

The task:

Create a database on MySQL, create several tables, create, modify, delete the records.
Create simple and complex queries.

Create database: **shops**

Tables:

- Rozetka (id_r, name_r, id_com, price_r)
- Allo (id_a, name_a, id_com, price_a)
- Company (id_c, name_c)

Rozetka:

- Kettle, Bosch, 1500
- TV, Samsung, 5000
- Smartphone, Samsung, 15,000
- Smartphone, Apple, 20,000
- Blender, Bosch, 500

Allo:

- TV, Samsung, 10000
- TV, Apple, 20,000
- Kettle, Samsung, 250
- Blender, Samsung, 600
- Smartphone, Apple, 15,000

Company:

- Bosch
- Samsung
- Magic
- Apple

A task:

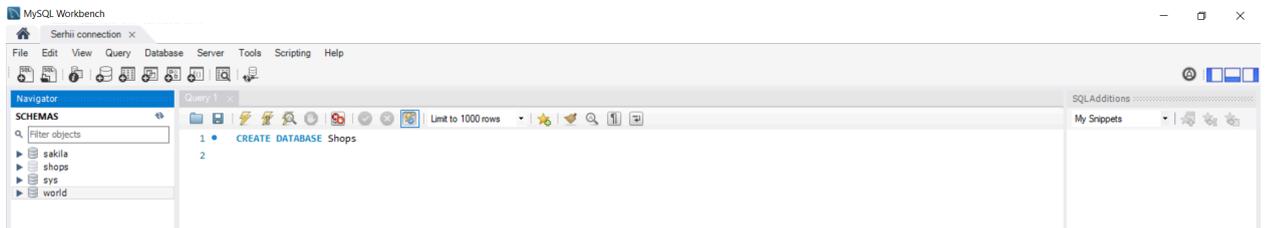
- Make a simple selection for each table
- Sort each table by company
- Group table by name
- Select the maximum price value for each table
- Calculate the total cost of goods in each table
- Make a selection of all smartphones from all tables (product name, price, company)
- Make a selection of all Bosch products for all products table
- Select Samsung products from all tables where price more than 600
- Select smartphones worth more than 15,000

Language of execution: English.

Task execution

Step 1. Creation the "Shops" database

```
CREATE DATABASE Shops
```



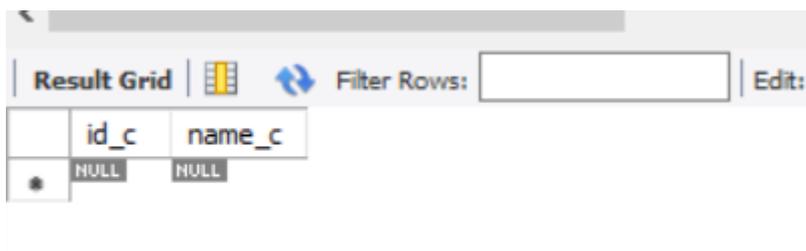
Step2. USE Shops

```
USE Shops;
```

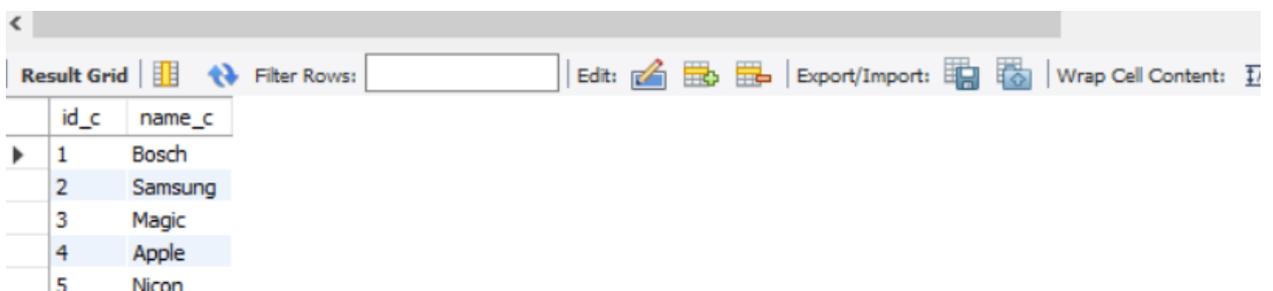
Step 3. Creation tables

Table 1 (Company)

```
CREATE TABLE Company (  
id_c int NOT NULL AUTO_INCREMENT,  
name_c varchar(20),  
PRIMARY KEY (id_c));
```



```
INSERT INTO Company (name_c) VALUES  
(  
'Bosch'),  
(  
'Samsung'),  
(  
'Magic'),  
(  
'Apple'),  
(  
'Nicon');
```



UPDATE Company SET name_c = 'Canon' WHERE id_c = 5

A screenshot of a database result grid. The grid has two columns: 'id_c' and 'name_c'. The rows are: 1 Bosch, 2 Samsung, 3 Magic, 4 Apple, 5 Canon, and a row with NULL values. The grid is titled 'Result Grid' and has a 'Filter Rows:' input field.

	id_c	name_c
▶	1	Bosch
	2	Samsung
	3	Magic
	4	Apple
	5	Canon
•	NULL	NULL

Table 2 (Rozetka)

```
CREATE TABLE Rozetka (  
id_r int NOT NULL AUTO_INCREMENT PRIMARY KEY,  
name_r varchar(20),  
id_com int,  
price_r decimal,  
FOREIGN KEY (id_com) REFERENCES Company(id_c));
```

A screenshot of a database result grid. The grid has four columns: 'id_r', 'name_r', 'id_com', and 'price_r'. The row contains NULL values for all columns. The grid is titled 'Result Grid' and has a 'Filter Rows:' input field and an 'Edit:' button.

	id_r	name_r	id_com	price_r
•	NULL	NULL	NULL	NULL

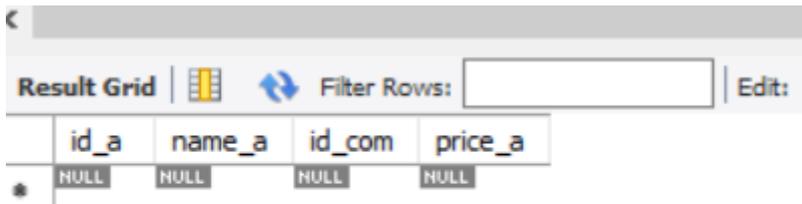
```
INSERT INTO Rozetka (name_r, id_com, price_r) VALUES  
( 'Kettle', 1, 1500),  
( 'TV', 2, 5000),  
( 'Smartphone', 2, 15000),  
( 'Smartphone', 4, 20000),  
( 'Blender', 1, 500);
```

A screenshot of a database result grid. The grid has four columns: 'id_r', 'name_r', 'id_com', and 'price_r'. The rows are: 1 Kettle, 2 TV, 3 Smartphone, 4 Smartphone, 5 Blender, and a row with NULL values. The grid is titled 'Result Grid' and has a 'Filter Rows:' input field.

	id_r	name_r	id_com	price_r
▶	1	Kettle	1	1500
	2	TV	2	5000
	3	Smartphone	2	15000
	4	Smartphone	4	20000
	5	Blender	1	500
•	NULL	NULL	NULL	NULL

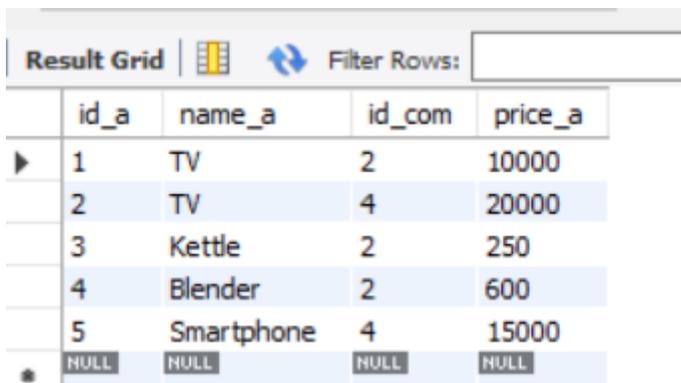
Table 3 (Allo)

```
CREATE TABLE Allo (  
id_a int NOT NULL AUTO_INCREMENT PRIMARY KEY,  
name_a varchar(20),  
id_com int,  
price_a decimal,  
FOREIGN KEY (id_com) REFERENCES Company(id_c));
```



	id_a	name_a	id_com	price_a
*	NULL	NULL	NULL	NULL

```
INSERT INTO Allo (name_a, id_com, price_a) VALUES  
( 'TV', 2, 10000),  
( 'TV', 4, 20000),  
( 'Kettle', 2, 250),  
( 'Blender', 2, 600),  
( 'Smartphone', 4, 15000);
```



	id_a	name_a	id_com	price_a
▶	1	TV	2	10000
	2	TV	4	20000
	3	Kettle	2	250
	4	Blender	2	600
	5	Smartphone	4	15000
*	NULL	NULL	NULL	NULL

Table 4 (Rozetka2)

```
CREATE TABLE Rozetka (  
id_r int NOT NULL AUTO_INCREMENT PRIMARY KEY,  
name_r varchar(20),  
id_com int,  
price_r decimal,  
FOREIGN KEY (id_com) REFERENCES Company(id_c));
```

Step 4. Deleting the records

Delete a row from the Company table

```
DELETE FROM Company WHERE id=5;
```

	id_c	name_c
▶	1	Bosch
	2	Samsung
	3	Magic
	4	Apple
•	NULL	NULL

Delete the Rozetka 2 table

DROP TABLE Rozetka2;

Step 5. Making a simple selection for each table

The Company table

SELECT * FROM Company;

	id_c	name_c
▶	1	Bosch
	2	Samsung
	3	Magic
	4	Apple
•	NULL	NULL

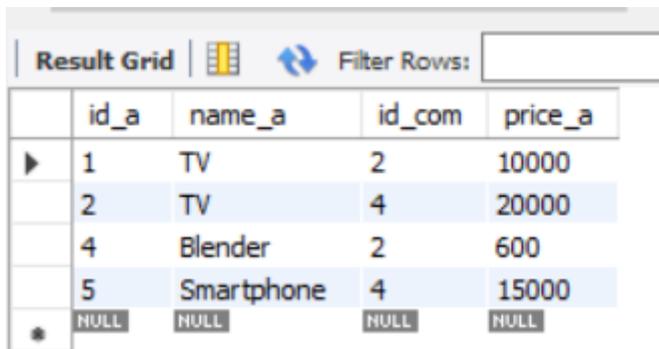
The Rozetka table

SELECT name_r FROM Rozetka WHERE id_r BETWEEN 2 AND 5;

	name_r
▶	TV
	Smartphone
	Smartphone
	Blender

The Allo table

```
SELECT * FROM Allo WHERE price_a > 500;
```



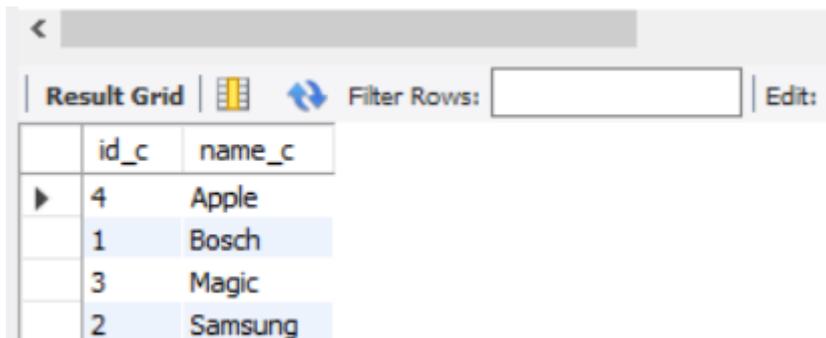
The screenshot shows a 'Result Grid' interface with a 'Filter Rows' input field. The table below displays the results of the query, showing items with a price greater than 500. The columns are id_a, name_a, id_com, and price_a.

	id_a	name_a	id_com	price_a
▶	1	TV	2	10000
	2	TV	4	20000
	4	Blender	2	600
	5	Smartphone	4	15000
*	NULL	NULL	NULL	NULL

Step 6. Sorting each table by company

The Company table

```
SELECT * FROM Company  
ORDER by name_c;
```

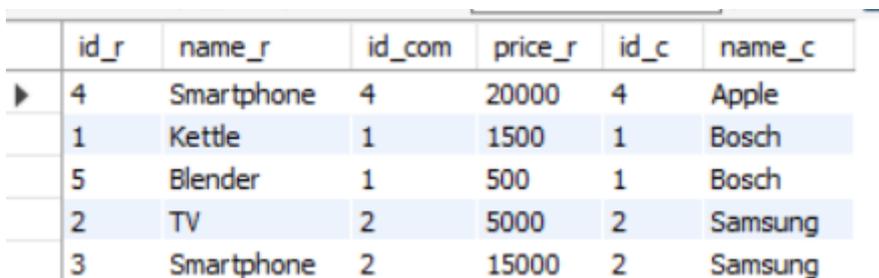


The screenshot shows a 'Result Grid' interface with a 'Filter Rows' input field and an 'Edit' button. The table below displays the results of the query, showing companies sorted by name. The columns are id_c and name_c.

	id_c	name_c
▶	4	Apple
	1	Bosch
	3	Magic
	2	Samsung

The Rozetka table

```
SELECT * FROM Rozetka  
INNER JOIN Company ON Rozetka.id_com = Company.id_c  
ORDER BY name_c;
```



The screenshot shows a 'Result Grid' interface with a 'Filter Rows' input field. The table below displays the results of the query, showing items from the Rozetka table joined with the Company table. The columns are id_r, name_r, id_com, price_r, id_c, and name_c.

	id_r	name_r	id_com	price_r	id_c	name_c
▶	4	Smartphone	4	20000	4	Apple
	1	Kettle	1	1500	1	Bosch
	5	Blender	1	500	1	Bosch
	2	TV	2	5000	2	Samsung
	3	Smartphone	2	15000	2	Samsung

The Allo table

```
SELECT * FROM Allo  
INNER JOIN Company ON Allo.id_com = Company.id_c  
ORDER BY name_c DESC;
```

	id_a	name_a	id_com	price_a	id_c	name_c
▶	1	TV	2	10000	2	Samsung
	3	Kettle	2	250	2	Samsung
	4	Blender	2	600	2	Samsung
	2	TV	4	20000	4	Apple
	5	Smartphone	4	15000	4	Apple

Step 7. Grouping table by name

The Company table

```
SELECT COUNT(id_c),name_c FROM Company
GROUP BY name_c;
```

	COUNT(id_c)	name_c
▶	1	Bosch
	1	Samsung
	1	Magic
	1	Apple

The Rozetka table

```
SELECT COUNT(id_c), name_c FROM Company
JOIN Rozetka ON Rozetka.id_com = Company.id_c
GROUP BY name_c;
```

	COUNT(id_c)	name_c
▶	2	Bosch
	2	Samsung
	1	Apple

The Allo table

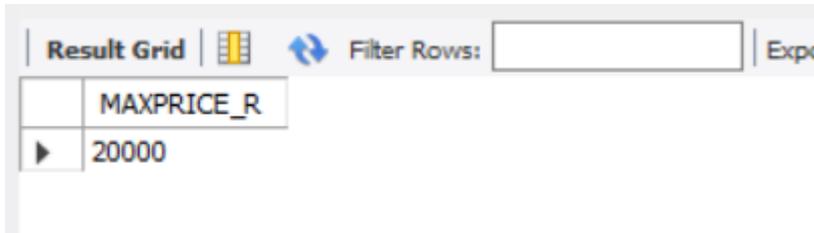
```
SELECT COUNT(name_a) FROM Allo GROUP BY name_a
HAVING name_a LIKE 'T%'
```

	COUNT(name_a)
▶	2

Step 8. Selecting the maximum price value for each table

The Rozetka table

```
SELECT MAX(price_r) AS MAXPRICE_R FROM Rozetka
```



	MAXPRICE_R
▶	20000

The Allo table

```
SELECT MAX(price_a) AS MAXPRICE_a FROM Allo
```

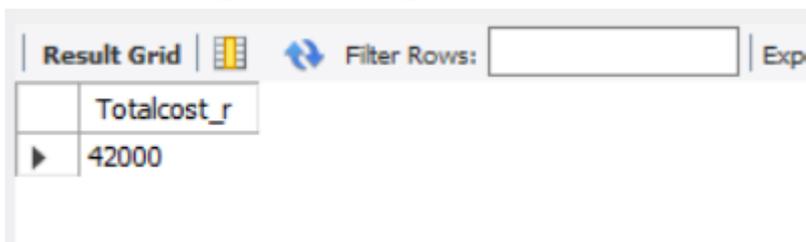


	MAXPRICE_a
▶	20000

Step 9. Calculating the total cost of goods in each table

The Rozetka table

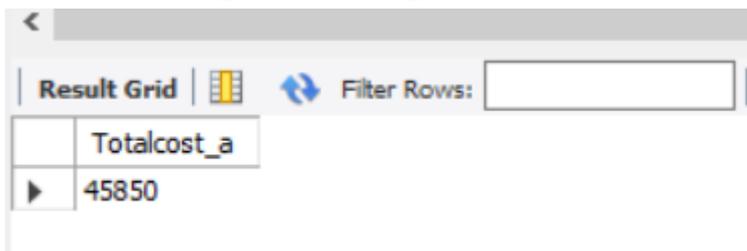
```
SELECT SUM(price_r) AS Totalcost_r FROM Rozetka;
```



	Totalcost_r
▶	42000

The Allo table

```
SELECT SUM(price_a) AS Totalcost_a FROM Allo
```

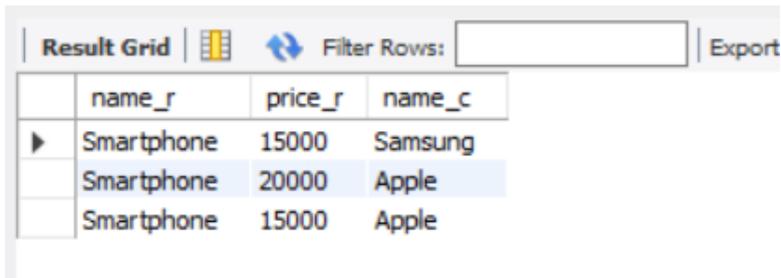


	Totalcost_a
▶	45850

Step 10. Making a selection of all smartphones from all tables (product name, price, company)

```
SELECT name_r, price_r, name_c FROM Rozetka JOIN Company ON Rozetka.id_com=Company.id_c  
WHERE rozetka.name_r LIKE 'Smartphone'  
UNION SELECT name_a, price_a, name_c FROM Allo JOIN Company ON Allo.id_com=Company.id_c
```

WHERE Allo.name_a LIKE 'Smartphone'

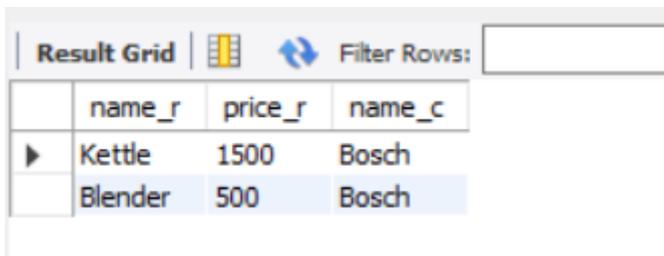


The screenshot shows a 'Result Grid' interface with a 'Filter Rows' input field and an 'Export' button. The table below displays the results of a query filtering for 'Smartphone'.

	name_r	price_r	name_c
▶	Smartphone	15000	Samsung
	Smartphone	20000	Apple
	Smartphone	15000	Apple

Step 11. Making a selection of all Bosch products for all products tables

```
SELECT name_r, price_r, name_c FROM Rozetka JOIN Company ON Rozetka.id_com=Company.id_c  
WHERE Company.name_c LIKE 'Bosch'  
UNION SELECT name_a, price_a, name_c FROM Allo JOIN Company ON Allo.id_com=Company.id_c  
WHERE Company.name_c LIKE 'Bosch'
```

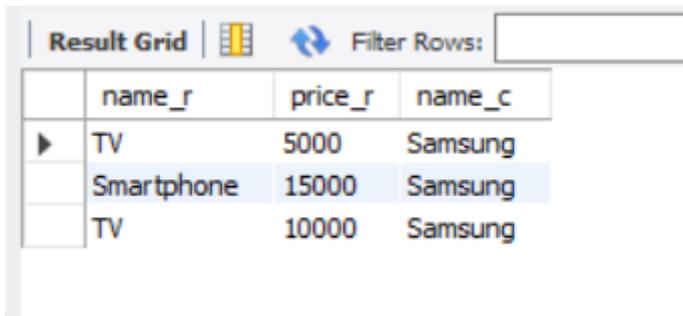


The screenshot shows a 'Result Grid' interface with a 'Filter Rows' input field. The table below displays the results of a query filtering for 'Bosch' products.

	name_r	price_r	name_c
▶	Kettle	1500	Bosch
	Blender	500	Bosch

Step 12. Selection Samsung products from all tables where price more than 600

```
SELECT name_r, price_r, name_c FROM Rozetka JOIN Company ON Rozetka.id_com=Company.id_c  
WHERE Company.name_c LIKE 'Samsung' AND Rozetka.price_r > 600  
UNION SELECT name_a, price_a, name_c FROM Allo JOIN Company ON Allo.id_com=Company.id_c  
WHERE Company.name_c LIKE 'Samsung' AND Allo.price_a > 600
```

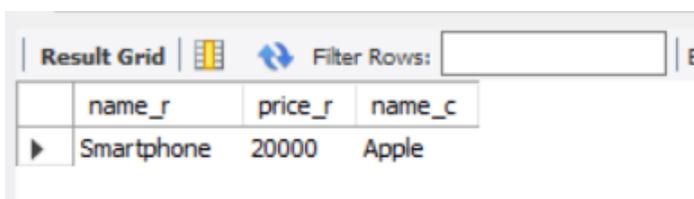


The screenshot shows a 'Result Grid' interface with a 'Filter Rows' input field. The table below displays the results of a query filtering for 'Samsung' products with a price greater than 600.

	name_r	price_r	name_c
▶	TV	5000	Samsung
	Smartphone	15000	Samsung
	TV	10000	Samsung

Step 13. Selection smartphones from all tables where price more than 15000

```
SELECT name_r, price_r, name_c FROM Rozetka JOIN Company ON Rozetka.id_com=Company.id_c  
WHERE name_r LIKE 'Smartphone' AND price_r > 15000  
UNION SELECT name_a, price_a, name_c FROM Allo JOIN Company ON Allo.id_com=Company.id_c  
WHERE name_a LIKE 'Smartphone' AND price_a > 15000
```



The screenshot shows a 'Result Grid' interface with a 'Filter Rows' input field. The table below displays the results of a query filtering for 'Smartphone' products with a price greater than 15000.

	name_r	price_r	name_c
▶	Smartphone	20000	Apple

