

Домашнее задание №4

Задание 1

Получите всю информацию о товарах из таблицы **products**.

```
DROP table products;
```

```
CREATE TABLE products (  
    id INT NOT NULL,  
    name NVARCHAR(255) NULL,  
    count INTEGER NULL,  
    price INTEGER NULL
```

```
);
```

```
INSERT INTO products (id, name, count, price)
```

```
VALUES
```

```
(1, 'Стиральная машина', 5, 10000),
```

```
(2, 'Холодильник', 0, 10000),
```

```
(3, 'Микроволновка', 3, 4000),
```

```
(4, 'Пылесос', 2, 4500),
```

```
(5, 'Вентилятор', 0, 700),
```

```
(6, 'Телевизор', 7, 31740);
```

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740

Affected rows: 6

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```
SELECT * FROM products
```

100 %

Результаты

	id	name	count	price
1	1	Стиральная машина	5	10000
2	2	Холодильник	0	10000
3	3	Микроволновка	3	4000
4	4	Пылесос	2	4500
5	5	Вентилятор	0	700
6	6	Телевизор	7	31740

Задание 2

Получите название (**name**) и цену (**price**) всех товаров из таблицы **products**.

```
CREATE TABLE products (  
  id INT NOT NULL,  
  name NVARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
INSERT INTO products (id, name, count, price)  
VALUES  
  (1, 'Стиральная машина', 5, 10000),  
  (2, 'Холодильник', 0, 10000),  
  (3, 'Микроволновка', 3, 4000),  
  (4, 'Пылесос', 2, 4500),  
  (5, 'Вентилятор', 0, 700),  
  (6, 'Телевизор', 7, 31740);
```

name	price
Стиральная машина	10000
Холодильник	10000
Микроволновка	4000
Пылесос	4500
Вентилятор	700
Телевизор	31740

Affected rows: 6

Решение:

SQLQuery1.sql - PT...y (PTG\Павел (56))*		
SELECT name , price FROM products		
100 %		
Результаты		
	name	price
1	Стиральная машина	10000
2	Холодильник	10000
3	Микроволновка	4000
4	Пылесос	4500
5	Вентилятор	700
6	Телевизор	31740

Задание 3

Выберите из таблицы **products** все записи, в которых цена (**price**) меньше 3000.

```
CREATE TABLE products (  
  id INT NOT NULL,  
  name NVARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
INSERT INTO products (id, name, count, price)  
VALUES  
  (1, 'Стиральная машина', 5, 10000),  
  (2, 'Холодильник', 0, 10000),  
  (3, 'Микроволновка', 3, 4000),  
  (4, 'Пылесос', 2, 4500),  
  (5, 'Вентилятор', 0, 700),  
  (6, 'Телевизор', 7, 31740),  
  (7, 'Тостер', 2, 2500),  
  (8, 'Принтер', 4, 3000);
```

id	name	count	price
5	Вентилятор	0	700
7	Тостер	2	2500

Affected rows: 2

Решение:

The screenshot shows a SQL query window titled "SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*". The query is: `SELECT id, name, count FROM products WHERE price < 3000`. Below the query, the "Results" tab is active, displaying a table with 4 columns: "id", "name", "count", and "price". The table contains two rows: row 1 with id 5, name "Вентилятор", and count 0; row 2 with id 7, name "Тостер", and count 2. The "price" column is not visible in the results table.

	id	name	count
1	5	Вентилятор	0
2	7	Тостер	2

Задание 4

Выберите из таблицы **products** имена (**name**) и цены (**price**) всех товаров, стоимостью от 10 000 и выше.

```
CREATE TABLE products (  
  id INT NOT NULL,  
  name NVARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
INSERT INTO products (id, name, count, price)  
VALUES  
  (1, 'Стиральная машина', 5, 10000),  
  (2, 'Холодильник', 0, 10000),  
  (3, 'Микроволновка', 3, 4000),  
  (4, 'Пылесос', 2, 4500),  
  (5, 'Вентилятор', 0, 700),  
  (6, 'Телевизор', 7, 31740),  
  (7, 'Тостер', 2, 2500),  
  (8, 'Принтер', 4, 3000);
```

name	price
Стиральная машина	10000
Холодильник	10000
Телевизор	31740

Affected rows: 3

Решение:

SQLQuery1.sql - PT...y (PTG\Павел (56))*		
<pre>SELECT name, price FROM products WHERE price >= 10000;</pre>		
100 %		
Результаты		
	name	price
1	Стиральная машина	10000
2	Холодильник	10000
3	Телевизор	31740

Задание 5

Получите из таблицы **products** имена (**name**) товаров, которые закончились.

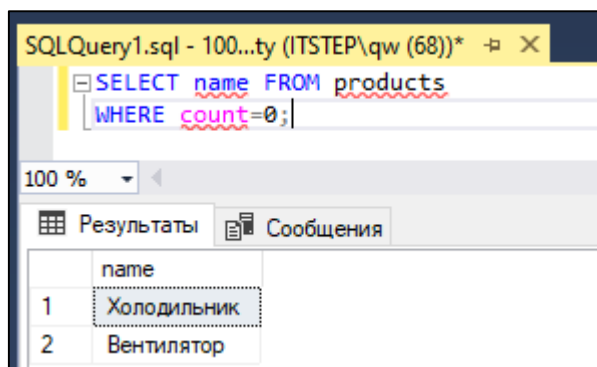
```
CREATE TABLE products (  
    id INT NOT NULL,  
    name NVARCHAR(255) NULL,  
    count INTEGER NULL,  
    price INTEGER NULL  
);  
INSERT INTO products (id, name, count, price)  
VALUES  
    (1, 'Стиральная машина', 5, 10000),  
    (2, 'Холодильник', 0, 10000),  
    (3, 'Микроволновка', 3, 4000),  
    (4, 'Пылесос', 2, 4500),  
    (5, 'Вентилятор', 0, 700),  
    (6, 'Телевизор', 7, 31740),  
    (7, 'Тостер', 2, 2500),  
    (8, 'Принтер', 4, 3000);
```

Query result:

name
Холодильник
Вентилятор

Affected rows: 2

Решение:



Задание 6

Выберите из таблицы **products** название (**name**) и цены (**price**) товаров, стоимостью до 4000 включительно.

```
CREATE TABLE products (  
  id INT NOT NULL,  
  name NVARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
INSERT INTO products (id, name, count, price)  
VALUES  
  (1, 'Стиральная машина', 5, 10000),  
  (2, 'Холодильник', 0, 10000),  
  (3, 'Микроволновка', 3, 4000),  
  (4, 'Пылесос', 2, 4500),  
  (5, 'Вентилятор', 0, 700),  
  (6, 'Телевизор', 7, 31740),  
  (7, 'Тостер', 2, 2500),  
  (8, 'Принтер', 4, 3000);
```

name	price
Микроволновка	4000
Вентилятор	700
Тостер	2500
Принтер	3000

Affected rows: 4

Решение:

SQLQuery1.sql - PT...y (PTG\Павел (56))*

```
SELECT name, price FROM products  
WHERE price <= 4000
```

100 %

Результаты Сообщения

	name	price
1	Микроволновка	4000
2	Вентилятор	700
3	Тостер	2500
4	Принтер	3000

Задание 1

Выберите из таблицы **orders** все заказы кроме отмененных. У отмененных заказов **status** равен **"cancelled"**.

```
CREATE TABLE orders (  
  id INT NOT NULL PRIMARY KEY,  
  user_id INTEGER NULL,  
  products_count INTEGER NULL,  
  sum INTEGER NULL,  
  status VARCHAR(20) NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status)  
VALUES  
  (1, 1, 2, 1300, 'new'),  
  (2, 18, 1, 200, 'cancelled'),  
  (3, 11, 1, 2140, 'in_progress'),  
  (4, 145, 5, 6800, 'new'),  
  (5, 23, 1, 999, 'new'),  
  (6, 1, 2, 7690, 'cancelled'),  
  (7, 17, 1, 1600, 'new'),  
  (8, 5, 4, 400, 'delivery'),  
  (9, 2355, 1, 1450, 'new'),  
  (10, 13, 7, 13000, 'new');
```

id	user_id	products_count	sum	status
1	1	2	1300	new
3	11	1	2140	in_progress
4	145	5	6800	new
5	23	1	999	new
7	17	1	1600	new
8	5	4	400	delivery
9	2355	1	1450	new
10	13	7	13000	new

Affected rows: 8

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```
SELECT * FROM orders  
WHERE status not IN ('cancelled');
```

100 %

Результаты

	id	user_id	products_count	sum	status
1	1	1	2	1300	new
2	3	11	1	2140	in_progress
3	4	145	5	6800	new
4	5	23	1	999	new
5	7	17	1	1600	new
6	8	5	4	400	delivery
7	9	2355	1	1450	new
8	10	13	7	13000	new

DROP TABLE orders; - добавлять перед кодом создания и заполнения таблицы

Задание 2

Выберите из таблицы **orders** все заказы содержащие более 3 товаров (**products_count**).

Вывести нужно только номер (**id**) и сумму (**sum**) заказа.

```
CREATE TABLE orders (  
  id INT NOT NULL PRIMARY KEY,  
  user_id INTEGER NULL,  
  products_count INTEGER NULL,  
  sum INTEGER NULL,  
  status VARCHAR(20) NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status)  
VALUES  
  (1, 1, 2, 1300, 'new'),  
  (2, 18, 1, 200, 'cancelled'),  
  (3, 11, 1, 2140, 'in_progress'),  
  (4, 145, 5, 6800, 'new'),  
  (5, 23, 1, 999, 'new'),  
  (6, 1, 2, 7690, 'cancelled'),  
  (7, 17, 1, 1600, 'new'),  
  (8, 5, 4, 400, 'delivery'),  
  (9, 2355, 1, 1450, 'new'),  
  (10, 13, 7, 13000, 'new'),  
  (11, 23, 3, 6500, 'new');
```

id	sum
4	6800
8	400
10	13000

Affected rows: 3

Решение:

SQLQuery1.sql - PT...y (PTG\Павел (56))*

```
SELECT id, sum FROM orders  
WHERE products_count > 3;
```

100 %

Результаты Сообщения

	id	sum
1	4	6800
2	8	400
3	10	13000

Задание 3

Выберите из таблицы **orders** все отмененные заказы. У отмененных заказов **status** равен **"cancelled"**.

```
CREATE TABLE orders (  
  id INT NOT NULL PRIMARY KEY,  
  user_id INTEGER NULL,  
  products_count INTEGER NULL,  
  sum INTEGER NULL,  
  status VARCHAR(20) NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status)  
VALUES  
  (1, 1, 2, 1300, 'new'),  
  (2, 18, 1, 200, 'cancelled'),  
  (3, 11, 1, 2140, 'in_progress'),  
  (4, 145, 5, 6800, 'new'),  
  (5, 23, 1, 999, 'new'),  
  (6, 1, 2, 7690, 'cancelled'),  
  (7, 17, 1, 1600, 'new'),  
  (8, 5, 4, 400, 'delivery'),  
  (9, 2355, 1, 1450, 'new'),  
  (10, 13, 7, 13000, 'new');
```

id	user_id	products_count	sum	status
2	18	1	200	cancelled
6	1	2	7690	cancelled

Affected rows: 2

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```
SELECT *FROM orders  
WHERE status IN ('cancelled')
```

100 %

Результаты

	id	user_id	products_count	sum	status
1	2	18	1	200	cancelled
2	6	1	2	7690	cancelled

Задание 4

Выберите из таблицы **orders** все отмененные (**cancelled**) и возвращенные (**returned**) товары.

```
CREATE TABLE orders (  
  id INT NOT NULL PRIMARY KEY,  
  user_id INTEGER NULL,  
  products_count INTEGER NULL,  
  sum INTEGER NULL,  
  status VARCHAR(20) NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status)  
VALUES  
  (1, 1, 2, 1300, 'new'),  
  (2, 18, 1, 200, 'cancelled'),  
  (3, 11, 1, 2140, 'in_progress'),  
  (4, 145, 5, 6800, 'new'),  
  (5, 23, 1, 999, 'new'),  
  (6, 1, 2, 7690, 'cancelled'),  
  (7, 17, 1, 1600, 'new'),  
  (8, 5, 4, 400, 'delivery'),  
  (9, 2355, 1, 1450, 'new'),  
  (10, 13, 7, 13000, 'returned');
```

id	user_id	products_count	sum	status
2	18	1	200	cancelled
6	1	2	7690	cancelled
10	13	7	13000	returned

Affected rows: 3

Решение:

SQLQuery1.sql - PT...y (PTG\Павел (56))*

```
SELECT * FROM orders  
WHERE status IN ('cancelled', 'returned')
```

100 %

Результаты Сообщения

	id	user_id	products_count	sum	status
1	2	18	1	200	cancelled
2	6	1	2	7690	cancelled
3	10	13	7	13000	returned

Задание 5

Выберите из таблицы **orders** все заказы, у которых сумма (**sum**) больше 3000 или количество товаров (**products_count**) от 3 и больше.

```
CREATE TABLE orders (  
    id INT NOT NULL PRIMARY KEY,  
    user_id INTEGER NULL,  
    products_count INTEGER NULL,  
    sum INTEGER NULL,  
    status VARCHAR(20) NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status)  
VALUES  
    (1, 1, 2, 1300, 'new'),  
    (2, 18, 1, 200, 'cancelled'),  
    (3, 11, 1, 2140, 'in_progress'),  
    (4, 145, 5, 6800, 'new'),  
    (5, 23, 1, 999, 'new'),  
    (6, 1, 2, 7690, 'cancelled'),  
    (7, 17, 1, 1600, 'new'),  
    (8, 5, 4, 400, 'delivery'),  
    (9, 2355, 1, 1450, 'new'),  
    (10, 13, 7, 13000, 'returned');
```

id	user_id	products_count	sum	status
4	145	5	6800	new
6	1	2	7690	cancelled
8	5	4	400	delivery
10	13	7	13000	returned

Affected rows: 4

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```
SELECT *FROM orders  
WHERE sum>3000 OR  
products_count >=3;
```

100 %

Результаты

	id	user_id	products_count	sum	status
1	4	145	5	6800	new
2	6	1	2	7690	cancelled
3	8	5	4	400	delivery
4	10	13	7	13000	returned

6. Выберите из таблицы **orders** все заказы, у которых сумма (**sum**) от 3000 и выше, а количество товаров (**products_count**) меньше 3.

```
CREATE TABLE orders (  
    id INT NOT NULL PRIMARY KEY,  
    user_id INTEGER NULL,  
    products_count INTEGER NULL,  
    sum INTEGER NULL,  
    status VARCHAR(20) NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status)  
VALUES  
    (1, 1, 2, 1300, 'new'),  
    (2, 18, 1, 200, 'cancelled'),  
    (3, 11, 1, 2140, 'in_progress'),  
    (4, 145, 5, 6800, 'new'),  
    (5, 23, 1, 999, 'new'),  
    (6, 1, 2, 7690, 'cancelled'),  
    (7, 17, 1, 1600, 'new'),  
    (8, 5, 4, 400, 'delivery'),  
    (9, 2355, 1, 1450, 'new'),  
    (10, 13, 7, 13000, 'returned'),  
    (11, 7, 3, 3000, 'returned'),  
    (12, 8, 1, 3000, 'new');
```

id	user_id	products_count	sum	status
6	1	2	7690	cancelled
12	8	1	3000	new

Affected rows: 2

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```
SELECT * FROM orders  
WHERE sum >= 3000 AND  
products_count < 3;
```

100 %

Результаты Сообщения

	id	user_id	products_count	sum	status
1	6	1	2	7690	cancelled
2	12	8	1	3000	new

Задание 7

Выберите из таблицы **orders** все отмененные заказы стоимостью от 3000 до 10000 рублей включительно.

```
CREATE TABLE orders (  
  id INT NOT NULL PRIMARY KEY,  
  user_id INTEGER NULL,  
  products_count INTEGER NULL,  
  sum INTEGER NULL,  
  status VARCHAR(20) NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status)  
VALUES  
  (1, 1, 2, 1300, 'new'),  
  (2, 18, 1, 10000, 'cancelled'),  
  (3, 11, 1, 2140, 'in_progress'),  
  (4, 145, 5, 6800, 'new'),  
  (5, 23, 1, 999, 'new'),  
  (6, 1, 2, 7690, 'cancelled'),  
  (7, 17, 1, 1600, 'new'),  
  (8, 5, 4, 400, 'delivery'),  
  (9, 2355, 1, 1450, 'new'),  
  (10, 13, 7, 13000, 'cancelled'),  
  (11, 144, 6, 3000, 'cancelled');
```

id	user_id	products_count	sum	status
2	18	1	10000	cancelled
6	1	2	7690	cancelled
11	144	6	3000	cancelled

Affected rows: 3

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```
SELECT * FROM orders  
WHERE status IN ('cancelled')  
AND sum BETWEEN 3000 AND 10000;
```

100 %

Результаты Сообщения

	id	user_id	products_count	sum	status
1	2	18	1	10000	cancelled
2	6	1	2	7690	cancelled
3	11	144	6	3000	cancelled

Задание 8

Выберите из таблицы **orders** все отмененные заказы исключая заказы стоимостью от 3000 до 10000 рублей включительно.

```
CREATE TABLE orders (  
  id INT NOT NULL PRIMARY KEY,  
  user_id INTEGER NULL,  
  products_count INTEGER NULL,  
  sum INTEGER NULL,  
  status VARCHAR(20) NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status)  
VALUES  
  (1, 1, 2, 1300, 'new'),  
  (2, 18, 1, 10000, 'cancelled'),  
  (3, 45, 3, 800, 'cancelled'),  
  (4, 11, 1, 2140, 'in_progress'),  
  (5, 145, 5, 6800, 'new'),  
  (6, 23, 1, 999, 'new'),  
  (7, 1, 2, 7690, 'cancelled'),  
  (8, 17, 1, 1600, 'new'),  
  (9, 5, 4, 400, 'delivery'),  
  (10, 2355, 1, 1450, 'new'),  
  (11, 13, 7, 13000, 'cancelled');
```

id	user_id	products_count	sum	status
3	45	3	800	cancelled
11	13	7	13000	cancelled

Affected rows: 2

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*					
<pre>SELECT * FROM orders WHERE status IN ('cancelled') AND sum NOT BETWEEN 3000 AND 10000;</pre>					
100 %					
Результаты					
	id	user_id	products_count	sum	status
1	3	45	3	800	cancelled
2	11	13	7	13000	cancelled

Номер 1

Выберите из таблицы **products** все товары в порядке возрастания цены (**price**).

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740
7	Тостер	2	2500
8	Принтер	4	3000

```
CREATE TABLE products (  
  id INT UNSIGNED NOT NULL PRIMARY KEY,  
  name VARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);
```

```
INSERT INTO products (id, name, count, price)
```

```
VALUES
```

```
(1, 'Стиральная машина', 5, 10000),  
(2, 'Холодильник', 0, 10000),  
(3, 'Микроволновка', 3, 4000),  
(4, 'Пылесос', 2, 4500),
```

(5, 'Вентилятор', 0, 700),

(6, 'Телевизор', 7, 31740),

(7, 'Тостер', 2, 2500),

(8, 'Принтер', 4, 3000);

id	name	count	price
5	Вентилятор	0	700
7	Тостер	2	2500
8	Принтер	4	3000
3	Микроволновка	3	4000
4	Пылесос	2	4500
1	Стиральная машина	5	10000
2	Холодильник	0	10000
6	Телевизор	7	31740

Affected rows: 8

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```
SELECT *FROM products  
ORDER BY price
```

100 %

Результаты

	id	name	count	price
1	5	Вентилятор	0	700
2	7	Тостер	2	2500
3	8	Принтер	4	3000
4	3	Микроволновка	3	4000
5	4	Пылесос	2	4500
6	1	Стиральная машина	5	10000
7	2	Холодильник	0	10000
8	6	Телевизор	7	31740

Номер 2

Выберите из таблицы **products** все товары в порядке убывания цены.

Выведите только имена (**name**) и цены (**price**).

products

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740
7	Тостер	2	2500
8	Принтер	4	3000

```
CREATE TABLE products (  
  id INT UNSIGNED NOT NULL PRIMARY KEY,  
  name VARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
  
INSERT INTO products (id, name, count, price)  
VALUES  
  (1, 'Стиральная машина', 5, 10000),
```

(2, 'Холодильник', 0, 10000),
(3, 'Микроволновка', 3, 4000),
(4, 'Пылесос', 2, 4500),
(5, 'Вентилятор', 0, 700),
(6, 'Телевизор', 7, 31740),
(7, 'Тостер', 2, 2500),
(8, 'Принтер', 4, 3000);

name	price
Телевизор	31740
Стиральная машина	10000
Холодильник	10000
Пылесос	4500
Микроволновка	4000
Принтер	3000
Тостер	2500
Вентилятор	700

Affected rows: 8

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```
SELECT name, price FROM products  
ORDER BY price DESC
```

100 %

Результаты Сообщения

	name	price
1	Телевизор	31740
2	Стиральная машина	10000
3	Холодильник	10000
4	Пылесос	4500
5	Микроволновка	4000
6	Принтер	3000
7	Тостер	2500
8	Вентилятор	700

Номер 3

Выберите из таблицы **products** все товары стоимостью 5000 и выше в порядке убывания цены (**price**).

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740
7	Тостер	2	2500
8	Принтер	4	3000
9	Наушники	7	5000

```
CREATE TABLE products (  
  id INT UNSIGNED NOT NULL PRIMARY KEY,  
  name VARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
  
INSERT INTO products (id, name, count, price)  
VALUES  
  (1, 'Стиральная машина', 5, 10000),
```

(2, 'Холодильник', 0, 10000),
 (3, 'Микроволновка', 3, 4000),
 (4, 'Пылесос', 2, 4500),
 (5, 'Вентилятор', 0, 700),
 (6, 'Телевизор', 7, 31740),
 (7, 'Тостер', 2, 2500),
 (8, 'Принтер', 4, 3000),
 (9, 'Наушники', 7, 5000);

id	name	count	price
6	Телевизор	7	31740
1	Стиральная машина	5	10000
2	Холодильник	0	10000
9	Наушники	7	5000

Affected rows: 4

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```

SELECT * FROM products
WHERE price >= 5000
ORDER BY price ASC
  
```

100 %

Результаты Сообщения

	id	name	count	price
1	9	Наушники	7	5000
2	1	Стиральная машина	5	10000
3	2	Холодильник	0	10000
4	6	Телевизор	7	31740

Номер 4

Выберите из таблицы **products** все товары стоимостью до 3000 рублей отсортированные в алфавитном порядке. Вывести нужно только имя (**name**), количество (**count**) и цену (**price**).

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740
7	Тостер	2	2500
8	Принтер	4	3000
9	Активные колонки	1	2900

```
CREATE TABLE products (  
  id INT UNSIGNED NOT NULL PRIMARY KEY,  
  name VARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
  
INSERT INTO products (id, name, count, price)  
VALUES  
  (1, 'Стиральная машина', 5, 10000),  
  (2, 'Холодильник', 0, 10000),  
  (3, 'Микроволновка', 3, 4000),  
  (4, 'Пылесос', 2, 4500),  
  (5, 'Вентилятор', 0, 700),  
  (6, 'Телевизор', 7, 31740),
```

(7, 'Тостер', 2, 2500),

(8, 'Принтер', 4, 3000),

(9, 'Активные колонки', 1, 2900);

query results:

name	count	price
Активные колонки	1	2900
Вентилятор	0	700
Тостер	2	2500

Affected rows: 3

Решение:

SQLQuery1.sql - PT...y (PTG\Павел (56))*

```
SELECT name, count, price FROM products  
WHERE price < 3000  
ORDER BY name ASC
```

100 %

Результаты

	name	count	price
1	Активные колонки	1	2900
2	Вентилятор	0	700
3	Тостер	2	2500

Номер 5

Выберите из таблицы **users** фамилии (**last_name**) и имена (**first_name**) всех пользователей.

Данные должны быть отсортированы сначала по фамилии, а затем по имени.

users			
id	first_name	last_name	birthday
1	Дмитрий	Петров	2000-03-14
2	Ольга	Антонова	1999-12-01
3	Сергей	Васильев	2002-02-20
4	Константин	Степаниденко	2004-03-07
5	Алена	Шикова	1999-08-17
6	Василина	Парамонова	2000-10-10
7	Александр	Пузаков	2002-02-20
8	Алина	Антонова	2002-01-01

```
CREATE TABLE users (  
    id INT UNSIGNED NOT NULL PRIMARY KEY,  
    first_name VARCHAR(50) NULL,  
    last_name VARCHAR(50) NULL,  
    birthday DATE NULL  
);  
  
INSERT INTO users (id, first_name, last_name, birthday)  
VALUES  
    (1, 'Дмитрий', 'Петров', '2000-03-14'),  
    (2, 'Ольга', 'Антонова', '1999-12-01'),  
    (3, 'Сергей', 'Васильев', '2002-02-20'),  
    (4, 'Константин', 'Степаниденко', '2004-03-07'),  
    (5, 'Алена', 'Шикова', '1999-08-17'),  
    (6, 'Василина', 'Парамонова', '2000-10-10'),  
    (7, 'Александр', 'Пузаков', '2002-02-20'),
```

(8, 'Алина', 'Антонова', '2002-01-01');

last_name	first_name
Антонова	Алина
Антонова	Ольга
Васильев	Сергей
Парамонова	Василина
Петров	Дмитрий
Пузаков	Александр
Степаниденко	Константин
Шикова	Алена

Affected rows: 8

Решение:

SQLQuery1.sql - 100...ty (ITSTEP\qw (68))*

```
SELECT last_name, first_name FROM users  
ORDER BY last_name ASC, first_name ASC
```

100 %

Результаты Сообщения

	last_name	first_name
1	Антонова	Алина
2	Антонова	Ольга
3	Васильев	Сергей
4	Парамонова	Василина
5	Петров	Дмитрий
6	Пузаков	Александр
7	Степаниденко	Константин
8	Шикова	Алена

Номер 6

Выберите из таблицы **users** всех пользователей с зарплатой от 40 000 рублей и выше. Данные нужно сначала отсортировать по убыванию зарплаты (**salary**), а затем в алфавитном порядке по имени (**first_name**).

users					
id	first_name	last_name	birthday	salary	job
1	Дмитрий	Петров	2000-03-14	25000	офис-менеджер
2	Ольга	Антонова	1999-12-01	41000	дизайнер
3	Сергей	Васильев	2002-02-20	40000	младший программист
4	Константин	Степаниденко	2004-03-07	30000	водитель
5	Алена	Шикова	1999-08-17	53000	фотограф
6	Василина	Парамонова	2000-10-10	28000	секретарь
7	Александр	Пузаков	2002-02-20	120000	ведущий программист
8	Алина	Антонова	2002-01-01	40000	верстальщик

```
CREATE TABLE users (  
    id INT UNSIGNED NOT NULL PRIMARY KEY,  
    first_name VARCHAR(50) NULL,  
    last_name VARCHAR(50) NULL,  
    birthday DATE NULL,  
    salary INTEGER NULL,  
    job VARCHAR(50) NULL  
);  
  
INSERT INTO users (id, first_name, last_name, birthday, salary, job)  
VALUES  
    (1, 'Дмитрий', 'Петров', '2000-03-14', 25000, 'офис-менеджер'),  
    (2, 'Ольга', 'Антонова', '1999-12-01', 41000, 'дизайнер'),  
    (3, 'Сергей', 'Васильев', '2002-02-20', 40000, 'младший программист'),  
    (4, 'Константин', 'Степаниденко', '2004-03-07', 30000, 'водитель'),  
    (5, 'Алена', 'Шикова', '1999-08-17', 53000, 'фотограф'),
```

(6, 'Василина', 'Парамонова', '2000-10-10', 28000, 'секретарь'),

(7, 'Александр', 'Пузаков', '2002-02-20', 120000, 'ведущий программист'),

(8, 'Алина', 'Антонова', '2002-01-01', 40000, 'верстальщик');

id	first_name	last_name	birthday	salary	job
7	Александр	Пузаков	2002-02-20	120000	ведущий программист
5	Алена	Шикова	1999-08-17	53000	фотограф
2	Ольга	Антонова	1999-12-01	41000	дизайнер
8	Алина	Антонова	2002-01-01	40000	верстальщик
3	Сергей	Васильев	2002-02-20	40000	младший программист

Affected rows: 5

Решение:

SQLQuery1.sql - PT...y (PTG\Павел (56))*						
<pre>SELECT * FROM users WHERE salary >= 40000 ORDER BY salary DESC, first_name ASC</pre>						
100 %						
Результаты						
	id	first_name	last_name	birthday	salary	job
1	7	Александр	Пузаков	2002-02-20	120000	ведущий программист
2	5	Алена	Шикова	1999-08-17	53000	фотограф
3	2	Ольга	Антонова	1999-12-01	41000	дизайнер
4	8	Алина	Антонова	2002-01-01	40000	верстальщик
5	3	Сергей	Васильев	2002-02-20	40000	младший программист

Номер 7

Выберите сотрудников из таблицы **users** с зарплатой (**salary**) меньше 30 000 рублей и отсортируйте данные по дате рождения (**birthday**). Сотрудников с нулевой зарплатой выбирать не нужно.

users					
id	first_name	last_name	birthday	salary	job
1	Дмитрий	Петров	2000-03-14	25000	офис-менеджер
2	Ольга	Антонова	1999-12-01	41000	дизайнер
3	Сергей	Васильев	2002-02-20	40000	младший программист
4	Константин	Степаниденко	2004-03-07	30000	водитель
5	Алена	Шикова	1999-08-17	0	фотограф
6	Василина	Парамнова	2000-02-10	28000	секретарь
7	Александр	Пузаков	2002-02-20	120000	ведущий программист
8	Алина	Антонова	2002-01-01	40000	верстальщик

```
CREATE TABLE users (  
    id INT UNSIGNED NOT NULL PRIMARY KEY,  
    first_name VARCHAR(50) NULL,  
    last_name VARCHAR(50) NULL,  
    birthday DATE NULL,  
    salary INTEGER NULL,  
    job VARCHAR(50) NULL  
);  
  
INSERT INTO users (id, first_name, last_name, birthday, salary, job)  
VALUES  
    (1, 'Дмитрий', 'Петров', '2000-03-14', 25000, 'офис-менеджер'),  
    (2, 'Ольга', 'Антонова', '1999-12-01', 41000, 'дизайнер'),  
    (3, 'Сергей', 'Васильев', '2002-02-20', 40000, 'младший программист'),  
    (4, 'Константин', 'Степаниденко', '2004-03-07', 30000, 'водитель'),  
    (5, 'Алена', 'Шикова', '1999-08-17', 0, 'фотограф'),
```

(6, 'Василина', 'Парамнова', '2000-02-10', 28000, 'секретарь'),

(7, 'Александр', 'Пузаков', '2002-02-20', 120000, 'ведущий программист'),

(8, 'Алина', 'Антонова', '2002-01-01', 40000, 'верстальщик');

id	first_name	last_name	birthday	salary	job
6	Василина	Парамнова	2000-02-10	28000	секретарь
1	Дмитрий	Петров	2000-03-14	25000	офис-менеджер

Affected rows: 2

Решение:

SQLQuery3.sql - 100...ty (ITSTEP\qw (53))*						
<pre>SELECT * FROM users WHERE salary < 30000 AND salary NOT IN (0) ORDER BY birthday</pre>						
100 %						
Результаты						
	id	first_name	last_name	birthday	salary	job
1	6	Василина	Парамнова	2000-02-10	28000	секретарь
2	1	Дмитрий	Петров	2000-03-14	25000	офис-менеджер

Номер 1

Выберите из таблицы **orders** 5 самых дорогих заказов за всё время.

Данные нужно отсортировать в порядке убывания цены.

Отмененные заказы не учитывайте.

orders

id	user_id	products_count	sum	status
1	1	2	1300	new
2	18	1	10000	cancelled
3	11	1	2140	in_progress
4	145	5	6800	new
5	23	1	999	new
6	1	2	7690	cancelled
7	17	1	1600	new
8	5	4	400	delivery
9	2355	1	1450	new
10	13	7	13000	cancelled

```
CREATE TABLE orders (  
  id INT NOT NULL PRIMARY KEY,  
  user_id INTEGER NULL,  
  products_count INTEGER NULL,  
  sum INTEGER NULL,  
  status VARCHAR(20) NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status)  
VALUES  
  (1, 1, 2, 1300, 'new'),  
  (2, 18, 1, 10000, 'cancelled'),  
  (3, 11, 1, 2140, 'in_progress'),  
  (4, 145, 5, 6800, 'new'),  
  (5, 23, 1, 999, 'new'),  
  (6, 1, 2, 7690, 'cancelled'),  
  (7, 17, 1, 1600, 'new'),  
  (8, 5, 4, 400, 'delivery'),  
  (9, 2355, 1, 1450, 'new'),  
  (10, 13, 7, 13000, 'cancelled');
```

id	user_id	products_count	sum	status
4	145	5	6800	new
3	11	1	2140	in_progress
7	17	1	1600	new
9	2355	1	1450	new
1	1	2	1300	new

Affected rows: 5

Решение:

SQLQuery3.sql - 100...ty (ITSTEP\qw (53))*

```
SELECT TOP 5 * FROM orders
WHERE status NOT IN ('cancelled')
ORDER BY sum DESC
```

100 %

Результаты Сообщения

	id	user_id	products_count	sum	status
1	4	145	5	6800	new
2	3	11	1	2140	in_progress
3	7	17	1	1600	new
4	9	2355	1	1450	new
5	1	1	2	1300	new

Номер 2

Выберите из таблицы **products** название и цены **трех** самых дешевых товаров, которые есть на складе.

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740
7	Тостер	2	2500
8	Принтер	4	3000
9	Активные колонки	1	2900

```
CREATE TABLE products (  
  id INT NOT NULL PRIMARY KEY,  
  name VARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
INSERT INTO products (id, name, count, price)  
VALUES  
  (1, 'Стиральная машина', 5, 10000),  
  (2, 'Холодильник', 0, 10000),  
  (3, 'Микроволновка', 3, 4000),  
  (4, 'Пылесос', 2, 4500),  
  (5, 'Вентилятор', 0, 700),  
  (6, 'Телевизор', 7, 31740),  
  (7, 'Тостер', 2, 2500),  
  (8, 'Принтер', 4, 3000),  
  (9, 'Активные колонки', 1, 2900);
```

name	price
Тостер	2500
Активные колонки	2900
Принтер	3000

Affected rows: 3

Решение:

SQLQuery1.sql - PT...y (PTG\Павел (56))*

```
SELECT TOP 3 * FROM products
WHERE count <> 0
ORDER BY price ASC
```

100 %

Результаты Сообщения

	id	name	count	price
1	7	Тостер	2	2500
2	9	Активные колонки	1	2900
3	8	Принтер	4	3000

Номер 3

Выберите из таблицы **orders** три последних заказа (по дате **date**) стоимостью от 3000 рублей и выше.

Данные отсортируйте по дате в обратном порядке.

orders					
id	user_id	products_count	sum	status	date
1	1	2	1300	new	2017-01-02
2	18	1	10000	cancelled	2017-01-02
3	11	1	2140	in_progress	2017-01-03
4	145	5	6800	new	2017-01-06
5	23	1	999	new	2017-01-09
6	1	2	7690	cancelled	2017-01-16
7	17	1	1600	new	2017-01-27
8	5	4	400	delivery	2017-02-01
9	2355	1	1450	new	2017-02-02
10	13	7	13000	cancelled	2017-02-02

```
CREATE TABLE orders (  
  id INT NOT NULL PRIMARY KEY,  
  user_id INTEGER NULL,  
  products_count INTEGER NULL,  
  sum INTEGER NULL,  
  status VARCHAR(20) NULL,  
  date DATE NULL  
);  
INSERT INTO orders (id, user_id, products_count, sum, status, date)  
VALUES  
  (1, 1, 2, 1300, 'new', '2017-01-02'),  
  (2, 18, 1, 10000, 'cancelled', '2017-01-02'),  
  (3, 11, 1, 2140, 'in_progress', '2017-01-03'),  
  (4, 145, 5, 6800, 'new', '2017-01-06'),  
  (5, 23, 1, 999, 'new', '2017-01-09'),  
  (6, 1, 2, 7690, 'cancelled', '2017-01-16'),  
  (7, 17, 1, 1600, 'new', '2017-01-27'),  
  (8, 5, 4, 400, 'delivery', '2017-02-01'),  
  (9, 2355, 1, 1450, 'new', '2017-02-02'),  
  (10, 13, 7, 13000, 'cancelled', '2017-02-02'),  
  (11, 23, 6, 3000, 'new', '2017-02-03');
```

id	user_id	products_count	sum	status	date
10	13	7	13000	cancelled	2017-02-02
6	1	2	7690	cancelled	2017-01-16
4	145	5	6800	new	2017-01-06

Affected rows: 3

Решение:

SQLQuery3.sql - 100...ty (ITSTEP\qw (53))*

```

SELECT top 3 * FROM orders
WHERE sum >= 3000
order by date desc

```

100 %

Результаты Сообщения

	id	user_id	products_count	sum	status	date
1	11	23	6	3000	new	2017-02-03
2	10	13	7	13000	cancelled	2017-02-02
3	6	1	2	7690	cancelled	2017-01-16

Номер 4

Сайт выводит товары по 5 штук. Выберите из таблицы **products** товары, которые пользователи увидят на 3 странице каталога при сортировке в порядке возрастания цены (**price**).

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740
7	Тостер	2	2500
8	Принтер	4	3000
9	Активные колонки	1	2900
10	Ноутбук	4	36990
11	Посудомоечная машина	0	17800
12	Видеорегистратор	23	4000
13	Смартфон	8	12300
14	Флешка	4	1400
15	Блендер	0	5500
16	Газовая плита	5	11900
17	Клавиатура	3	1800

```
CREATE TABLE products (  
  id INT NOT NULL PRIMARY KEY,  
  name VARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
INSERT INTO products (id, name, count, price)
```

VALUES

```
(1, 'Стиральная машина', 5, 10000),  
(2, 'Холодильник', 0, 10000),  
(3, 'Микроволновка', 3, 4000),  
(4, 'Пылесос', 2, 4500),  
(5, 'Вентилятор', 0, 700),  
(6, 'Телевизор', 7, 31740),  
(7, 'Тостер', 2, 2500),  
(8, 'Принтер', 4, 3000),  
(9, 'Активные колонки', 1, 2900),  
(10, 'Ноутбук', 4, 36990),  
(11, 'Посудомоечная машина', 0, 17800),  
(12, 'Видеорегистратор', 23, 4000),  
(13, 'Смартфон', 8, 12300),  
(14, 'Флешка', 4, 1400),  
(15, 'Блендер', 0, 5500),  
(16, 'Газовая плита', 5, 11900),  
(17, 'Клавиатура', 3, 1800);
```

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
16	Газовая плита	5	11900
13	Смартфон	8	12300
11	Посудомоечная машина	0	17800

Affected rows: 5

Решение:

SQLQuery1.sql - PT...y (PTG\Павел (56))*

```
SELECT * FROM products  
ORDER BY price ASC  
OFFSET 10 rows  
FETCH next 5 rows only
```

100 %

Результаты Сообщения

	id	name	count	price
1	2	Холодильник	0	10000
2	1	Стиральная машина	5	10000
3	16	Газовая плита	5	11900
4	13	Смартфон	8	12300
5	11	Посудомоечная машина	0	17800

Номер 5

В таблице **products** 17 записей. Сайт выводит название (**name**) и цену (**price**) товаров в алфавитном порядке, по 6 записей на страницу. Напишите SQL запрос для получения списка товаров для формирования последней страницы каталога.

Товары, которых нет на складе, выводить не надо (таких товаров 3).

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	2	700
6	Телевизор	7	31740
7	Тостер	2	2500
8	Принтер	4	3000
9	Активные колонки	1	2900
10	Ноутбук	4	36990
11	Посудомоечная машина	0	17800
12	Видеорегистратор	23	4000
13	Смартфон	8	12300
14	Флешка	4	1400
15	Блендер	0	5500
16	Газовая плита	5	11900
17	Клавиатура	3	1800

```
CREATE TABLE products (  
  id INT NOT NULL PRIMARY KEY,  
  name VARCHAR(255) NULL,
```

```

count INTEGER NULL,
price INTEGER NULL
);
INSERT INTO products (id, name, count, price)
VALUES
(1, 'Стиральная машина', 5, 10000),
(2, 'Холодильник', 0, 10000),
(3, 'Микроволновка', 3, 4000),
(4, 'Пылесос', 2, 4500),
(5, 'Вентилятор', 2, 700),
(6, 'Телевизор', 7, 31740),
(7, 'Тостер', 2, 2500),
(8, 'Принтер', 4, 3000),
(9, 'Активные колонки', 1, 2900),
(10, 'Ноутбук', 4, 36990),
(11, 'Посудомоечная машина', 0, 17800),
(12, 'Видеорегистратор', 23, 4000),
(13, 'Смартфон', 8, 12300),
(14, 'Флешка', 4, 1400),
(15, 'Блендер', 0, 5500),
(16, 'Газовая плита', 5, 11900),
(17, 'Клавиатура', 3, 1800);

```

name	price
Тостер	2500
Флешка	1400

Affected rows: 2

Решение:

SQLQuery3.sql - 100...ty (ITSTEP\qw (53))*

```

SELECT name, price FROM products
WHERE count <> 0
ORDER BY name
offset 11 rows

```

100 %

Результаты Сообщения

	name	price
1	Тостер	2500
2	Флешка	1400

В конце каждого номера нужно вывести получившуюся таблицу на экран

Номер 1

Добавьте в таблицу **orders** данные о новом заказе стоимостью 3000 рублей. В заказе 3 товара (**products**).

orders (исходная
таблица)

id	products	sum
1	2	1300
2	1	10000
3	1	2140
4	5	6800
5	1	999

```
CREATE TABLE orders (  
  id INT NOT NULL PRIMARY KEY,  
  products INTEGER NULL,  
  sum INTEGER NULL
```

```
);
```

```
INSERT INTO orders (id, products, sum)
```

```
VALUES
```

```
(1, 2, 1300),
```

```
(2, 1, 10000),
```

```
(3, 1, 2140),
```

```
(4, 5, 6800),
```

```
(5, 1, 999);
```

The screenshot shows a SQL query editor window titled "SQLQuery3.sql - 100...ty (ITSTEP\qw (53))*". The query entered is:

```
INSERT INTO orders (id, products, sum)  
VALUES (6, 3, 3000);  
  
select * from orders;
```

Below the query editor, there is a tab labeled "Результаты" (Results) which displays the updated table. The table has 6 rows and 3 columns: id, products, and sum. The first row (id=1) is highlighted with a blue selection box.

	id	products	sum
1	1	2	1300
2	2	1	10000
3	3	1	2140
4	4	5	6800
5	5	1	999
6	6	3	3000

Номер 2

Добавьте в таблицу **products** новый товар — «Xbox», стоимостью 30000 рублей в количестве (**count**) трех штук.

products (исходная таблица)

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740

```
CREATE TABLE products (  
    id INT NOT NULL PRIMARY KEY,  
    name NVARCHAR(255) NULL,  
    count INTEGER NULL,  
    price INTEGER NULL  
);  
INSERT INTO products (id, name, count, price)  
VALUES  
    (1, 'Стиральная машина', 5, 10000),  
    (2, 'Холодильник', 0, 10000),  
    (3, 'Микроволновка', 3, 4000),  
    (4, 'Пылесос', 2, 4500),  
    (5, 'Вентилятор', 0, 700),  
    (6, 'Телевизор', 7, 31740);
```

Решение:

```
INSERT INTO products (id, name, count, price)  
VALUES (7, 'Xbox', 3, 30000)
```

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740
7	Xbox	3	30000

Номер 3

Добавьте в таблицу **products** новый товар — «iMac 21», стоимостью 100100 рублей. Товар пока не завезли на склад.

products (исходная таблица)

id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740
7	Xbox	3	30000

```
CREATE TABLE products (  
  id INT NOT NULL PRIMARY KEY,  
  name NVARCHAR(255) NULL,  
  count INTEGER NULL,  
  price INTEGER NULL  
);  
INSERT INTO products (id, name, count, price)  
VALUES  
  (1, 'Стиральная машина', 5, 10000),  
  (2, 'Холодильник', 0, 10000),  
  (3, 'Микроволновка', 3, 4000),  
  (4, 'Пылесос', 2, 4500),  
  (5, 'Вентилятор', 0, 700),  
  (6, 'Телевизор', 7, 31740),  
  (7, 'Xbox', 3, 30000);
```

Решение:

```
INSERT INTO products (id, name, count, price)  
VALUES (8, 'iMac 21', 0, 100100);
```

Результаты		Сообщения		
	id	name	count	price
1	1	Стиральная машина	5	10000
2	2	Холодильник	0	10000
3	3	Микроволновка	3	4000
4	4	Пылесос	2	4500
5	5	Вентилятор	0	700
6	6	Телевизор	7	31740
7	7	Xbox	3	30000
8	8	iMac 21	0	100100

Номер 4

Добавьте в таблицу **users** нового пользователя Антона Пепеляева с датой рождения 12 июля 1992 года

users (исходная таблица)

id	first_name	last_name	birthday
1	Дмитрий	Петров	2000-03-14
2	Ольга	Антонова	1999-12-01
3	Сергей	Васильев	2002-02-20
4	Константин	Степаниденко	2004-03-07
5	Алена	Шикова	1999-08-17
6	Василина	Парамонова	2000-10-10
7	Александр	Пузаков	2002-02-20
8	Алина	Антонова	2002-01-01

```
CREATE TABLE users (  
  id INT,  
  first_name NVARCHAR(50) NULL,  
  last_name NVARCHAR(50) NULL,  
  birthday DATE NULL  
);  
INSERT INTO users (id, first_name, last_name, birthday)  
VALUES  
  (1, 'Дмитрий', 'Петров', '2000-03-14'),  
  (2, 'Ольга', 'Антонова', '1999-12-01'),  
  (3, 'Сергей', 'Васильев', '2002-02-20'),  
  (4, 'Константин', 'Степаниденко', '2004-03-07'),  
  (5, 'Алена', 'Шикова', '1999-08-17'),  
  (6, 'Василина', 'Парамонова', '2000-10-10'),  
  (7, 'Александр', 'Пузаков', '2002-02-20'),  
  (8, 'Алина', 'Антонова', '2002-01-01');
```

Решение:

```
INSERT INTO users (id, first_name, last_name, birthday)  
VALUES (9, 'Антон', 'Пепеляев', '1992-07-12')
```

Результаты		Сообщения		
	id	first_name	last_name	birthday
1	1	Дмитрий	Петров	2000-03-14
2	2	Ольга	Антонова	1999-12-01
3	3	Сергей	Васильев	2002-02-20
4	4	Константин	Степаниденко	2004-03-07
5	5	Алена	Шикова	1999-08-17
6	6	Василина	Парамонова	2000-10-10
7	7	Александр	Пузаков	2002-02-20
8	8	Алина	Антонова	2002-01-01
9	9	Антон	Пепеляев	1992-07-12

Номер 5

Новые записи в таблицу можно добавить не только с помощью **VALUES**, но и с помощью **SET**.

Следующие два запроса идентичны:

```
INSERT INTO table (field1, field2) VALUES (value1, value2);
```

```
INSERT INTO table SET field1=value1, field2=value2;
```

Добавьте в таблицу **users** нового пользователя Никиту Петрова. Дату рождения не указывайте.

Используйте ключевое слово **SET**.

users (исходная таблица)

id	first_name	last_name	birthday
1	Дмитрий	Петров	2000-03-14
2	Ольга	Антонова	1999-12-01
3	Сергей	Васильев	2002-02-20
4	Константин	Степаниденко	null
5	Алена	Шикова	1999-08-17
6	Василина	Парамонова	2000-10-10
7	Александр	Пузаков	2002-02-20
8	Алина	Антонова	2002-01-01
9	Антон	Пепеляев	1992-07-12

```
CREATE TABLE users (  
    id INT,  
    first_name NVARCHAR(50) NULL,  
    last_name NVARCHAR(50) NULL,  
    birthday DATE NULL  
);  
INSERT INTO users (id, first_name, last_name, birthday)  
VALUES  
    (1, 'Дмитрий', 'Петров', '2000-03-14'),  
    (2, 'Ольга', 'Антонова', '1999-12-01'),  
    (3, 'Сергей', 'Васильев', '2002-02-20'),  
    (4, 'Константин', 'Степаниденко', NULL),  
    (5, 'Алена', 'Шикова', '1999-08-17'),  
    (6, 'Василина', 'Парамонова', '2000-10-10'),  
    (7, 'Александр', 'Пузаков', '2002-02-20'),  
    (8, 'Алина', 'Антонова', '2002-01-01'),
```

```
(9, 'Антон', 'Пепеляев', '1992-07-12');
```

Решение: мне не удалось именно добавить Никиту с помощью SET, однако я нашёл информацию что SET работает только совместно с UPDATE, поэтому я сначала добавил запись, а потом изменил её с помощью SET.

```
INSERT INTO users (id, first_name, last_name)
VALUES (10, 'Кибита', 'Ветров');
```

id	first_name	last_name	birthday
1	Дмитрий	Петров	2000-03-14
2	Ольга	Антонова	1999-12-01
3	Сергей	Васильев	2002-02-20
4	Константин	Степаниденко	<i>null</i>
5	Алена	Шикова	1999-08-17
6	Василина	Парамонова	2000-10-10
7	Александр	Пузаков	2002-02-20
8	Алина	Антонова	2002-01-01
9	Антон	Пепеляев	1992-07-12
10	Кибита	Ветров	<i>null</i>

```
UPDATE users
SET first_name = 'Никита', last_name = 'Петров'
WHERE id =10;
```

Number of Records: 10

id	first_name	last_name	birthday
1	Дмитрий	Петров	2000-03-14
2	Ольга	Антонова	1999-12-01
3	Сергей	Васильев	2002-02-20
4	Константин	Степаниденко	<i>null</i>
5	Алена	Шикова	1999-08-17
6	Василина	Парамонова	2000-10-10
7	Александр	Пузаков	2002-02-20
8	Алина	Антонова	2002-01-01
9	Антон	Пепеляев	1992-07-12
10	Никита	Петров	<i>null</i>

Номер 6

Записи в таблицу можно добавлять не по одной, а сразу по несколько (в пакетном режиме).

Для этого нужно вставить несколько блоков значений, разделенных запятыми.

Следующий запрос добавит сразу три записи:

```
INSERT INTO table (field1, field2)
VALUES
(value1_1, value1_2),
(value2_1, value2_2),

(value3_1, value3_2);
```

Добавьте одним SQL запросом в таблицу **products** следующие товары:

* iPhone 7, цена 59990, 1 шт.

* iPhone 8, цена 64990, 3 шт.

* iPhone X, цена 79900, 2 шт.

products (исходная таблица)

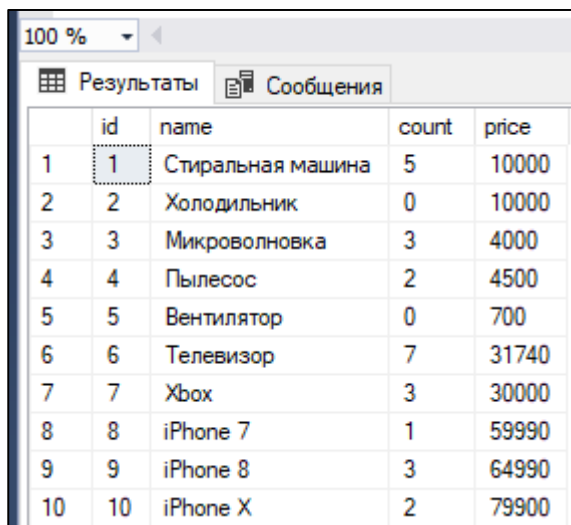
id	name	count	price
1	Стиральная машина	5	10000
2	Холодильник	0	10000
3	Микроволновка	3	4000
4	Пылесос	2	4500
5	Вентилятор	0	700
6	Телевизор	7	31740
7	Xbox	3	30000

```
CREATE TABLE products (
  id INT UNSIGNED,
  name NVARCHAR(255) NULL,
  count INTEGER NULL,
  price INTEGER NULL
);
INSERT INTO products (id, name, count, price)
VALUES
(1, 'Стиральная машина', 5, 10000),
(2, 'Холодильник', 0, 10000),
```

```
(3, 'Микроволновка', 3, 4000),  
(4, 'Пылесос', 2, 4500),  
(5, 'Вентилятор', 0, 700),  
(6, 'Телевизор', 7, 31740),  
(7, 'Xbox', 3, 30000);
```

Решение:

```
INSERT INTO products (id, name, count, price)  
VALUES  
    (8, 'iPhone 7', 1, 59990),  
    (9, 'iPhone 8', 3, 64990),  
    (10, 'iPhone X', 2, 79900);
```



The screenshot shows a database query result window. At the top, there is a zoom level dropdown set to '100 %' and a back arrow. Below this are two tabs: 'Результаты' (Results) and 'Сообщения' (Messages). The 'Результаты' tab is active, displaying a table with 5 columns: 'id', 'name', 'count', and 'price'. The table contains 10 rows of data, with the first row (id=1) highlighted. The data is as follows:

	id	name	count	price
1	1	Стиральная машина	5	10000
2	2	Холодильник	0	10000
3	3	Микроволновка	3	4000
4	4	Пылесос	2	4500
5	5	Вентилятор	0	700
6	6	Телевизор	7	31740
7	7	Xbox	3	30000
8	8	iPhone 7	1	59990
9	9	iPhone 8	3	64990
10	10	iPhone X	2	79900