

LABORATORY WORK 3. DATA MANIPULATION USING SQL LANGUAGE: SELECT QUERIES AND THEIR BASIC FEATURES

Goal: learn how to use the SQL SELECT statement for data querying, using the MySQL database.

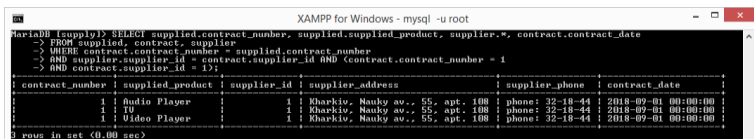
Progress

1. Create and execute SQL SELECT queries

Query 1

Form a list of goods delivered by supplier 1 (Petrov P. P.) under contract 1 (figure 3.1).

```
SELECT supplied.contract_number, supplied.supplied_product, supplier.*, contract.contract_date
FROM supplied, contract, supplier
WHERE contract.contract_number = supplied.contract_number
AND supplier.supplier_id = contract.supplier_id AND (contract.contract_number = 1
AND contract.supplier_id = 1);
```



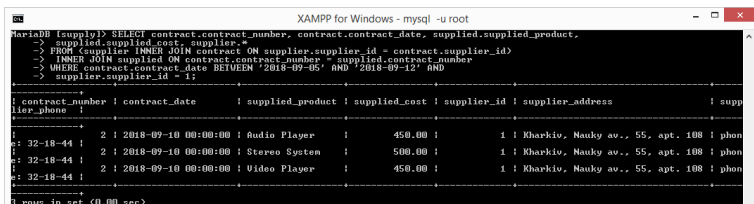
contract_number	supplied_product	supplier_id	supplier_address	supplier_phone	contract_date
1	Audio Player	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44	2018-09-01 00:00:00
1	TV	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44	2018-09-01 00:00:00
1	Video Player	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44	2018-09-01 00:00:00

Figure 3.1 – Result of query 1

Query 2

Form a list of the goods delivered by supplier 1 (Petrov P. P.) in the period from 2018-09-05 to 09/08/2012 (figure 3.2).

```
SELECT contract.contract_number, contract.contract_date, supplied.supplied_product,
supplied.supplied_cost, supplier.*
FROM (supplier INNER JOIN contract ON supplier.supplier_id = contract.supplier_id)
INNER JOIN supplied ON contract.contract_number = supplied.contract_number
WHERE contract.contract_date BETWEEN '2018-09-05' AND '2018-09-12' AND
supplier.supplier_id = 1;
```



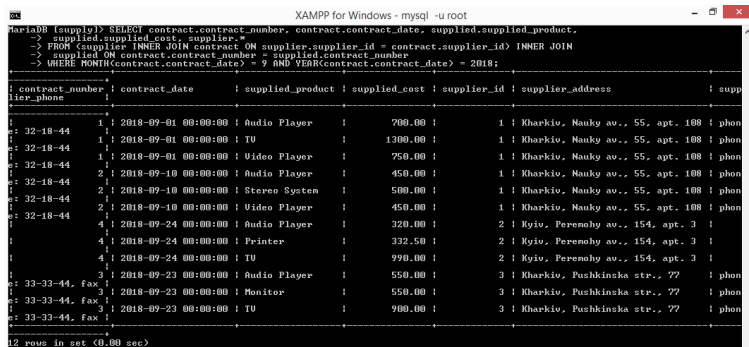
contract_number	contract_date	supplied_product	supplied_cost	supplier_id	supplier_address	supplier_phone
2	2018-09-10 00:00:00	Audio Player	450.00	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44
2	2018-09-10 00:00:00	Stereo System	500.00	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44
2	2018-09-10 00:00:00	Video Player	450.00	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44

Figure 3.2 – Result of query 2

Query 3

Form a list of goods that were delivered in month 9 of 2018 including the name of the supplier and delivery date (figure 3.3).

```
SELECT contract.contract_number, contract.contract_date, supplied.supplied_product,
       supplied.supplied_cost, supplier.*
FROM (supplier INNER JOIN contract ON supplier.supplier_id = contract.supplier_id) INNER JOIN
     supplied ON contract.contract_number = supplied.contract_number
WHERE MONTH(contract.contract_date) = 9 AND YEAR(contract.contract_date) = 2018;
```



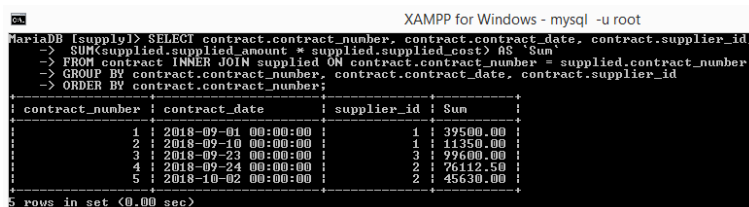
contract_number	contract_date	supplied_product	supplied_cost	supplier_id	supplier_address	supplier_phone
1	2018-09-01 00:00:00	Audio Player	700.00	1	Kharkiv, Nauky av., 55, apt. 108	phon
1	2018-09-01 00:00:00	TV	1300.00	1	Kharkiv, Nauky av., 55, apt. 108	phon
1	2018-09-01 00:00:00	Video Player	750.00	1	Kharkiv, Nauky av., 55, apt. 108	phon
2	2018-09-10 00:00:00	Audio Player	450.00	1	Kharkiv, Nauky av., 55, apt. 108	phon
2	2018-09-10 00:00:00	Stereo System	500.00	1	Kharkiv, Nauky av., 55, apt. 108	phon
2	2018-09-10 00:00:00	Video Player	450.00	1	Kharkiv, Nauky av., 55, apt. 108	phon
4	2018-09-24 00:00:00	Audio Player	320.00	2	Kyiv, Peremohy av., 154, apt. 3	phon
4	2018-09-24 00:00:00	Printer	332.50	2	Kyiv, Peremohy av., 154, apt. 3	phon
4	2018-09-24 00:00:00	TV	990.00	2	Kyiv, Peremohy av., 154, apt. 3	phon
3	2018-09-23 00:00:00	Audio Player	550.00	3	Kharkiv, Pushkinska str., 77	phon
3	2018-09-23 00:00:00	Monitor	550.00	3	Kharkiv, Pushkinska str., 77	phon
3	2018-09-23 00:00:00	TV	900.00	3	Kharkiv, Pushkinska str., 77	phon

Figure 3.3 – Result of query 3

Query 4

Form a list of contracts (number, date, title) and the total amount for each contract (batch size multiplied by the price per unit and summed up by the contract). The list should be sorted by contract numbers (figure 3.4).

```
SELECT contract.contract_number, contract.contract_date, contract.supplier_id,
       SUM(supplied.supplied_amount * supplied.supplied_cost) AS `Sum`
FROM contract INNER JOIN supplied ON contract.contract_number = supplied.contract_number
GROUP BY contract.contract_number, contract.contract_date, contract.supplier_id
ORDER BY contract.contract_number;
```



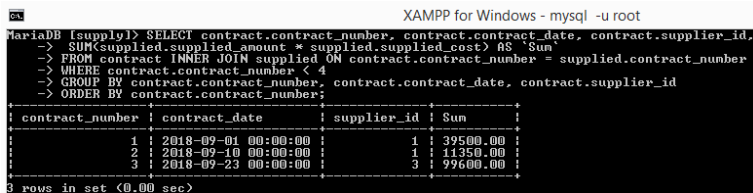
contract_number	contract_date	supplier_id	Sum
1	2018-09-01 00:00:00	1	39500.00
2	2018-09-10 00:00:00	1	11350.00
3	2018-09-23 00:00:00	3	92500.00
4	2018-09-24 00:00:00	2	76112.50
5	2018-10-02 00:00:00	2	45630.00

Figure 3.4 – Result of query 4

Query 5

Form a list of contracts (number, date, title) and the total amount for each contract (batch size multiplied by the price per unit and summed up by the contract). The list should be sorted by increasing the total amounts for each contract. After that, the filter must be applied to the list, in order to exclude from the result of the query those records for which the contract number is less than 4 (figure 3.5).

```
SELECT contract.contract_number, contract.contract_date, contract.supplier_id,  
       SUM(supplied.supplied_amount * supplied.supplied_cost) AS `Sum`  
FROM contract INNER JOIN supplied ON contract.contract_number = supplied.contract_number  
WHERE contract.contract_number < 4  
GROUP BY contract.contract_number, contract.contract_date, contract.supplier_id  
ORDER BY contract.contract_number;
```



contract_number	contract_date	supplier_id	Sum
1	2018-09-01 00:00:00	1	39500.00
2	2018-09-10 00:00:00	1	11350.00
3	2018-09-23 00:00:00	3	99600.00

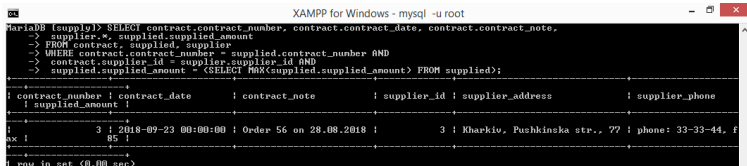
3 rows in set (0.00 sec)

Figure 3.5 – Result of query 5

Query 6

Display the information on the largest batch of goods in all contracts with the supplier, as well as the number and the date of the contract (figure 3.6).

```
SELECT contract.contract_number, contract.contract_date, contract.contract_note,  
       supplier.*, supplied.supplied_amount  
FROM contract, supplied, supplier  
WHERE contract.contract_number = supplied.contract_number AND  
       contract.supplier_id = supplier.supplier_id AND  
       supplied.supplied_amount = (SELECT MAX(supplied.supplied_amount) FROM supplied);
```



contract_number	contract_date	contract_note	supplier_id	supplier_address	supplier_phone
3	2018-09-23 00:00:00	Order 56 on 28.08.2018	3	Kharkiv, Pushkinska str., 77	phone: 33-33-44, f

1 row in set (0.00 sec)

Figure 3.6 – Result of query 6

Query 7

Form a list of suppliers (name and code) with which no contract has been concluded (figure 3.7).

```
SELECT * FROM supplier
WHERE supplier_id NOT IN (SELECT supplier_id FROM supplier);
```



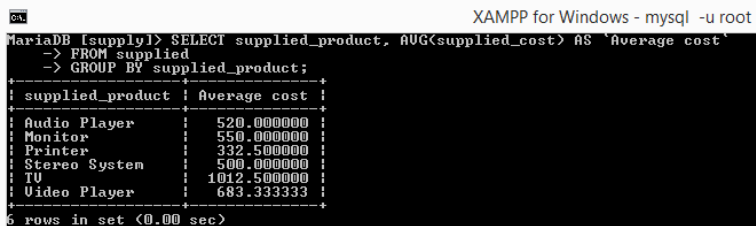
```
C:\XAMPP for Windows
MariaDB [supply]> SELECT * FROM supplier
-> WHERE supplier_id NOT IN (SELECT supplier_id FROM supplier);
Empty set (0.00 sec)
```

Figure 3.7 – Result of query 7

Query 8

Form a list of the names of supplied goods with an indication of the average delivery price per unit (regardless of the supplier) (figure 3.8).

```
SELECT supplied_product, AVG(supplied_cost) AS `Average cost`
FROM supplied
GROUP BY supplied_product;
```



```
C:\XAMPP for Windows - mysql -u root
MariaDB [supply]> SELECT supplied_product, AVG(supplied_cost) AS `Average cost`
-> FROM supplied
-> GROUP BY supplied_product;
+-----+-----+
| supplied_product | Average cost |
+-----+-----+
| Audio Player    | 520.000000   |
| Monitor         | 550.000000   |
| Printer         | 332.500000   |
| Stereo System   | 500.000000   |
| TV              | 1012.500000  |
| Video Player    | 683.333333   |
+-----+-----+
6 rows in set (0.00 sec)
```

Figure 3.8 – Result of query 8

Query 9

Form a list of goods (name, quantity and price, supplier), for which the price per unit is more than average (figure 3.9).

```
SELECT supplied_product, supplied_amount, supplied_cost, supplier.*
FROM (supplier INNER JOIN contract ON supplier.supplier_id = contract.supplier_id)
INNER JOIN supplied ON contract.contract_number = supplied.contract_number
WHERE supplied_cost > (SELECT AVG(supplied_cost) FROM supplied);
```

supplied_product	supplied_amount	supplied_cost	supplier_id	supplier_address	supplier_phone
Audio Player	25	780.00	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44
TU	10	1300.00	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44
Video Player	12	750.00	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44
TU	9	920.00	2	Kyiv, Peremohy av., 154, apt. 3	
TU	14	860.00	2	Kyiv, Peremohy av., 154, apt. 3	
Video Player	17	850.00	3	Kharkiv, Pushkinska str., 77	phone: 33-33-44, fax:

6 rows in set (0.00 sec)

Figure 3.9 – Result of query 9

Query 10

Display information about the five most expensive products (name, price per unit, supplier) (figure 3.10).

```
SELECT supplied_product, supplied_cost, supplier.*
FROM (supplier INNER JOIN contract ON supplier.supplier_id = contract.supplier_id)
INNER JOIN supplied ON contract.contract_number = supplied.contract_number
ORDER BY supplied_cost DESC
LIMIT 1;
```

supplied_product	supplied_cost	supplier_id	supplier_address	supplier_phone
TU	1300.00	1	Kharkiv, Nauky av., 55, apt. 108	phone: 32-18-44

1 row in set (0.00 sec)

Figure 3.10 – Result of query 10

Query 11

Form a supplier list with code, address, and supplier information. When forming supplier data for individuals, display the surname and initials, and for legal entities – the name (figure 3.11).

```
SELECT supplier.supplier_id, supplier.supplier_address,
IFNULL(supplier_org.supplier_org_name, CONCAT(RTRIM(supplier_person.supplier_last_name), ' ',
SUBSTRING(supplier_person.supplier_first_name, 1, 1), ' ',
SUBSTRING(supplier_person.supplier_middle_name, 1, 1), ' ') AS 'Supplier'
FROM (supplier LEFT JOIN supplier_person ON supplier.supplier_id = supplier_person.supplier_id)
LEFT JOIN supplier_org ON supplier.supplier_id = supplier_org.supplier_id;
```

supplier_id	supplier_address	Supplier
1	Kharkiv, Nauky av., 55, apt. 108	Petrov P. P.
2	Kyiv, Peremohy av., 154, apt. 3	Interfruit Ltd.
3	Kharkiv, Pushkinska str., 77	Ivanov I. I.
4	Odesa, Derebasivska str., 75	Transservice LLC
5	Poltava, Soborna str., 15, apt. 43	Sydorov S. S.

5 rows in set (0.00 sec)

Figure 3.11 – Result of query 11

Query 12

Form a list of contracts (include the number, delivery date, and supplier information), the total number of goods delivered and the total amount for each contract. When forming the supplier data for individuals, display the last name and initials, and for legal entities – the name. The result should contain only those contracts based on which the goods were delivered (e.g., the result of the query should not contain so-called “empty” contracts) (figure 3.12).

```
SELECT contract.contract_number, contract.contract_date,
       IFNULL(supplier_org.supplier_org_name, CONCAT(RTRIM(supplier_person.supplier_last_name), ' ',
       SUBSTRING(supplier_person.supplier_first_name, 1, 1), '. ')) AS 'Supplier',
       SUM(supplier.supplied_amount) AS 'Size',
       SUM(supplier.supplied_cost * supplier.supplied_amount) AS 'Total'
FROM ((supplier LEFT JOIN supplier_person ON supplier.supplier_id = supplier_person.supplier_id)
LEFT JOIN supplier_org ON supplier.supplier_id = supplier_org.supplier_id)
INNER JOIN contract ON contract.supplier_id = supplier.supplier_id)
INNER JOIN supplied ON contract.contract_number = supplied.contract_number
GROUP BY supplier.supplier_id, supplier.supplier_address,
       IFNULL(supplier_org.supplier_org_name, CONCAT(RTRIM(supplier_person.supplier_last_name), ' ',
       SUBSTRING(supplier_person.supplier_first_name, 1, 1), '. ')) AS 'Supplier',
       SUBSTRING(supplier_person.supplier_middle_name, 1, 1), '. '))
ORDER BY contract.contract_number;
```

Terminal window showing the execution of Query 12 in MySQL:

```
XAMPP for Windows - mysql -u root
MariaDB [supply]> SELECT contract.contract_number, contract.contract_date,
-> IFNULL(supplier_org.supplier_org_name, CONCAT(RTRIM(supplier_person.supplier_last_name), ' ',
-> SUBSTRING(supplier_person.supplier_first_name, 1, 1), '. ')) AS 'Supplier',
-> SUBSTRING(supplier_person.supplier_middle_name, 1, 1), '. ')) AS 'Supplier',
-> SUM(supplier.supplied_amount) AS 'Size',
-> SUM(supplier.supplied_cost * supplier.supplied_amount) AS 'Total'
-> FROM ((supplier LEFT JOIN supplier_person ON supplier.supplier_id = supplier_person.supplier_id)
-> LEFT JOIN supplier_org ON supplier.supplier_id = supplier_org.supplier_id)
-> INNER JOIN contract ON contract.supplier_id = supplier.supplier_id)
-> INNER JOIN supplied ON contract.contract_number = supplied.contract_number
-> GROUP BY supplier.supplier_id, supplier.supplier_address,
-> IFNULL(supplier_org.supplier_org_name, CONCAT(RTRIM(supplier_person.supplier_last_name), ' ',
-> SUBSTRING(supplier_person.supplier_first_name, 1, 1), '. ')) AS 'Supplier',
-> SUBSTRING(supplier_person.supplier_middle_name, 1, 1), '. '))
-> ORDER BY contract.contract_number;
+-----+-----+-----+-----+-----+
| contract_number | contract_date | Supplier | Size | Total |
+-----+-----+-----+-----+-----+
| 1 | 2018-09-01 00:00:00 | Petrov P. P. | 71 | 50850.00 |
| 3 | 2018-09-23 00:00:00 | Ivanov I. I. | 148 | 99600.00 |
| 4 | 2018-09-24 00:00:00 | Interfruit Ltd. | 183 | 121742.50 |
+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)
```

Figure 3.12 – Result of query 12

Query 13

Form a list of goods (with the number of the contract and delivery date) delivered by suppliers 1 (Petrov P. P.) and 2 (Interfruit) (figure 3.13).

```
SELECT supplied.contract_number, contract.contract_date,
       supplied.supplied_product, supplier.supplier_id
FROM supplied, contract, supplier
WHERE contract.contract_number = supplied.contract_number
AND supplier.supplier_id = contract.supplier_id AND contract.supplier_id = 1
UNION
SELECT supplied.contract_number, contract.contract_date,
       supplied.supplied_product, supplier.supplier_id
FROM supplied, contract, supplier
WHERE contract.contract_number = supplied.contract_number
AND supplier.supplier_id = contract.supplier_id AND contract.supplier_id = 2
ORDER BY supplier_id, contract_number;
```

```

XAMPP for Windows - mysql -u root
MariaDB [supply]> SELECT supplied.contract_number, contract.contract_date,
-> supplied.supplied_product, supplier.supplier_id
-> FROM supplied, contract, supplier
-> WHERE contract.contract_number = supplied.contract_number
-> AND supplier.supplier_id = contract.supplier_id AND contract.supplier_id = 1
-> UNION
-> SELECT supplied.contract_number, contract.contract_date,
-> supplied.supplied_product, supplier.supplier_id
-> FROM supplied, contract, supplier
-> WHERE contract.contract_number = supplied.contract_number
-> AND supplier.supplier_id = contract.supplier_id AND contract.supplier_id = 2
-> ORDER BY supplier_id, contract_number;
+-----+-----+-----+-----+
| contract_number | contract_date | supplied_product | supplier_id |
+-----+-----+-----+-----+
| 1 | 2018-09-01 00:00:00 | Audio Player | 1 |
| 1 | 2018-09-01 00:00:00 | TV | 1 |
| 1 | 2018-09-01 00:00:00 | Video Player | 1 |
| 2 | 2018-09-10 00:00:00 | Audio Player | 1 |
| 2 | 2018-09-10 00:00:00 | Stereo System | 1 |
| 2 | 2018-09-10 00:00:00 | Video Player | 1 |
| 4 | 2018-09-24 00:00:00 | Printer | 2 |
| 4 | 2018-09-24 00:00:00 | TV | 2 |
| 4 | 2018-09-24 00:00:00 | Audio Player | 2 |
| 5 | 2018-10-02 00:00:00 | Audio Player | 2 |
| 5 | 2018-10-02 00:00:00 | TV | 2 |
| 5 | 2018-10-02 00:00:00 | Video Player | 2 |
+-----+-----+-----+-----+
12 rows in set (0.00 sec)

```

Figure 3.13 – Result of query 13

Query 14

Form a nomenclature of goods (a list of product names) that were supplied only by supplier 1 (Petrov P. P.), or only supplier 2 (Interfruit), or both supplier 1 and supplier 2 (figure 3.14).

```

SELECT DISTINCT supplied.supplied_product
FROM supplied, contract
WHERE contract.contract_number = supplied.contract_number AND contract.supplier_id = 1
UNION
SELECT DISTINCT supplied.supplied_product
FROM supplied, contract
WHERE contract.contract_number = supplied.contract_number AND contract.supplier_id = 2
ORDER BY supplied_product;

```

```

XAMPP for Windows - mysql -u root
MariaDB [supply]> SELECT DISTINCT supplied.supplied_product
-> FROM supplied, contract
-> WHERE contract.contract_number = supplied.contract_number AND contract.supplier_id = 1
-> UNION
-> SELECT DISTINCT supplied.supplied_product
-> FROM supplied, contract
-> WHERE contract.contract_number = supplied.contract_number AND contract.supplier_id = 2
-> ORDER BY supplied_product;
+-----+
| supplied_product |
+-----+
| Audio Player |
| Printer |
| Stereo System |
| TV |
| Video Player |
+-----+
5 rows in set (0.00 sec)

```

Figure 3.14 – Result of query 14

Query 15

Generate a list of items that should demonstrate the frequency of deliveries. Include only items that shipped more than once to the list. The list should be sorted by decreasing the supply frequency (figure 3.15).

```
SELECT supplied_product, COUNT(supplied_product) AS `SupplyFrequency`
FROM supplied
GROUP BY supplied_product
HAVING COUNT(supplied_product) > 1
ORDER BY COUNT(supplied_product) DESC;
```

SQL XAMPP for Windows - mysql -u root

```
MariaDB [supplied]> SELECT supplied_product, COUNT(supplied_product) AS `SupplyFrequency`
-> FROM supplied
-> GROUP BY supplied_product
-> HAVING COUNT(supplied_product) > 1
-> ORDER BY COUNT(supplied_product) DESC;
```

supplied_product	SupplyFrequency
Audio Player	5
TU	4
Video Player	3

3 rows in set (0.00 sec)

Figure 3.15 – Result of query 15

Query 16

Retrieve data on quantitative dynamics of goods deliveries during 2018. The data should be aggregated in months and presented as a table with lines of product names, and columns are the numbers of months in 2018. At the intersection of a row and a column, the quantity of this product delivered in this month should be displayed (figure 3.16).

```
SELECT supplied_product, SUM(IF(MONTH(contract_date) = 1, supplied_amount, 0)) AS `Jan`,
SUM(IF(MONTH(contract_date) = 2, supplied_amount, 0)) AS `Feb`,
SUM(IF(MONTH(contract_date) = 3, supplied_amount, 0)) AS `Mar`,
SUM(IF(MONTH(contract_date) = 4, supplied_amount, 0)) AS `Apr`,
SUM(IF(MONTH(contract_date) = 5, supplied_amount, 0)) AS `May`,
SUM(IF(MONTH(contract_date) = 6, supplied_amount, 0)) AS `Jun`,
SUM(IF(MONTH(contract_date) = 7, supplied_amount, 0)) AS `Jul`,
SUM(IF(MONTH(contract_date) = 8, supplied_amount, 0)) AS `Aug`,
SUM(IF(MONTH(contract_date) = 9, supplied_amount, 0)) AS `Sep`,
SUM(IF(MONTH(contract_date) = 10, supplied_amount, 0)) AS `Oct`,
SUM(IF(MONTH(contract_date) = 11, supplied_amount, 0)) AS `Nov`,
SUM(IF(MONTH(contract_date) = 12, supplied_amount, 0)) AS `Dec`
FROM contract, supplied
WHERE contract.contract_number = supplied.contract_number AND YEAR(contract_date) = 2018
GROUP BY supplied_product
ORDER BY supplied_product;
```

SQL XAMPP for Windows - mysql -u root

```
MariaDB [supplied]> SELECT supplied_product, SUM(IF(MONTH(contract_date) = 1, supplied_amount, 0)) AS `Jan`,
-> SUM(IF(MONTH(contract_date) = 2, supplied_amount, 0)) AS `Feb`,
-> SUM(IF(MONTH(contract_date) = 3, supplied_amount, 0)) AS `Mar`,
-> SUM(IF(MONTH(contract_date) = 4, supplied_amount, 0)) AS `Apr`,
-> SUM(IF(MONTH(contract_date) = 5, supplied_amount, 0)) AS `May`,
-> SUM(IF(MONTH(contract_date) = 6, supplied_amount, 0)) AS `Jun`,
-> SUM(IF(MONTH(contract_date) = 7, supplied_amount, 0)) AS `Jul`,
-> SUM(IF(MONTH(contract_date) = 8, supplied_amount, 0)) AS `Aug`,
-> SUM(IF(MONTH(contract_date) = 9, supplied_amount, 0)) AS `Sep`,
-> SUM(IF(MONTH(contract_date) = 10, supplied_amount, 0)) AS `Oct`,
-> SUM(IF(MONTH(contract_date) = 11, supplied_amount, 0)) AS `Nov`,
-> SUM(IF(MONTH(contract_date) = 12, supplied_amount, 0)) AS `Dec`
-> FROM contract, supplied
-> WHERE contract.contract_number = supplied.contract_number AND YEAR(contract_date) = 2018
-> GROUP BY supplied_product
-> ORDER BY supplied_product;
```

supplied_product	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Audio Player	0	0	0	0	0	0	0	0	63	33	0	0
Monitor	0	0	0	0	0	0	0	0	85	0	0	0
Printer	0	0	0	0	0	0	0	0	41	0	0	0
Stereo System	0	0	0	0	0	0	0	0	11	0	0	0
TU	0	0	0	0	0	0	0	0	118	14	0	0
Video Player	0	0	0	0	0	0	0	0	20	17	0	0

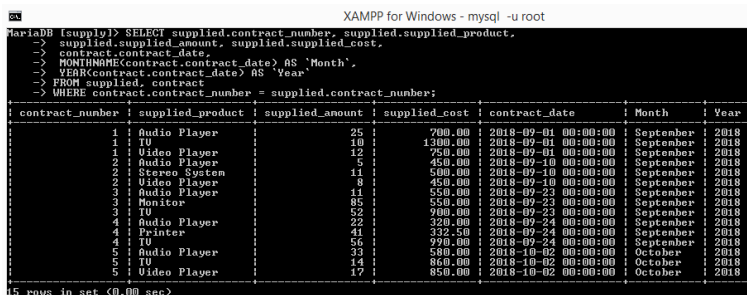
6 rows in set (0.00 sec)

Figure 3.16 – Result of query 16

Query 17

Form a list of supplied goods. For each item in this list, the following information must be shown: contract number, product name, unit number, unit price, delivery date, month name, and year number (figure 3.17).

```
SELECT supplied.contract_number, supplied.supplied_product,  
       supplied.supplied_amount, supplied.supplied_cost,  
       contract.contract_date,  
       MONTHNAME(contract.contract_date) AS `Month`,  
       YEAR(contract.contract_date) AS `Year`  
FROM supplied, contract  
WHERE contract.contract_number = supplied.contract_number;
```



contract_number	supplied_product	supplied_amount	supplied_cost	contract_date	Month	Year
1	Audio Player	25	700.00	2018-09-01 00:00:00	September	2018
1	TV	10	1300.00	2018-09-01 00:00:00	September	2018
1	Video Player	12	750.00	2018-09-01 00:00:00	September	2018
2	Audio Player	5	450.00	2018-09-10 00:00:00	September	2018
2	Stereo System	11	500.00	2018-09-10 00:00:00	September	2018
2	Video Player	8	450.00	2018-09-10 00:00:00	September	2018
3	Audio Player	11	550.00	2018-09-23 00:00:00	September	2018
3	Monitor	85	550.00	2018-09-23 00:00:00	September	2018
3	TV	52	900.00	2018-09-23 00:00:00	September	2018
4	Audio Player	22	320.00	2018-09-24 00:00:00	September	2018
4	Printer	41	332.50	2018-09-24 00:00:00	September	2018
4	TV	56	990.00	2018-09-24 00:00:00	September	2018
5	Audio Player	33	580.00	2018-10-02 00:00:00	October	2018
5	TV	14	860.00	2018-10-02 00:00:00	October	2018
5	Video Player	17	850.00	2018-10-02 00:00:00	October	2018

Figure 3.17 – Result of query 17

2. Make a report for the laboratory work

The report should include the main stages of laboratory work and screenshots that demonstrate them.

3. Questions

1. What SQL statement is used to retrieve data from one or several tables?
2. Show the common structure of the SELECT statement.
3. Which form of the SQL SELECT statement might be used if it is required to display all columns of a certain table?
4. Which construction is used to select records that satisfy search criteria?
5. What keyword is used to exclude duplicate rows?
6. Which construction is used to sort values by single or multiple columns?
7. How the reverse sorting might be implemented?
8. Which keyword is used to limit the range of retrieved records?
9. Which construction is used to group retrieved records?

10. Name the aggregation functions, their purpose, and basic features.
11. How to give the new name to a specific column?
12. What is the purpose of the HAVING keyword? What is the difference between this keyword from WHERE?
13. Name basic arithmetic, logic, and comparison operators, their purpose, and examples of usage.
14. The purpose of the MONTH function and examples of its usage.
15. The purpose of the YEAR function and examples of its usage.
16. The purpose of the IFNULL function and examples of its usage.
17. The purpose of the CONCAT function and examples of its usage.
18. The purpose of the RTRIM function and examples of its usage.
19. The purpose of the SUBSTRING function and examples of its usage.
20. The purpose of the IF function and examples of its usage.
21. Which operator is used to combine the results of two queries?