

## **Лабораторная №3**

### **Обработка запросов в базе данных**

#### **Задания №1**

1. Напишите оператор SQL для создания новой базы данных с именем `addressbook`
2. Какой оператор используется для получения информации о таблице? Как используется этот оператор?
3. Как получить список всех баз данных, доступных в системе?
4. Напишите оператор для записи следующих данных в таблицу `employee_data`

Имя: Рудольф  
Фамилия: Курочкин  
Должность: Программист  
Возраст: 34  
Стаж работы в компании: 2  
Зарплата: 95000  
Надбавки: 17000  
email: rudolf@yandex.ru

5. Приведите две формы оператора `SELECT`, которые будут выводить все данные из таблицы `employee_data`.
6. Как извлечь данные столбцов `f_name`, `email` из таблицы `employee_data`?
7. Напишите оператор для вывода данных из столбцов `salary`, `perks` и `yos` таблицы `employee_data`.
8. Как узнать число строк в таблице с помощью оператора `SELECT`?
9. Как извлечь данные столбцов `salary`, `l_name` из таблицы `employee_data`?

#### **Возможные решения:**

1. `create database addressbook;`

или

`CREATE DATABASE addressbook;`

```
mysql> create database adressbook;  
Query OK, 1 row affected (0.01 sec)
```

Примечание: Операторы SQL не различают регистр символов, однако имена таблиц и имена баз данных могут различать регистр символов, в зависимости от используемой операционной системы.

## 2. Оператор `DESCRIBE`, например:

Для выполнения этой команды у вас должна быть предварительно выбрана база данных

```
set adressbook;
```

И создана таблица с названием "`employee_data`" (см. лаб. раб. №2)

```
DESCRIBE employee_data;
```

```
mysql> use adressbook;
Database changed
mysql> create table employee_data;
ERROR 1113 (42000): A table must have at least 1 column
mysql> create table employee_data
  -> (f_name, l_name, title, age, yos, salary, perks, email)
  -> ^C
mysql> CREATE TABLE employee_data
  -> (
  -> emp_id int unsigned not null auto_increment primary key,
  -> f_name varchar(20),
  -> l_name varchar(20),
  -> title varchar(30),
  -> age int,
  -> yos int,
  -> salary int,
  -> perks int,
  -> email varchar(60)
  -> );
Query OK, 0 rows affected (0.03 sec)

mysql> describe employee_data;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra          |
+-----+-----+-----+-----+-----+-----+
| emp_id | int(10) unsigned | NO   | PRI | NULL    | auto_increment |
| f_name | varchar(20)      | YES  |     | NULL    |                |
| l_name | varchar(20)      | YES  |     | NULL    |                |
| title  | varchar(30)      | YES  |     | NULL    |                |
| age    | int(11)          | YES  |     | NULL    |                |
| yos    | int(11)          | YES  |     | NULL    |                |
| salary | int(11)          | YES  |     | NULL    |                |
| perks  | int(11)          | YES  |     | NULL    |                |
| email  | varchar(60)      | YES  |     | NULL    |                |
+-----+-----+-----+-----+-----+-----+
9 rows in set (0.02 sec)
```

## 3. `SHOW DATABASES;` (в приглашении mysql)

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| adressbook |
| information_schema |
| mysql      |
| performance_schema |
| sys        |
+-----+
5 rows in set (0.01 sec)
```

#### 4. INSERT INTO employee\_data

```
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Рудольф", "Курочкин", "программист", 34, 2, 95000, 17000,
"rudolf@yandex.ru");
```

```
mysql> INSERT INTO employee_data
-> (f_name, l_name, title, age, yos, salary, perks, email)
-> values
-> ("Рудольф", "Курочкин", "программист", 34, 2, 95000, 17000, "rudolf@yandex.ru");
Query OK, 1 row affected (0.01 sec)
```

Примечание: Текстовые строки заключаются в кавычки.

5. SELECT emp\_id, f\_name, l\_name, title, age, yos, salary, perks, email from employee\_data;

```
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp_id | f_name      | l_name      | title      | age | yos | salary | perks | email      |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1      | Рудольф     | Курочкин    | программист | 34  | 2   | 95000  | 17000 | rudolf@yandex.ru |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

или

SELECT \* from employee\_data;

```
mysql> SELECT * from employee_data;
+-----+-----+-----+-----+-----+-----+-----+-----+
| emp_id | f_name      | l_name      | title      | age | yos | salary | perks | email      |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1      | Рудольф     | Курочкин    | программист | 34  | 2   | 95000  | 17000 | rudolf@yandex.ru |
+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Вторая форма лучше. Ее легче использовать и труднее ошибиться.

6. Чтобы вывести данные столбцов f\_name и email, используем следующий оператор.

select f\_name, email from employee\_data;

```
mysql> select f_name, email from employee_data;
+-----+-----+
| f_name      | email      |
+-----+-----+
| Рудольф     | rudolf@yandex.ru |
+-----+-----+
1 row in set (0.00 sec)
```

7. `SELECT salary, perks, yos from employee_data;`

```
+-----+-----+-----+
| salary | perks | yos  |
+-----+-----+-----+
| 95000 | 17000 | 2    |
+-----+-----+-----+
1 row in set (0.00 sec)
```

8. Последняя строка вывода любого оператора `SELECT` содержит число полученных строк. Поэтому при выводе всех данных в любом столбце (или всех столбцах), последняя строка будет указывать число строк в таблице.

```
+-----+-----+-----+
1 row in set (0.00 sec)
```

9. `select salary, l_name from employee_data;`

```
+-----+-----+-----+
| salary | l_name |
+-----+-----+-----+
| 95000 | Курочкин |
+-----+-----+-----+
1 row in set (0.00 sec)
```

### **Выборка данных с помощью условий**

Теперь более подробно рассмотрим формат оператора `SELECT`.

Его полный формат имеет вид:

`SELECT имена_столбцов from имена_таблицы [WHERE ...условия];`

В операторе `SELECT` условия являются необязательными.

Оператор `SELECT` без условий выводит все данные из указанных столбцов. Одним из достоинств *RDBMS* является возможность извлекать данные на основе определенных условий.

Для начала работы добавим ещё 9 пользователей в нашу таблицу

```

INSERT INTO employee_data
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Иван", "Иванов", "стажер", 25, 1, 35000, 5000, "ivanov@yandex.ru");
INSERT INTO employee_data
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Иван", "Лепёткин", "администратор", 38, 7, 130000, 23000, "lepetkin@yandex.ru");
INSERT INTO employee_data
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Елизаветта", "Бурундук", "бухгалтер", 63, 38, 32000, 11000,
"burunduk@yandex.ru");
INSERT INTO employee_data
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Серафим", "Майборода", "программист", 32, 5, 95000, 17000,
"myboroda@yandex.ru");
INSERT INTO employee_data
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Карина", "Куценко", "клинер", 40, 0, 25000, 4000, "kutsenko@yandex.ru");
INSERT INTO employee_data
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Федор", "Достоевский", "дизайнер", 48, 12, 110000, 13000,
"dostoyevsky@yandex.ru");
INSERT INTO employee_data
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Ия", "Сопкина", "дизайнер", 34, 2, 110000, 13000, "sopkina@yandex.ru");
INSERT INTO employee_data
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Василий", "Тёркин", "Web-разработчик", 51, 3, 30000, 11000, "terka@yandex.ru");
INSERT INTO employee_data
(f_name, l_name, title, age, yos, salary, perks, email)
values
("Светлана", "Ержанова", "секретарь", 33, 3, 60000, 9000, "erjanova@yandex.ru");

```

### **Операторы сравнения = и !=**

```
SELECT f_name, l_name from employee_data where f_name = 'Иван';
```

Результат запроса приведен на рис. 1.

f_name	l_name
Иван	Иванов
Иван	Лепёткин

2 rows in set (0.01 sec)

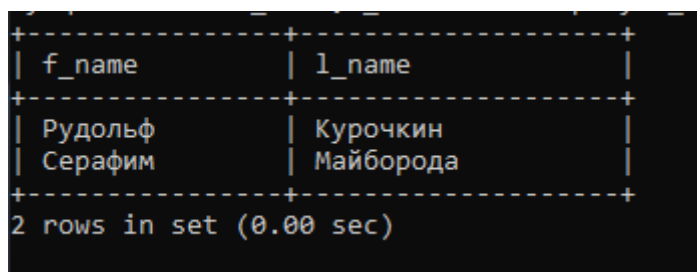
**Рис. 1.** Выборка столбцов с условием для поля "имя".

Этот оператор выводит имена и фамилии всех сотрудников, которые имеют имя Иван. Отметим, что слово Иван в условии заключено в одиночные кавычки. Можно

использовать также двойные кавычки. Кавычки являются обязательными, так как MySQL будет порождать ошибку при их отсутствии. Кроме того сравнения MySQL не различают регистр символов, что означает, что с равным успехом можно использовать "Иван", "иван" и даже "ИвАн".

```
SELECT f_name, l_name from employee_data where  
title="программист";
```

Результат запроса приведен на рис. 2.



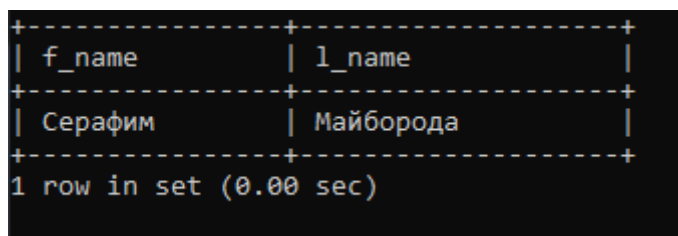
```
+-----+-----+  
| f_name | l_name |  
+-----+-----+  
| Рудольф | Курочкин |  
| Серафим | Майборода |  
+-----+-----+  
2 rows in set (0.00 sec)
```

**Рис. 2.** Выборка столбцов с условием для поля "должность"

Выбирает имена и фамилии всех сотрудников, которые являются программистами.

```
SELECT f_name, l_name from employee_data where age = 32;
```

Результат запроса приведен на рис. 3.



```
+-----+-----+  
| f_name | l_name |  
+-----+-----+  
| Серафим | Майборода |  
+-----+-----+  
1 row in set (0.00 sec)
```

**Рис. 3.** Выборка столбцов с условием для поля "возраст"

Это список имен и фамилий всех сотрудников с возрастом 32 года. Вспомните, что тип столбца `age` был задан как `int`, поэтому кавычки вокруг 32 не требуются. Это - незначительное различие между текстовым и целочисленным типами столбцов.

Оператор `!=` означает 'не равно' и является противоположным оператору равенства.

### ***Операторы больше и меньше***

Давайте получим имена и фамилии всех сотрудников, которые старше 32 лет.

```
SELECT f_name, l_name from employee_data where age > 32;
```

Результат запроса приведен на рис. 4.

f_name	l_name
Рудольф	Курочкин
Иван	Лепёткин
Елизаветта	Бурундук
Карина	Куценко
Федор	Достоевский
Ия	Сопкина
Василий	Тёркин
Светлана	Ержанова

9 rows in set (0.00 sec)

**Рис. 4.** Выборка столбцов с условием "больше" для поля "возраст"

Попробуем найти сотрудников, которые получают зарплату больше 120000.

```
SELECT f_name, l_name from employee_data where salary > 120000;
```

Результат запроса приведен на рис. 5.

f_name	l_name
Иван	Лепёткин

1 row in set (0.00 sec)

**Рис. 5.** Выборка столбцов с условием "больше" для поля "зарплата"

Теперь перечислим всех сотрудников, которые имеют стаж работы в компании менее 3 лет.

```
SELECT f_name, l_name from employee_data where yos < 3;
```

Результат запроса приведен на рис. 6.

f_name	l_name
Рудольф	Курочкин
Иван	Иванов
Карина	Куценко
Ия	Сопкина

4 rows in set (0.00 sec)

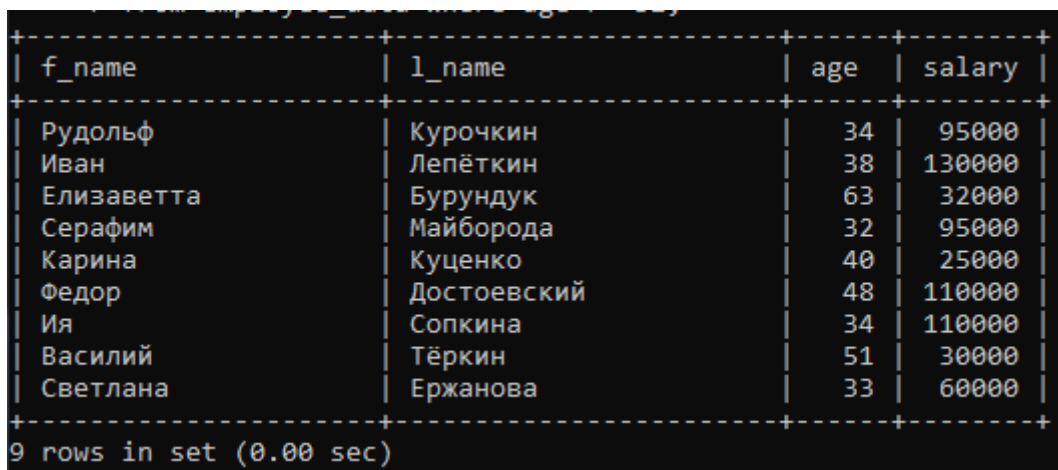
**Рис. 6.** Выборка столбцов с условием "меньше" для поля "стаж"

## Операторы <= и >=

Используемые в основном с целочисленными данными операторы меньше или равно ( <= ) и больше или равно ( >= ) обеспечивают дополнительные возможности.

```
select f_name, l_name, age, salary
from employee_data where age >= 32;
```

Результат запроса приведен на рис. 7.



f_name	l_name	age	salary
Рудольф	Курочкин	34	95000
Иван	Лепёткин	38	130000
Елизаветта	Бурундук	63	32000
Серафим	Майборода	32	95000
Карина	Куценко	40	25000
Федор	Достоевский	48	110000
Ия	Сопкина	34	110000
Василий	Тёркин	51	30000
Светлана	Ержанова	33	60000

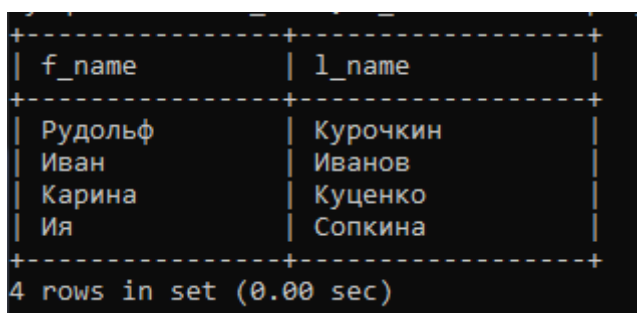
9 rows in set (0.00 sec)

**Рис. 7.** Выборка столбцов с условием "больше или равно" для поля "возраст"

Выборка содержит имена, возраст и зарплаты сотрудников, которым больше 32 лет.

```
select f_name, l_name from employee_data where yos<= 2;
```

Результат запроса приведен на рис. 8.



f_name	l_name
Рудольф	Курочкин
Иван	Иванов
Карина	Куценко
Ия	Сопкина

4 rows in set (0.00 sec)

**Рис. 8.** Выборка столбцов с условием "меньше или равно" для поля "стаж"

Запрос выводит имена сотрудников, которые работают в компании не более 2 лет.



## **Задания № 2:**

1. Напишите оператор `SELECT` для извлечения идентификационного номера сотрудников, которые старше 30 лет.
2. Напишите оператор `SELECT` для извлечения имен и фамилий всех Web-разработчиков.
3. Что выведет следующий оператор `SELECT`:

```
SELECT * from employee_data where salary <=100000;
```

4. Как вывести зарплаты и надбавки сотрудников, которые получают в качестве надбавок более 16000?
5. Перечислите имена всех сотрудников (фамилия, а затем имя), которые занимают должность бухгалтера.

## ***Возможные решения:***

1. `select emp_id from employee_data where age > 30;`
2. `select f_name, l_name from employee_data where title='Web-разработчик';`
3. Следующий оператор выводит всю информацию о сотрудниках, которые получают зарплату не больше 100000.  
  
`SELECT * from employee_data where salary <=100000;`
4. `select salary, perks from employee_data where perks > 16000;`
5. `select l_name, f_name from employee_data where title = 'бухгалтер';`

```

+-----+
| emp_id |
+-----+
| 1      |
| 3      |
| 4      |
| 5      |
| 6      |
| 7      |
| 8      |
| 10     |
| 11     |
+-----+
9 rows in set (0.00 sec)

mysql> select f_name, l_name from employee_data where title='Web-разработчик';
+-----+-----+
| f_name | l_name |
+-----+-----+
| Василий | Тёркин |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT * from employee_data where salary <=100000;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| emp_id | f_name | l_name | title | age | yos | salary | perks | email |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1      | Рудольф | Курочкин | программист | 34 | 2 | 95000 | 17000 | rudolf@yandex.ru |
| 2      | Иван | Иванов | стажер | 25 | 1 | 35000 | 5000 | ivanov@yandex.ru |
| 4      | Елизаветта | Бурундук | бухгалтер | 63 | 38 | 32000 | 11000 | burunduk@yandex.ru |
| 5      | Серафим | Майборода | программист | 32 | 5 | 95000 | 17000 | myboroda@yandex.ru |
| 6      | Карина | Куценко | клинер | 40 | 0 | 25000 | 4000 | kutsenko@yandex.ru |
| 10     | Светлана | Ержанова | секретарь | 33 | 3 | 60000 | 9000 | erjanova@yandex.ru |
| 11     | Василий | Тёркин | Web-разработчик | 31 | 3 | 30000 | 11000 | terka@yandex.ru |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
7 rows in set (0.00 sec)

mysql> select salary, perks from employee_data where perks > 16000;
+-----+-----+
| salary | perks |
+-----+-----+
| 95000 | 17000 |
| 130000 | 23000 |
| 95000 | 17000 |
+-----+-----+
3 rows in set (0.00 sec)

mysql> select l_name, f_name from employee_data where title = 'бухгалтер';
+-----+-----+
| l_name | f_name |
+-----+-----+
| Бурундук | Елизаветта |
+-----+-----+

```

## Поиск текстовых данных по шаблону

В данной части мы рассмотрим *поиск текстовых данных по шаблону* с помощью предложения **where** и оператора **LIKE**.

Оператор сравнения на *равенство* ( = ) помогает выбрать одинаковые строки. Таким образом, чтобы перечислить имена сотрудников, которых зовут Иван, можно воспользоваться следующим оператором **SELECT**.

```
select f_name, l_name from employee_data where f_name = "Иван";
```

Результат запроса приведен на рис. 9.

```

+-----+-----+
| f_name | l_name |
+-----+-----+
| Иван   | Гусев   |
| Иван   | Макаров |
+-----+-----+
2 rows in set (0.00 sec)

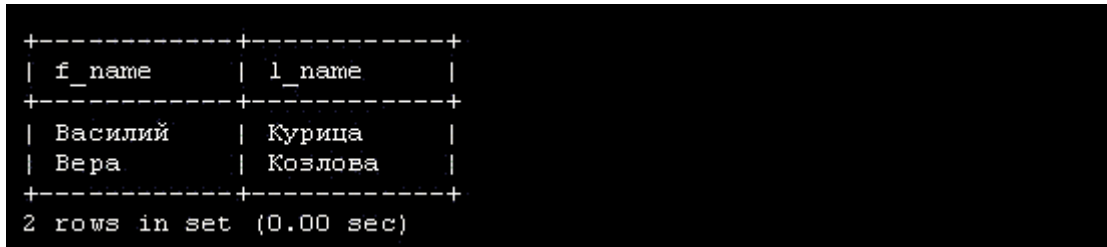
```

**Рис. 9.** Результат поиска сотрудников, которых зовут Иван

Как быть, если надо вывести данные о сотрудниках, имя которых начинается с буквы В? Язык *SQL* позволяет выполнить *поиск* строковых данных по шаблону. Для этого в предложении *where* используется оператор *LIKE* следующим образом.

```
select f_name, l_name from employee_data where f_name LIKE "В%";
```

Результат запроса приведен на рис. 10.



f_name	l_name
Василий	Курица
Вера	Козлова

2 rows in set (0.00 sec)

**Рис. 10.** Результат поиска сотрудников, имя которых начинается с буквы В

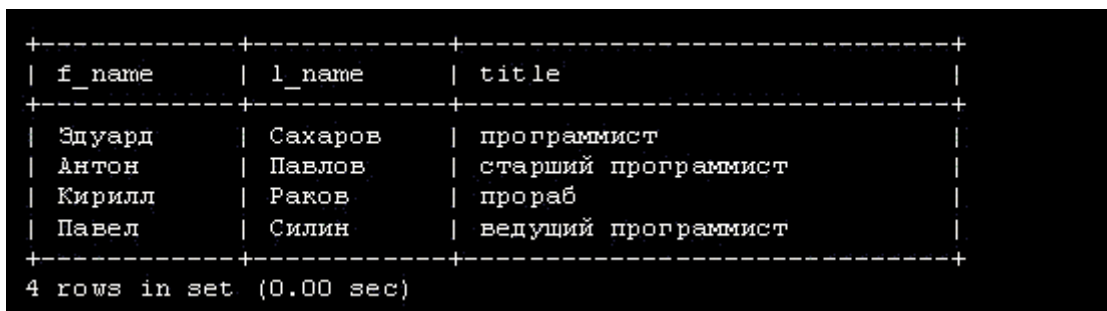
Можно видеть, что здесь в условии вместо знака равенства используется *LIKE* и знак процента в шаблоне.

Знак % действует как символ-заместитель (аналогично использованию \* в системах *DOS* и *Linux*). Он заменяет собой любую последовательность символов. Таким образом "В%" обозначает все строки, которые начинаются с буквы В. Аналогично "%В" выбирает строки, которые заканчиваются символом В, а "%В%" строки, которые содержат букву В.

Давайте выведем, например, всех сотрудников, которые имеют в названии должности строку "про".

```
select f_name, l_name, title from employee_data  
where title like '%про%';
```

Результат запроса приведен на рис. 11.



f_name	l_name	title
Эдуард	Сахаров	программист
Антон	Павлов	старший программист
Кирилл	Раков	прораб
Павел	Силин	ведущий программист

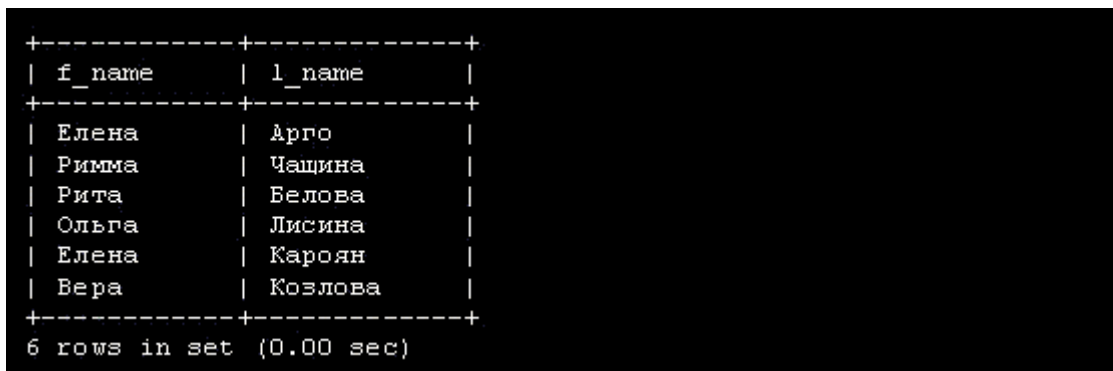
4 rows in set (0.00 sec)

**Рис. 11.** Результат поиска сотрудников, в названии должности которых содержится строка "про"

Перечислим всех сотрудников, имена которых заканчиваются буквой 'а'. Это очень просто сделать.

```
mysql> select f_name, l_name from employee_data
where f_name like '%a';
```

Результат запроса приведен на рис. 12.



f_name	l_name
Елена	Арго
Римма	Чашина
Рита	Белова
Ольга	Лисина
Елена	Кароян
Вера	Козлова

6 rows in set (0.00 sec)

**Рис. 12.** Результат поиска сотрудников, имена которых заканчиваются буквой 'а'

### **Задания № 3:**

1. Перечислить всех сотрудников, фамилии которых начинаются с буквы К.
2. Вывести имена всех сотрудников в отделе продаж.
3. Что выведет следующий оператор

```
SELECT f_name, l_name, salary from
employee_data where f_name like '%к%';
```

4. Перечислить фамилии и должности всех программистов

### **Возможные решения:**

```
1. select l_name, f_name from employee_data where l_name like
'Р%';
```

```
2. select f_name, l_name from employee_data where title like
'%продавец%';
```

3. Этот оператор выводит имена, фамилии и зарплаты всех сотрудников, у которых имя содержит букву 'к'.

```
SELECT f_name, l_name, salary from employee_data where f_name
like '%к%';
```

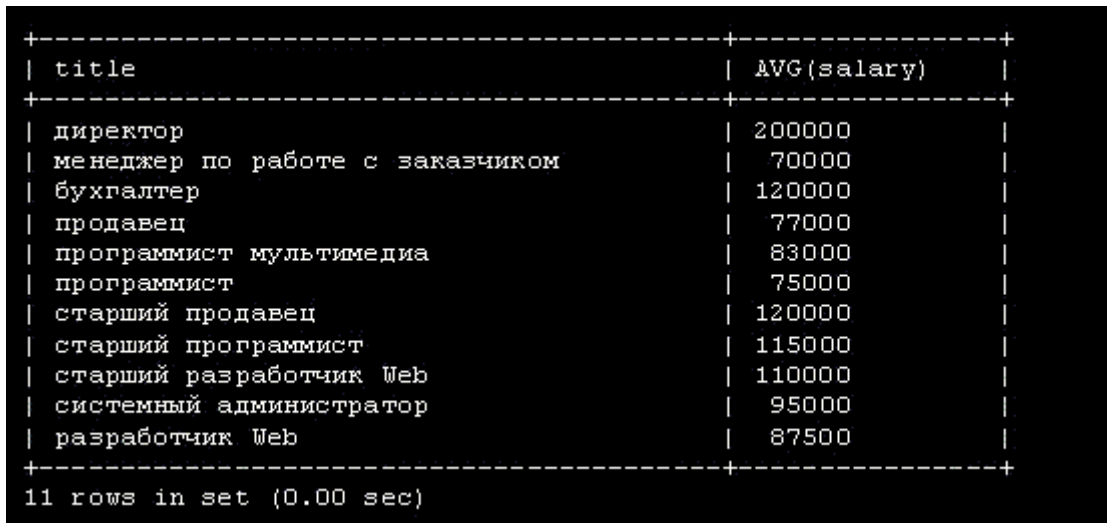
```
4. SELECT l_name, title from employee_data where title like
'%программист%';
```

## Предложение HAVING

Чтобы вывести среднюю зарплату сотрудников в различных подразделениях (должностях), используется предложение `GROUP BY`, например:

```
select title, AVG(salary)
from employee_data
GROUP BY title;
```

Результат запроса приведен на рис. 13.



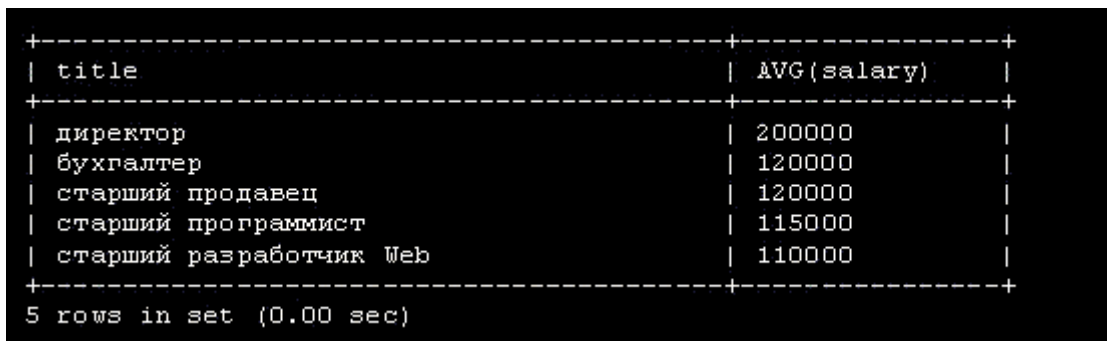
```
+-----+-----+
| title                                | AVG(salary) |
+-----+-----+
| директор                            | 200000      |
| менеджер по работе с заказчиком    | 70000       |
| бухгалтер                           | 120000      |
| продавец                             | 77000       |
| программист мультимедиа             | 83000       |
| программист                         | 75000       |
| старший продавец                    | 120000      |
| старший программист                 | 115000      |
| старший разработчик Web             | 110000      |
| системный администратор             | 95000       |
| разработчик Web                     | 87500       |
+-----+-----+
11 rows in set (0.00 sec)
```

**Рис. 13.** Вывод средней зарплаты сотрудников по подразделениям

Предположим теперь, что требуется вывести только те *подразделения*, где средняя зарплата более 100000. Это можно сделать с помощью предложения `HAVING`.

```
select title, AVG(salary)
from employee_data
GROUP BY title
HAVING AVG(salary) > 100000;
```

Результат запроса приведен на рис. 14.



```
+-----+-----+
| title                                | AVG(salary) |
+-----+-----+
| директор                            | 200000      |
| бухгалтер                           | 120000      |
| старший продавец                    | 120000      |
| старший программист                 | 115000      |
| старший разработчик Web             | 110000      |
+-----+-----+
5 rows in set (0.00 sec)
```

**Рис. 14.** Вывод средней зарплаты определённого диапазона по подразделениям

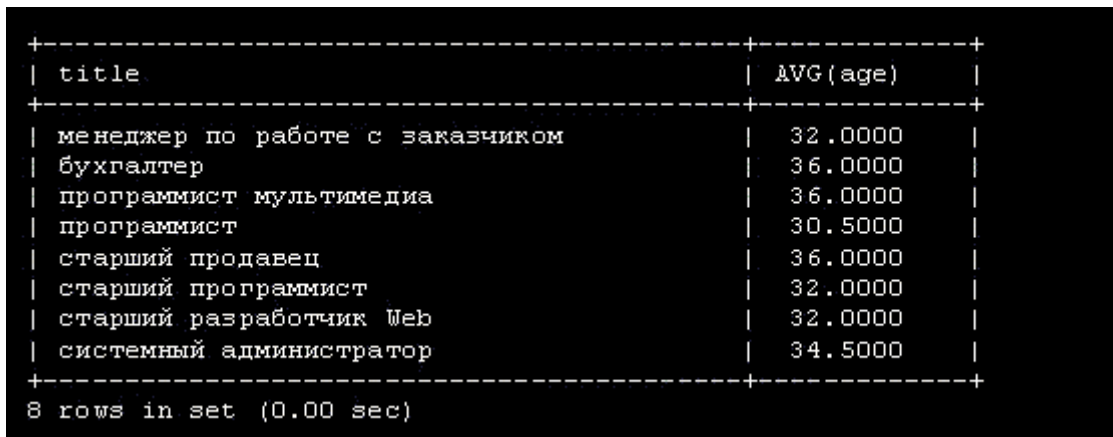
#### **Задание № 4:**

Вывести подразделения и средний возраст, где средний возраст больше 30.

#### ***Возможное решение***

```
mysql> select title, AVG(age)
      -> from employee_data
      -> GROUP BY title
      -> HAVING AVG(age) > 30;
```

Результат запроса приведен на рис. 15.



title	AVG(age)
менеджер по работе с заказчиком	32.0000
бухгалтер	36.0000
программист мультимедиа	36.0000
программист	30.5000
старший продавец	36.0000
старший программист	32.0000
старший разработчик Web	32.0000
системный администратор	34.5000

8 rows in set (0.00 sec)

**Рис. 15.** Вывод подразделения и среднего возраста, где средний возраст больше 30 лет

#### **Удаление записей из таблицы**

Для *удаления записей из таблицы* можно использовать оператор **DELETE**.

Оператор удаления **DELETE** требует задания имени таблицы и необязательных условий.

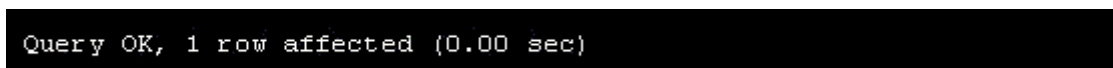
```
DELETE from имя_таблицы [WHERE условия];
```

Примечание: Если никакие условия не будут заданы, то удаляются все данные в таблице.

Предположим, один из специалистов *по мультимедиа* 'Василий Пупкин' уволился из компании. Надо удалить его *запись*.

```
DELETE from employee_data
WHERE l_name = 'Пупкин';
```

Результат запроса приведен на рис. 16.



Query OK, 1 row affected (0.00 sec)

**Рис. 16.** Результат удаления записи из таблицы

## Самостоятельная работа

На основе таблицы `employee_data` создайте различные 6 запросов:

- с помощью условий больше, меньше, равно;
- с помощью предложения `where`;
- с помощью оператора `LIKE`;
- предложение `GROUP BY`;
- предложения `HAVING`.