----+
| 1       | о рыбалке  | 1         |
| 3       | ночные клубы | 1         |
| 4       | о рыбалке  | 4         |
+----+-----+-----+
3 rows in set (0.00 sec)

mysql>
```

<p>NOT (значение содержится)</p> <p>IN не</p>	<p>Отбираются значения, кроме указанных</p> <p>Пример: SELECT * FROM topics WHERE id_author NOT IN (1, 4);</p> <p>Результат:</p> 
<p>LIKE (соответствие)</p>	<p>Отбираются значения, соответствующие образцу</p> <p>Пример: SELECT * FROM topics WHERE topic_name LIKE 'вел%';</p> <p>Результат:</p>  <p>Возможные метасимволы оператора LIKE будут рассмотрены ниже.</p>

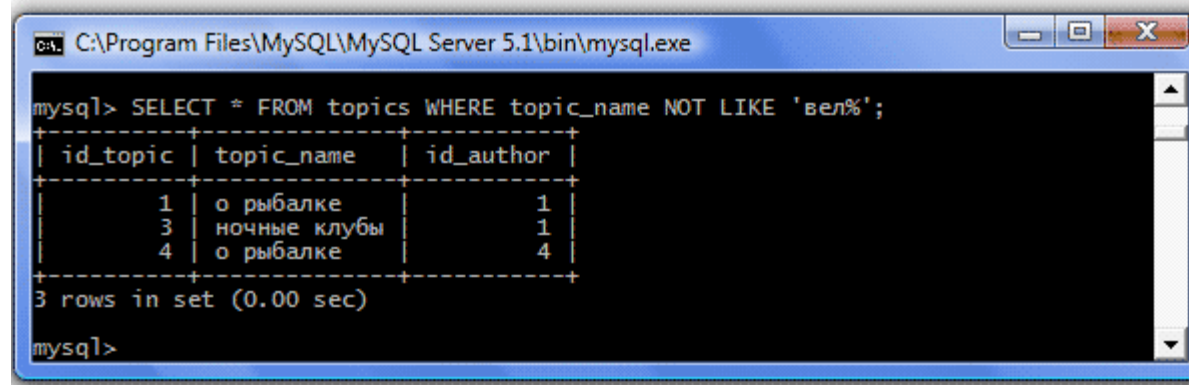
NOT LIKE (не
соответствие)

Отбираются значения, не соответствующие образцу

Пример:

```
SELECT * FROM topics WHERE topic_name NOT LIKE 'вел%';
```

Результат:



The screenshot shows a MySQL command window titled "C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe". The prompt is "mysql>". The user has entered the query: "SELECT * FROM topics WHERE topic_name NOT LIKE 'вел%';". The result is displayed in a table format with columns "id_topic", "topic_name", and "id_author". The table contains three rows of data. Below the table, it says "3 rows in set (0.00 sec)". The prompt "mysql>" is visible at the bottom.

id_topic	topic_name	id_author
1	о рыбалке	1
3	ночные клубы	1
4	о рыбалке	4

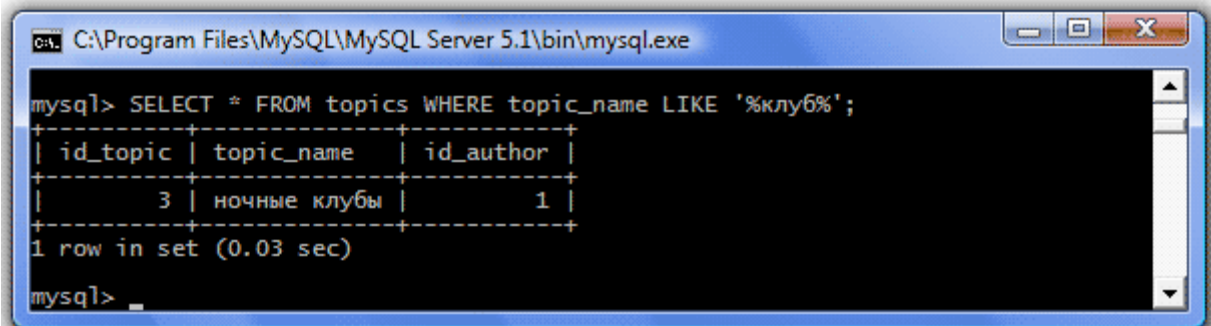
3 rows in set (0.00 sec)

mysql>

Метасимволы оператора LIKE

Поиск с использованием метасимволов может осуществляться только в текстовых полях.

Самый распространенный метасимвол — %. Он означает любые символы. Например, если нам надо найти слова, начинающиеся с букв «вел», то напомним LIKE 'вел%', а если хотим найти слова, которые содержат символы «клуб», то напомним LIKE '%клуб%'. Например:



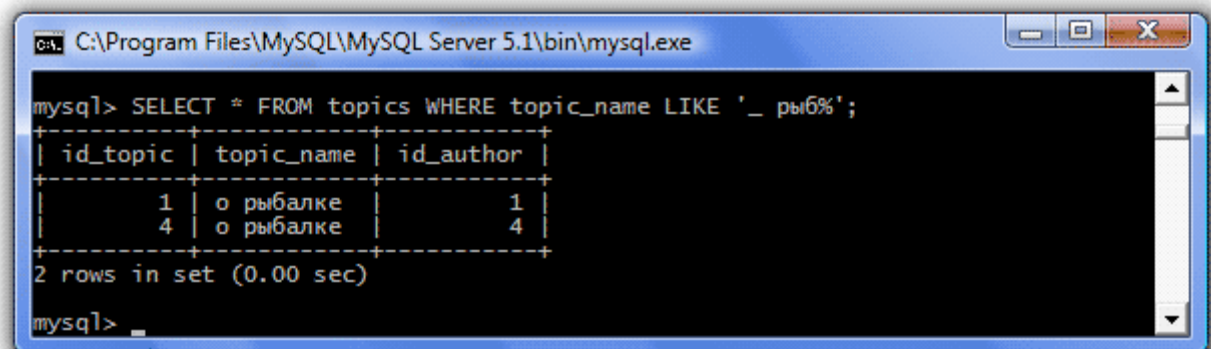
```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe

mysql> SELECT * FROM topics WHERE topic_name LIKE '%клуб%';
+-----+-----+-----+
| id_topic | topic_name | id_author |
+-----+-----+-----+
|        3 | ночные клубы |        1 |
+-----+-----+-----+
1 row in set (0.03 sec)

mysql> _
```

Еще один часто используемый метасимвол — _.

В отличие от %, который обозначает несколько или ни одного символа, нижнее подчеркивание обозначает ровно один символ. Например:



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

mysql> SELECT * FROM topics WHERE topic_name LIKE '_ рыб%';
+-----+-----+-----+
| id_topic | topic_name | id_author |
+-----+-----+-----+
|        1 | о рыбалке |        1 |
|        4 | о рыбалке |        4 |
+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> _
```

Обратите внимание на пробел между метасимволом и «рыб», если его пропустить, то запрос не сработает, т.к. метасимвол _ обозначает ровно один символ, а пробел — это тоже символ.

Изменение данных в таблице

Предположим, что нашему форуму нужны модераторы. Для этого в таблицу users надо добавить столбец с ролью пользователя. Для добавления столбцов в таблицу используется оператор ALTER TABLE — ADD COLUMN.

Добавим столбец role в таблицу users:

1. ALTER TABLE users ADD COLUMN role varchar(20);

Столбец появился в конце таблицы:

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe
Database changed
mysql> ALTER TABLE users ADD COLUMN role varchar(20);
Query OK, 5 rows affected (1.16 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> describe users;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| id_user | int(10)       | NO   | PRI | NULL    | auto_increment |
| name    | varchar(20)   | NO   |     | NULL    |                |
| email   | varchar(50)   | NO   |     | NULL    |                |
| password | varchar(15)   | NO   |     | NULL    |                |
| role    | varchar(20)   | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.36 sec)

mysql> _
```

Для того, чтобы указать местоположение столбца используются ключевые слова: `FIRST` — новый столбец будет первым, и `AFTER` — указывает после какого столбца поместить новый.

Давайте добавим еще два столбца: один — `kol` — количество оставленных сообщений, а другой — `rating` — рейтинг пользователя. Оба столбца вставим после поля `password`:

1. `ALTER TABLE users ADD COLUMN kol int(10) AFTER password, ADD COLUMN rating varchar(20) AFTER kol;`

```
C:\Program Files\MySQL\MySQL Server 5.1\bin\mysql.exe
mysql> ALTER TABLE users ADD COLUMN kol int(10) AFTER password,
-> ADD COLUMN rating varchar(20) AFTER kol;
Query OK, 5 rows affected (0.25 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> describe users;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| id_user | int(10)       | NO   | PRI | NULL    | auto_increment |
| name    | varchar(20)   | NO   |     | NULL    |                |
| email   | varchar(50)   | NO   |     | NULL    |                |
| password | varchar(15)   | NO   |     | NULL    |                |
| kol     | int(10)       | YES  |     | NULL    |                |
| rating  | varchar(20)   | YES  |     | NULL    |                |
| role    | varchar(20)   | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
7 rows in set (0.10 sec)

mysql> _
```

Теперь надо назначить роль модератора какому-нибудь пользователю, пусть это будет `sergey` с `id=1`. Для обновления уже существующих данных служит оператор `UPDATE`.

Давайте сделаем Сергея модератором:

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

mysql> select * from users;
+-----+-----+-----+-----+-----+-----+-----+
| id_user | name  | email          | password | kol | rating | role      |
+-----+-----+-----+-----+-----+-----+-----+
| 1       | sergey | sergey@mail.ru | 1111     | NULL | NULL    | модератор |
| 2       | valera | valera@mail.ru | 2222     | NULL | NULL    | NULL      |
| 3       | katy   | katy@gmail.ru  | 3333     | NULL | NULL    | NULL      |
| 4       | sveta  | sveta@rambler.ru | 4444     | NULL | NULL    | NULL      |
| 5       | oleg   | oleg@yandex.ru | 55555    | NULL | NULL    | NULL      |
+-----+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

mysql>
```

Изменять данные можно и сразу в нескольких строках и во всей таблице. Например, решили давать рейтинг в зависимости от количества оставленных пользователем сообщений. Давайте в нашу таблицу сначала внесем значения столбца kol:

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

mysql> UPDATE users SET kol=50
-> WHERE id_user=1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> UPDATE users SET kol=30
-> WHERE id_user=2;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> UPDATE users SET kol=45
-> WHERE id_user=3;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> UPDATE users SET kol=20
-> WHERE id_user=4;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> UPDATE users SET kol=2
-> WHERE id_user=5;
Query OK, 1 row affected (0.02 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> select * from users;
+-----+-----+-----+-----+-----+-----+-----+
| id_user | name  | email          | password | kol | rating | role      |
+-----+-----+-----+-----+-----+-----+-----+
| 1       | sergey | sergey@mail.ru | 1111     | 50  | NULL    | модератор |
| 2       | valera | valera@mail.ru | 2222     | 30  | NULL    | NULL      |
| 3       | katy   | katy@gmail.ru  | 3333     | 45  | NULL    | NULL      |
| 4       | sveta  | sveta@rambler.ru | 4444     | 20  | NULL    | NULL      |
| 5       | oleg   | oleg@yandex.ru | 55555    | 2   | NULL    | NULL      |
+-----+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

mysql>
```

А теперь зададим рейтинг Профи тем, у кого количество сообщений больше 30:


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

mysql> UPDATE users SET rating='Профи'
-> WHERE kol>30;
Query OK, 2 rows affected (0.00 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> select * from users;
+----+-----+-----+-----+-----+-----+-----+
| id_user | name  | email          | password | kol | rating | role      |
+----+-----+-----+-----+-----+-----+-----+
| 1      | sergey | sergey@mail.ru | 1111    | 50 | Профи  | модератор |
| 2      | valera | valera@mail.ru | 2222    | 30 | NULL   | NULL      |
| 3      | katy   | katy@gmail.ru  | 3333    | 45 | Профи  | NULL      |
| 4      | sveta  | sveta@rambler.ru | 4444    | 20 | NULL   | NULL      |
| 5      | oleg   | oleg@yandex.ru | 55555   | 2  | NULL   | NULL      |
+----+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

mysql>
```

Данные изменились в двух строках, согласно заданному условию. Понятно, что если в запросе опустить условие, то данные будут обновлены во всех строках таблицы.

Предположим, что не нравится название Рейтинг у нашего столбца, надо переименовать столбец в Репутация — reputation. Для изменения имени существующего столбца используется оператор CHANGE.

Давайте поменяем rating на reputation:

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

mysql> ALTER TABLE users CHANGE rating reputation varchar(20);
Query OK, 5 rows affected (0.15 sec)
Records: 5  Duplicates: 0  Warnings: 0

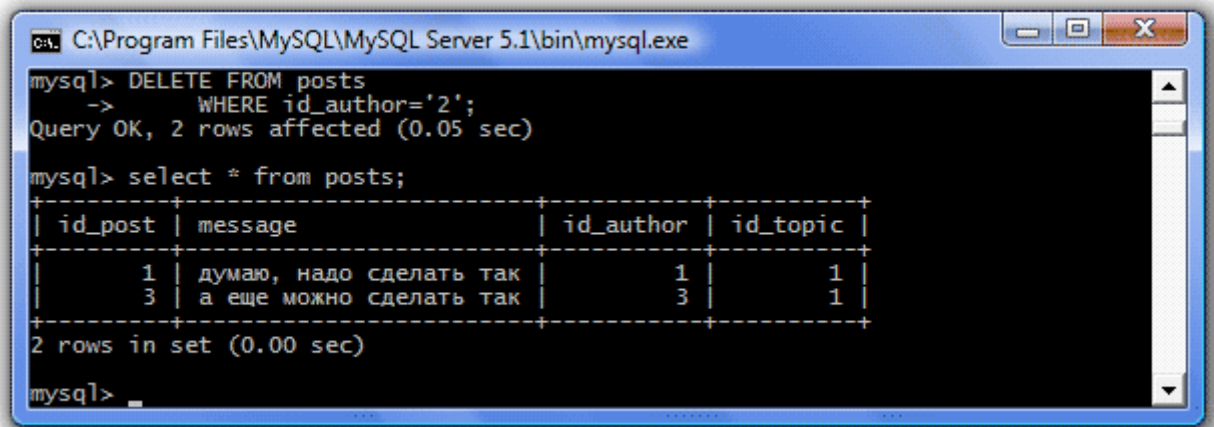
mysql> select * from users;
+----+-----+-----+-----+-----+-----+-----+
| id_user | name  | email          | password | kol | reputation | role      |
+----+-----+-----+-----+-----+-----+-----+
| 1      | sergey | sergey@mail.ru | 1111    | 50 | Профи      | модератор |
| 2      | valera | valera@mail.ru | 2222    | 30 | NULL       | NULL      |
| 3      | katy   | katy@gmail.ru  | 3333    | 45 | Профи      | NULL      |
| 4      | sveta  | sveta@rambler.ru | 4444    | 20 | NULL       | NULL      |
| 5      | oleg   | oleg@yandex.ru | 55555   | 2  | NULL       | NULL      |
+----+-----+-----+-----+-----+-----+-----+
5 rows in set (0.00 sec)

mysql>
```

Обратите внимание, что тип столбца надо указывать даже, если он не меняется. Кстати, если нам понадобится изменить только тип столбца, то будем использовать оператор MODIFY.

Рассмотрим — оператор `DELETE`, который позволяет удалять строки из таблицы.

Давайте из таблицы сообщений удалим те записи, которые оставял пользователь `valera` (`id=2`):



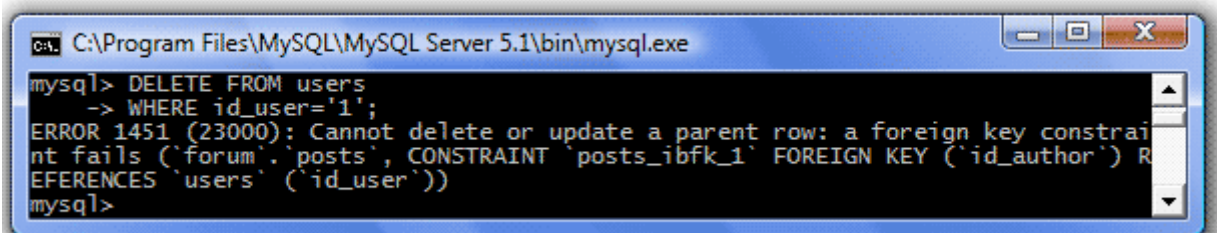
```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe
mysql> DELETE FROM posts
-> WHERE id_author='2';
Query OK, 2 rows affected (0.05 sec)

mysql> select * from posts;
+-----+-----+-----+-----+
| id_post | message                | id_author | id_topic |
+-----+-----+-----+-----+
| 1       | думаю, надо сделать так | 1         | 1         |
| 3       | а еще можно сделать так | 3         | 1         |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

Если опустить условие, то из таблицы будут удалены все данные.

Следует помнить, что данные СУБД даст удалить только в том случае, если они не являются внешними ключами для данных из других таблиц (поддержка целостности БД). Например, если удалить из таблицы `users` пользователя, который оставял сообщения, то нам это не удастся.



```
C:\Program Files\MySQL\MySQL Server 5.1\bin>mysql.exe
mysql> DELETE FROM users
-> WHERE id_user='1';
ERROR 1451 (23000): Cannot delete or update a parent row: a foreign key constraint fails ('forum`.`posts`, CONSTRAINT `posts_ibfk_1` FOREIGN KEY (`id_author`) REFERENCES `users` (`id_user`))
mysql>
```

Сначала надо удалить его сообщения, а уж потом и его самого.