EXERCISE: 2 22P/22P

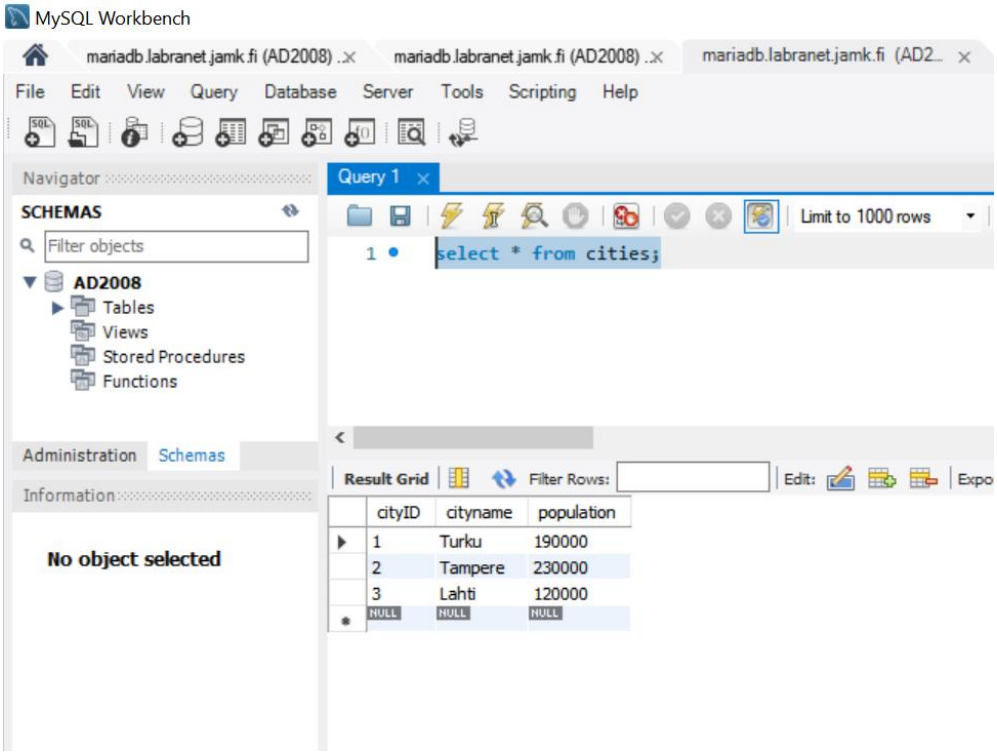
Task 1 [5p/5p score]

TASK1 A

A. Search all cities (cities); show all columns

- The SELECT statement is used to select data from a database.
- * is used to select all the coiumns in the tables
- The following SQL statement selects all the records in the "cities" table: and the results are in the screen shots.

```
select * from cities;
```

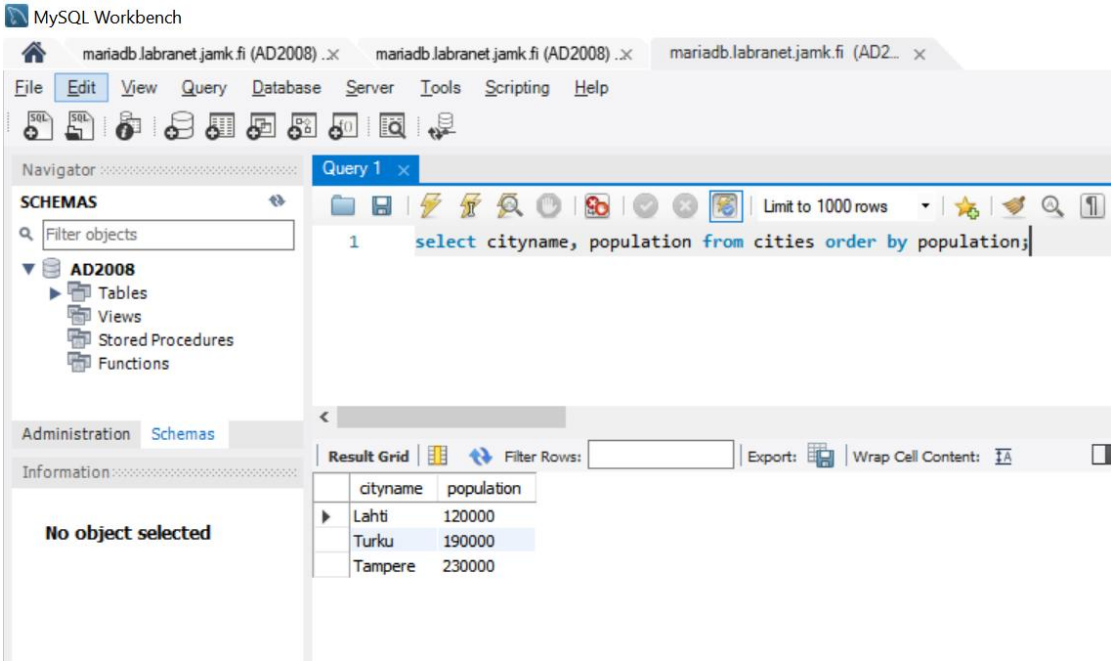


TASK1 B

B. Search for all city names and population numbers. Sort by population from smallest to largest.

- Here column1, column2,(cityname, population) ... are the field names of the table(cities) we want to select data from.
- The ORDER BY keyword is used to sort the result-set in ascending or descending order.
- The ORDER BY keyword sorts the records in ascending order by default
- The following SQL statement selects the fields(cityname, population) from the "cities" table: and sort them by population, the results are in the screen shots.

```
select cityname, population from cities order by population;
```

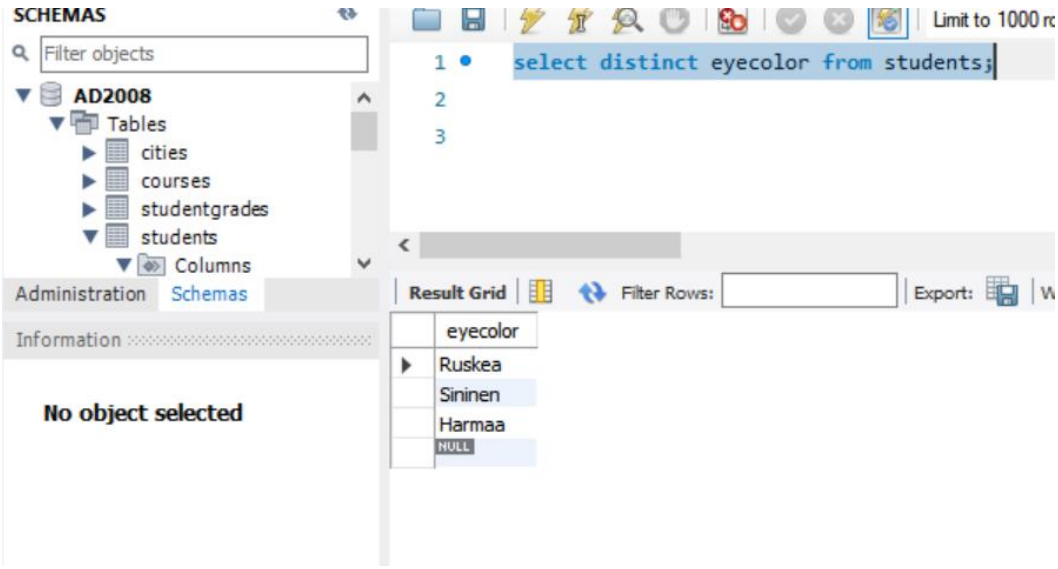


TASK1 C

C. What different eye colors do students have? Each color can be printed only once and the result set can only have one column.

- The SELECT DISTINCT statement is used to return only distinct (different) values. Inside a table(students), a column(eyecolor) often contains many duplicate values; and sometimes required only want to list the different (distinct) values.
- The following statement shows the distinct eyecolor value data,the results are shown in the screenshot

```
select distinct eyecolor from students;
```

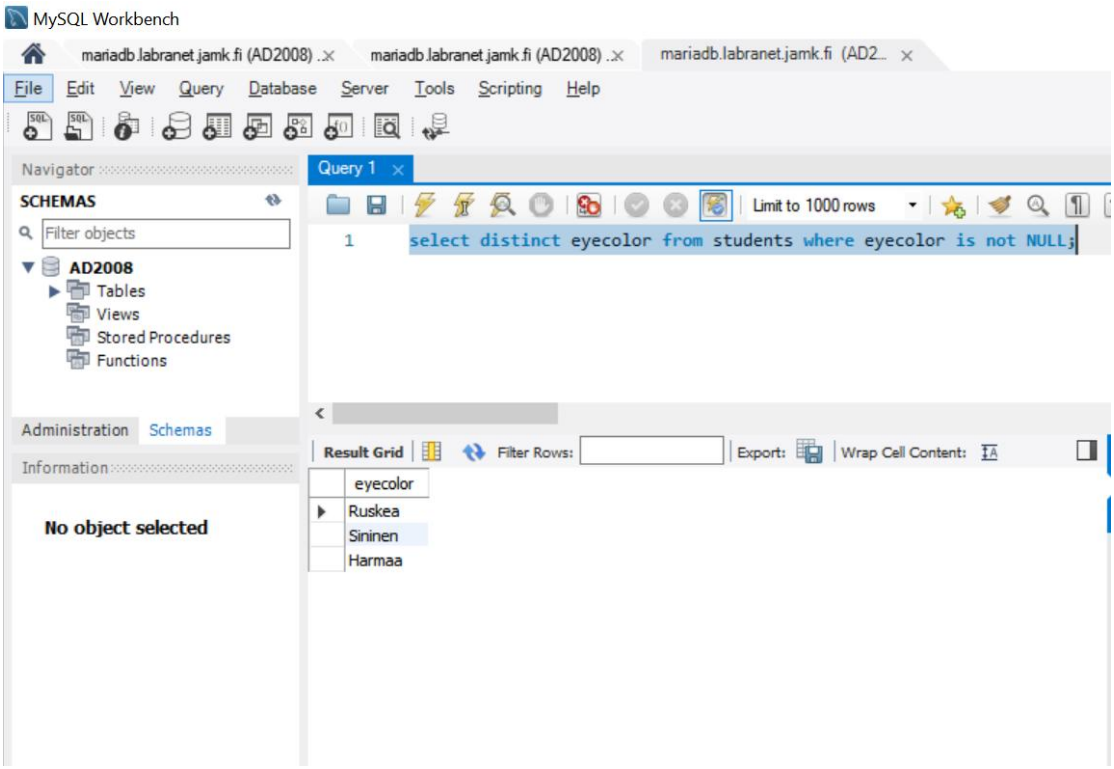


- A NULL value is different from a zero value or a field that contains spaces. A field with a NULL value is one that has been left blank during record creation!

- The IS NOT NULL operator is used to test for non-empty values (NOT NULL values).

- The following SQL lists all students with a value in the "eyecolor" field, and used above keyword to retrieve only mentioned values except Null, the results are in the screen shots.

```
select distinct eyecolor from students where eyecolor is not NULL;
```



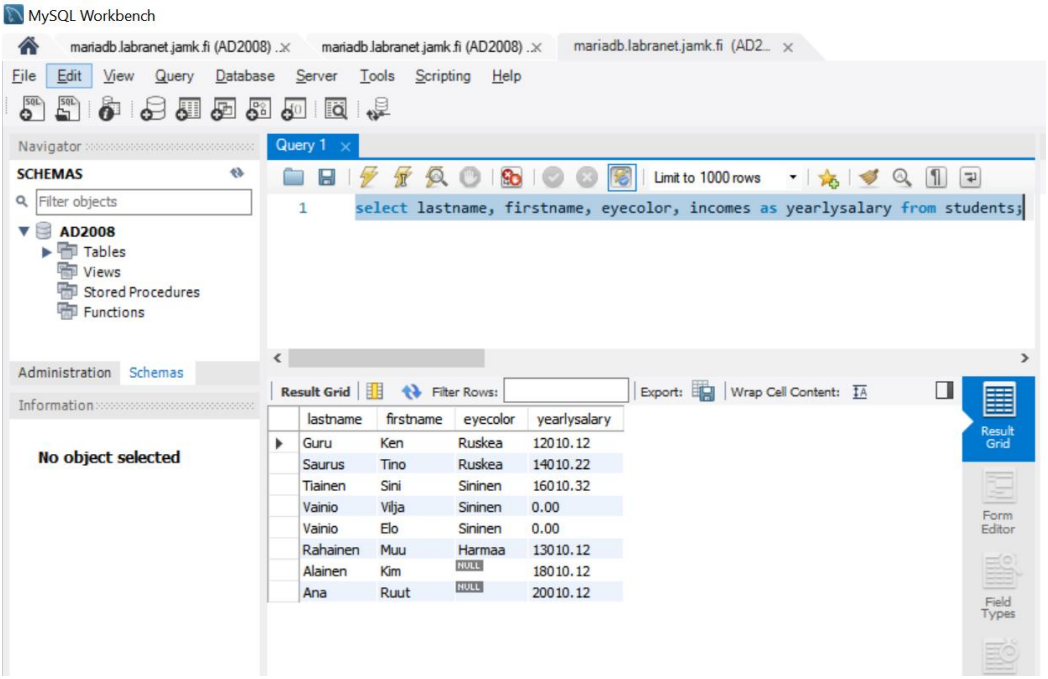
TASK1 D

D. Search for students' last and first names, eye color and incomes: the heading of the incomes column must be Yearly Salary.

- SQL aliases are used to give a table, or a column in a table, a temporary name. Aliases are often used to make column names more readable. An alias only exists for the duration of that query. An alias is created with the AS keyword.The following SQL statements

- The following SQL statement selects the fields(lastname firstname eyecolor and income as yearlysalary) from the "students" table: and sort them by population, the results are in the screen shots.(AS keyword changes the name temporarly)

```
select lastname, firstname, eyecolor, incomes as yearlysalary from
students;
```

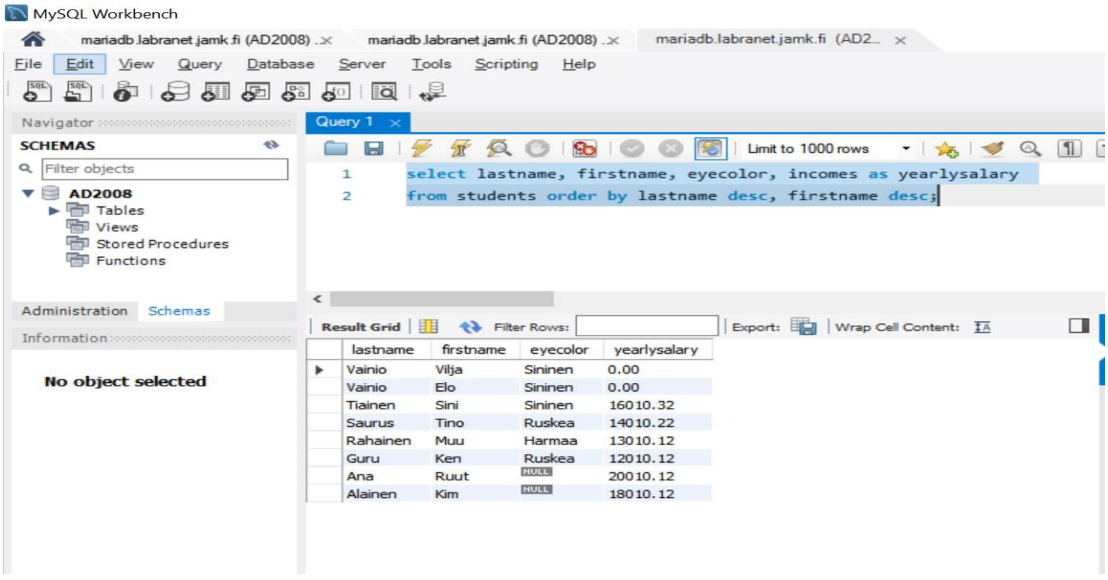


TASK1 E

E. Sort the previous task primarily by last name and secondarily by first name, both in descending order (reverse alphabetical order).

- The previous task statement is changed by sorting the firstname lastname in reverse alphabetical order
- The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.
- The following statement sort both columns in reverse alphabetical order using above mentioned keywords and the results are in the screenshoot.

```
select lastname, firstname, eyecolor, incomes as yearlysalary
from students order by lastname desc, firstname desc;
```



Task 2 [5p/5p score]

TASK 2 A

A. Search for the last and first names of students whose eye color is 'Blue'.

- The WHERE clause is used to filter records. It is used to extract only those records that fulfill a specified condition.SQL requires single quotes around text values (most database systems will also allow double quotes). However, numeric fields should not be enclosed in quotes:

- The following SQL statement selects the fields(lastname, firstname) from the eyecolor "sininen", in the "students" table, the results are in the screenshot.

```
select lastname, firstname from students where eyecolor = 'sininen';
```

```
MariaDB [AD2008]> select lastname, firstname from students where eyecolor = 'sininen';
+----------+-----------+
| lastname | firstname |
+----------+-----------+
| Tiainen  | Sini      |
| Vainio   | Vilja     |
| Vainio   | Elo       |
+----------+-----------+
3 rows in set (1.116 sec)
```

TASK 2 B

B. List those students with incomes less than 14010.22? Search for names and income.

- The comparision operators (<, >, <>, <=, >=, !=, !>, !<) can be used in the WHERE clause:
- The following statement where clause check the condition (< 14010.22) an extracts only those records, with the colums (lastname, firstnames and incomes) the results are shown in the screenshot.

```
select lastname, firstname, incomes from students where incomes <
14010.22;
```

```
MariaDB [AD2008]> SELECT lastname, firstname, incomes FROM students WHERE incomes < 14010.22;
+----------+-----------+----------+
| lastname | firstname | incomes  |
+----------+-----------+----------+
| Guru     | Ken       | 12010.12 |
| Vainio   | Vilja     |     0.00 |
| Vainio   | Elo       |     0.00 |
| Rahainen | Muu       | 13010.12 |
+----------+-----------+----------+
4 rows in set (0.041 sec)
```

TASK 2 C

C. List those students with incomes less than or equal to 14010.22? Search for names and income. Sort by salary in descending order.

- In the following statement comparision operator where clause is checked and sorted the incomes in descending order, the results are in the screen shot.

```
select lastname, firstname, incomes from students where incomes <=
14010.22 order by incomes desc;
```

```
MariaDB [AD2008]> SELECT lastname, firstname, incomes FROM students WHERE incomes <= 14010.22 ORDER BY incomes DESC;
+----------+-----------+----------+
| lastname | firstname | incomes  |
+----------+-----------+----------+
| Saurus   | Tino      | 14010.22 |
| Rahainen | Muu       | 13010.12 |
| Guru     | Ken       | 12010.12 |
| Vainio   | Vilja     |     0.00 |
| Vainio   | Elo       |     0.00 |
+----------+-----------+----------+
5 rows in set (0.054 sec)
```

TASK 2 D

D. Find the students from Turku (hometown = 1) whose eye color is 'Blue'. Print columns studentID, lastname, firstname, eyecolorand hometown(integer).

- The WHERE clause can be combined with AND, OR, and NOT operators.
  The AND and OR operators are used to filter records based on more than one condition:
  The AND operator displays a record if all the conditions separated by AND are TRUE.

- In the following statement I used AND to check the two where clause conditions(eyecolor and hometown), the results are in the screenshot with the mentioned columns.

```
select studentID, lastname, firstname, eyecolor, hometown from
students where eyecolor = 'sininen'and hometown = 1;
```

```
MariaDB [AD2008]> select studentID, lastname, firstname, eyecolor, hometown
    -> from students where eyecolor = 'sininen'and hometown = 1;
+-----------+----------+-----------+----------+----------+
| studentID | lastname | firstname | eyecolor | hometown |
+-----------+----------+-----------+----------+----------+
|      2003 | Tiainen  | Sini      | Sininen  |        1 |
+-----------+----------+-----------+----------+----------+
1 row in set (0.037 sec)
```

TASK 2 E

E.List all the students from Turku and, in addition to them , all those with gray eyes. Print all columns.

- The OR operator displays a record if any of the conditions separated by OR is TRUE.
- The following statement print all the columns and check the condition (students from turku and also students with harmaa eyecolor) with OR operator in the where clause,the results are in the screenshot.

```
select * from students where hometown = 1 or eyecolor = 'harmaa';
```

```
MariaDB [AD2008]> select * from students where hometown = 1 or eyecolor = 'harmaa';
+-----------+----------+-----------+------------+----------+----------+---------+----------+
| studentID | lastname | firstname | birthdate  | eyecolor | incomes  | taxrate | hometown |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
|      2001 | Guru     | Ken       | 2001-11-11 | Ruskea   | 12010.12 |    5.10 |        1 |
|      2002 | Saurus   | Tino      | 2002-11-11 | Ruskea   | 14010.22 |    6.20 |        1 |
|      2003 | Tiainen  | Sini      | 2003-11-11 | Sininen  | 16010.32 |    7.30 |        1 |
|      2006 | Rahainen | Muu       | 2006-11-11 | Harmaa   | 13010.12 |    5.80 |        2 |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
4 rows in set (0.046 sec)
```

Task 3 [5p/5p]

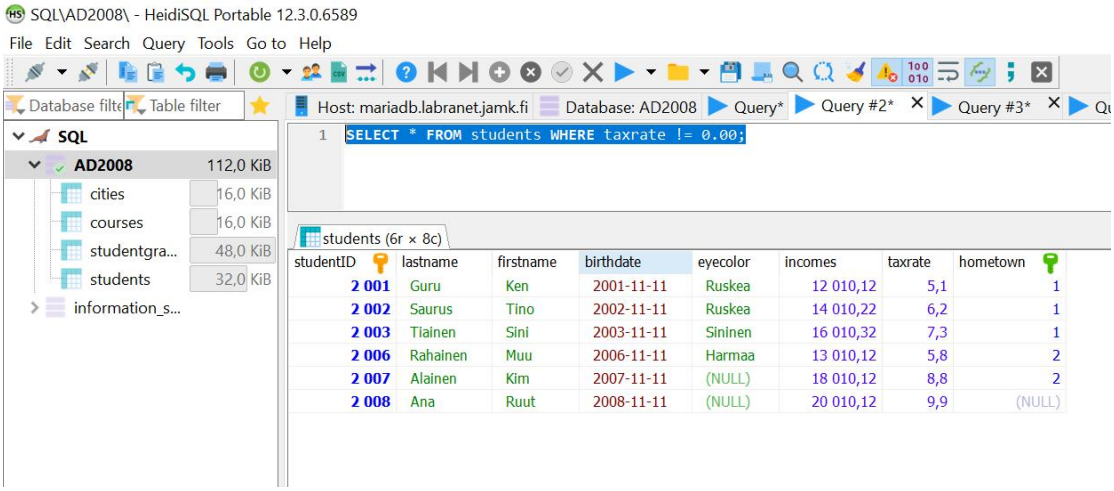
TASK 3 A

A. Search all columns for students whose tax rate is not 0.00

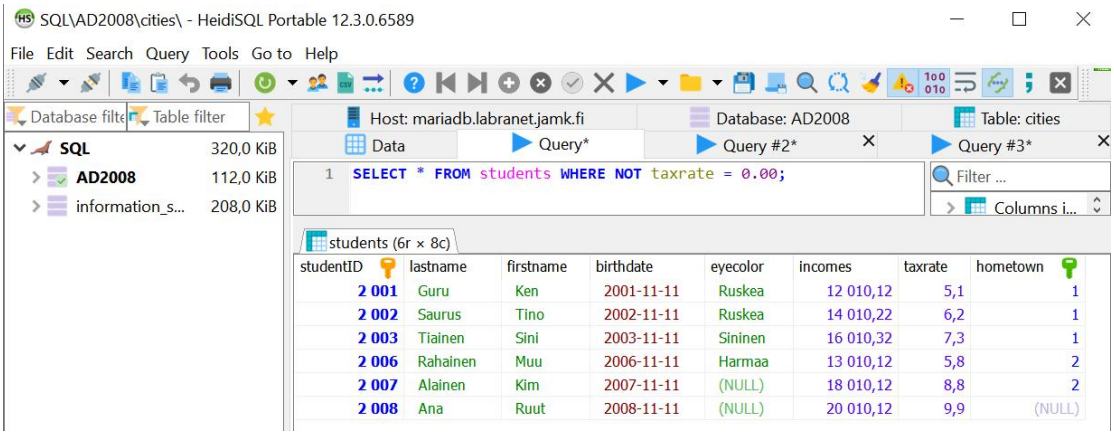
Is not operator may be written as !=

- The NOT operator displays a record if the condition(s) is NOT TRUE.
- In the following statement I used != comparision operator and NOT operator to check the condition is not ,both the results are in the screenshot.

```
SELECT * FROM students WHERE taxrate != 0.00;
```




```
SELECT * FROM students WHERE NOT taxrate = 0.00;
```



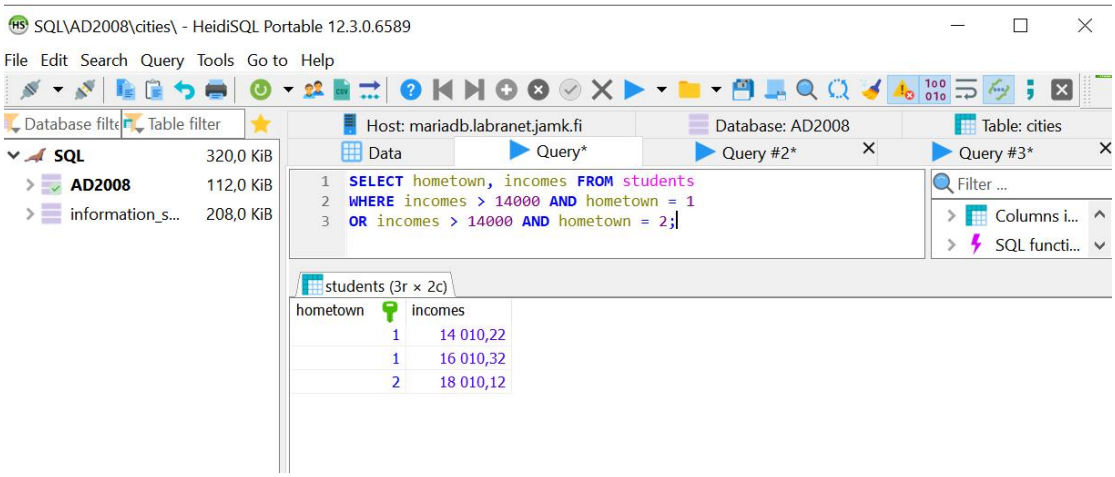
TASK 3 B

B. Search for students from Turku or Tampere with an income of more than 14000.

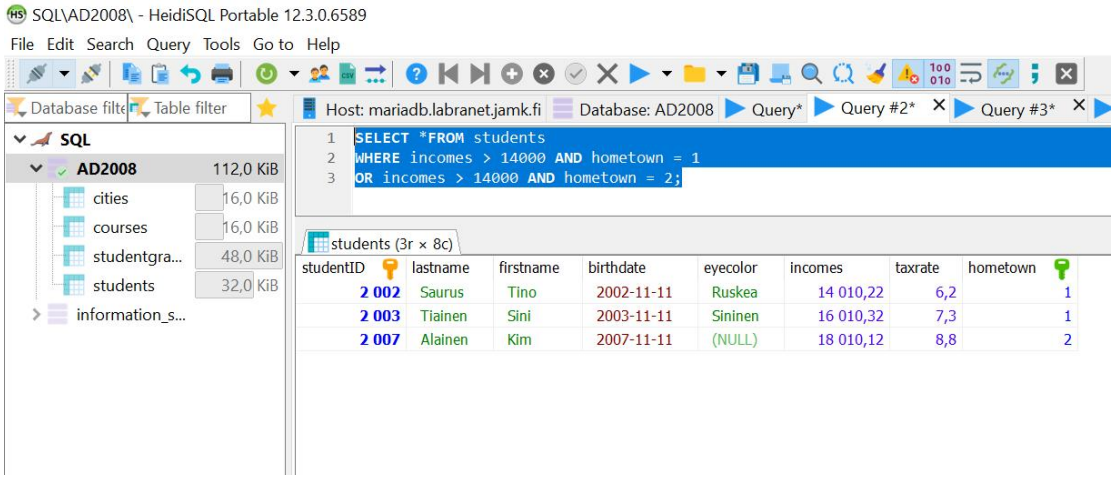
- The WHERE clause can be combined with AND, OR, and NOT operators.
  The AND and OR operators are used to filter records based on more than one condition:
  The AND operator displays a record if all the conditions separated by AND are TRUE.

- In the following statement I used AND, OR to check the two where clause conditions(incomes and hometown), the results are in the screenshot.

```
SELECT hometown, incomes FROM students
WHERE incomes > 14000 AND hometown = 1
OR incomes > 14000 AND hometown = 2;
```




```sql
SELECT *FROM students
WHERE incomes > 14000 AND hometown = 1
OR incomes > 14000 AND hometown = 2;
```

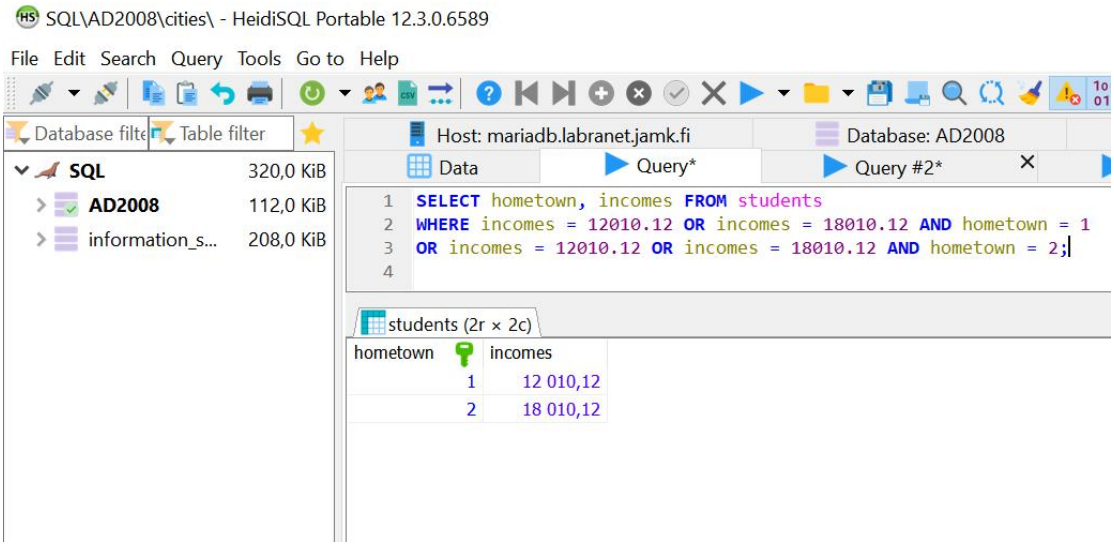


TASK 3 C

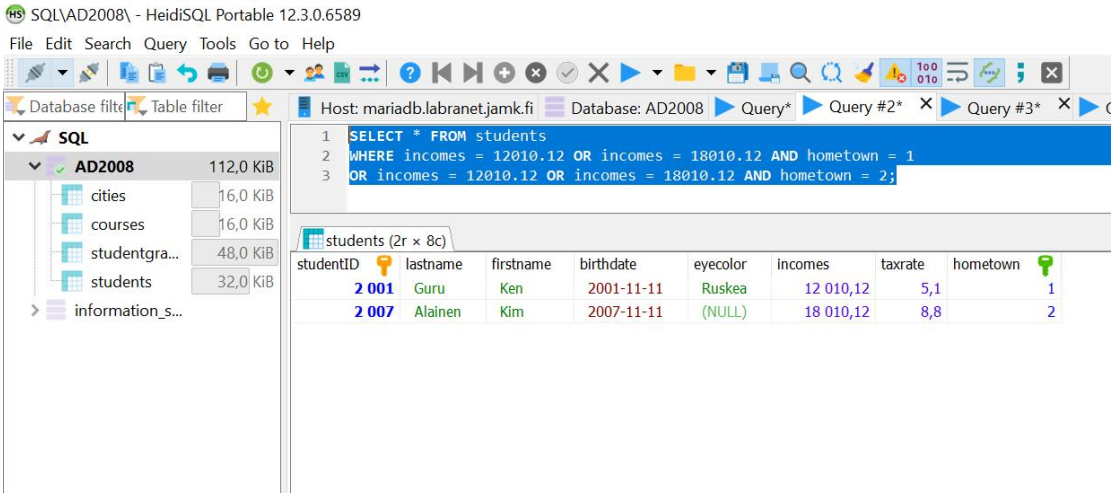
C. Search for students from Turku or Tampere with an income of either 12010.12 or 18010.12

- The following statement is used to execute the students with incomes 12010.12 or 18010.12 from hometown turku(1) or tampere(2) using where clause AND OR operators, the results are in the screenshot.

```
SELECT hometown, incomes FROM students
WHERE incomes = 12010.12 OR incomes = 18010.12 AND hometown = 1
OR incomes = 12010.12 OR incomes = 18010.12 AND hometown = 2;
```




```
SELECT * FROM students
WHERE incomes = 12010.12 OR incomes = 18010.12 AND hometown = 1
OR incomes = 12010.12 OR incomes = 18010.12 AND hometown = 2;
```

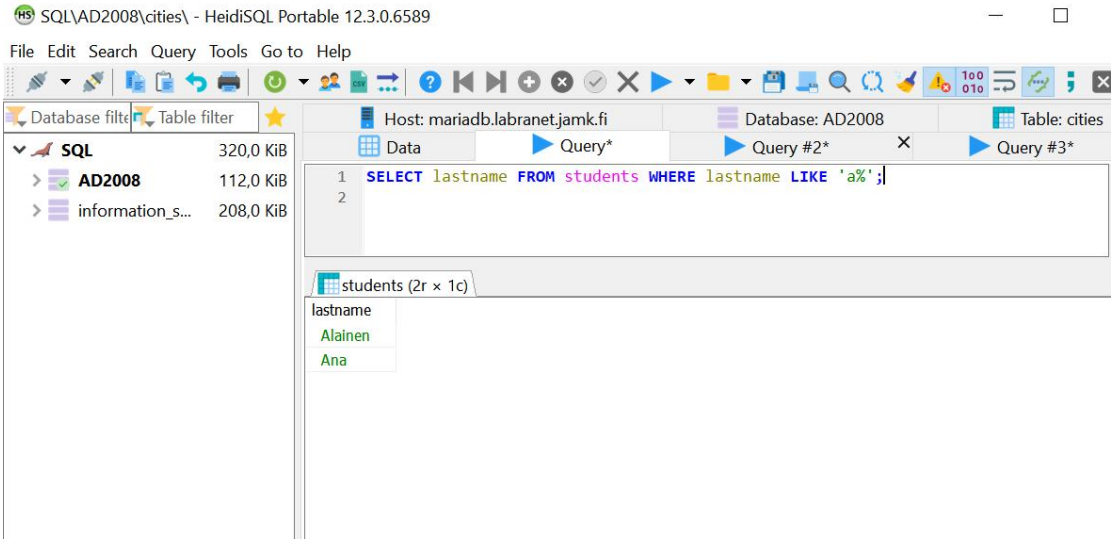


TASK 3 D

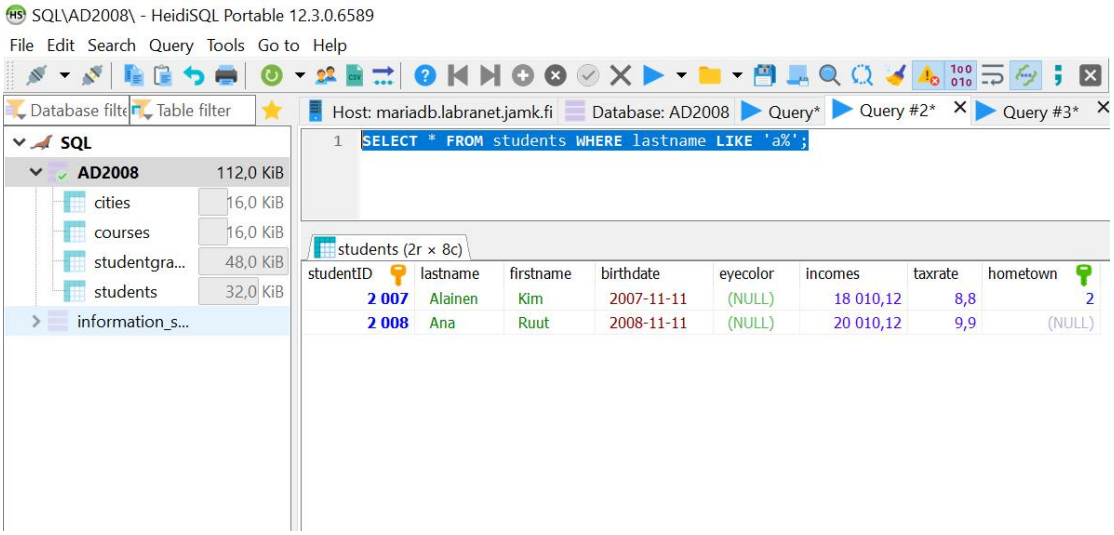
D. Search for students , whose last name starts with the letter A.

- The LIKE operator is used in a WHERE clause to search for a specified pattern in a column. There are two wildcards often used in conjunction with the LIKE operator: The percent sign (%) represents zero, one, or multiple characters. The underscore sign (_) represents one, single character
- WHERE lastname LIKE ‘a%’ ------Finds any values that start with “a”
- In the following statement all students with lastname starts with “a” the results are in the screenshot.

```
SELECT lastname FROM students WHERE lastname LIKE 'a%';
```




```
SELECT * FROM students WHERE lastname LIKE 'a%';
```

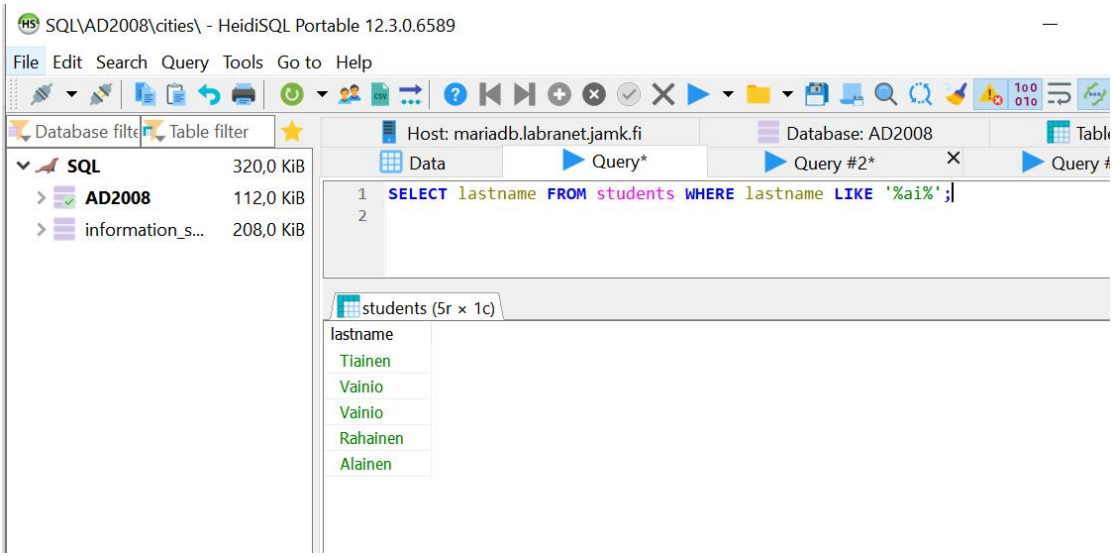


TASK 3 E

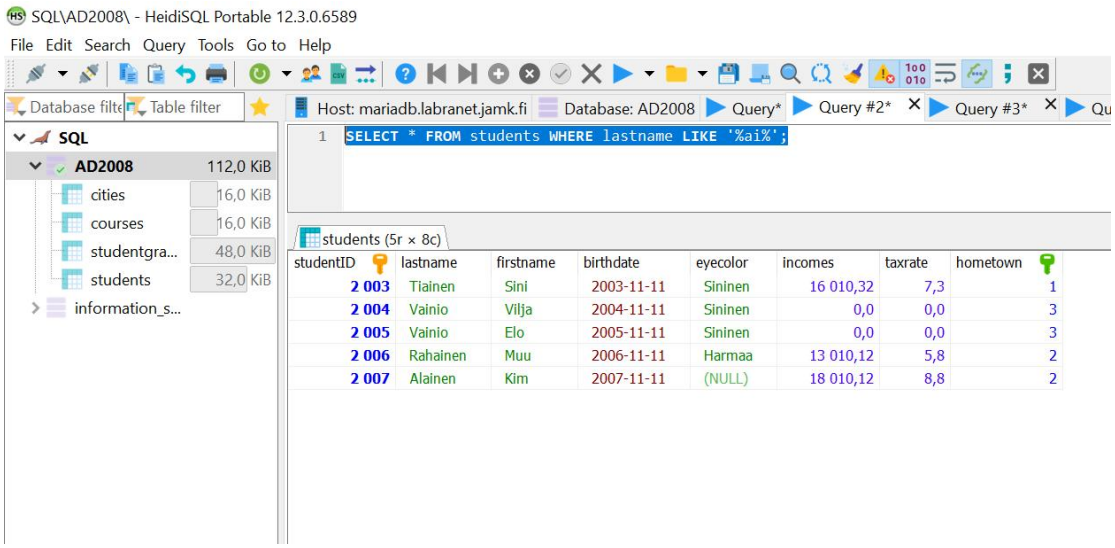
E. Search for students whose last name contains the string 'ai'

- Where lastname LIKE ‘%ai%’--------finds any values that have “ai” in any position
- In the following statement all students with lastname contains string with “ai” are executed , the results are in the screenshot

```
SELECT lastname FROM students WHERE lastname LIKE '%ai%';
```




```
SELECT * FROM students WHERE lastname LIKE '%ai%';
```



Task 4 [5p/5p]

TASK 4 A

A. Search for students whose last name is not a.

- Where lastname NOT LIKE '%a%'--------finds any values that does have "a" in any position
- In the following statement all students with lastname contains string without "a" are executed , the results are in the screenshot

```
SELECT lastname FROM students WHERE lastname NOT LIKE '%a%';
```

```
MariaDB [AD2008]> SELECT lastname FROM students WHERE lastname NOT LIKE '%a%';
+----------+
| lastname |
+----------+
| Guru     |
+----------+
1 row in set (0.038 sec)
```

```
SELECT * FROM students WHERE lastname NOT LIKE '%a%';
```

```
MariaDB [AD2008]> SELECT * FROM students WHERE lastname NOT LIKE '%a%';
+-----------+----------+-----------+------------+----------+----------+---------+----------+
| studentID | lastname | firstname | birthdate  | eyecolor | incomes  | taxrate | hometown |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
|      2001 | Guru     | Ken       | 2001-11-11 | Ruskea   | 12010.12 |    5.10 |        1 |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
1 row in set (0.029 sec)
```

TASK 4 B

B. Search for students whose tax rate is between 0.00 and 5.10 Sort by tax rate. (Use the BETWEEN attribute)

- The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates.The BETWEEN operator is inclusive: begin and end values are included.
- In the following statement I used between operator to select taxrate range between 0.00 and 5.10 and sort by ascending order which is default, the results are in the screeen short

```
SELECT taxrate from students where taxrate between 0.00 and 5.10 order by taxrate;
```

```
MariaDB [AD2008]> select taxrate from students where taxrate between 0.00 and 5.10 order by taxrate;
+---------+
| taxrate |
+---------+
|    0.00 |
|    0.00 |
|    5.10 |
+---------+
3 rows in set (0.050 sec)
```

```
SELECT * from students where taxrate between 0.00 and 5.10 order by taxrate;
```

```
MariaDB [AD2008]> SELECT * from students where taxrate between 0.00 and 5.10 order by taxrate;
+-----------+----------+-----------+------------+----------+----------+---------+----------+
| studentID | lastname | firstname | birthdate  | eyecolor | incomes  | taxrate | hometown |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
|      2004 | Vainio   | Vilja     | 2004-11-11 | Sininen  |     0.00 |    0.00 |        3 |
|      2005 | Vainio   | Elo       | 2005-11-11 | Sininen  |     0.00 |    0.00 |        3 |
|      2001 | Guru     | Ken       | 2001-11-11 | Ruskea   | 12010.12 |    5.10 |        1 |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
3 rows in set (0.037 sec)
```

TASK 4 C

C. Search for students whose tax rate is 0.00, 6.20 or 7.30 (use the IN attribute)

- The IN operator allows you to specify multiple values in a WHERE clause. The IN operator is a shorthand for multiple OR conditions.
- The following SQL statement I used IN operator to selects all taxrate that are with values 0.00,6,20 0r 7.30,the results in the screenshot.

SELECT * from students where taxrate IN (0.00, 6.20, 7.30);

```
MariaDB [AD2008]> SELECT * from students where taxrate IN (0.00, 6.20, 7.30);
+-----------+----------+-----------+------------+----------+----------+---------+----------+
| studentID | lastname | firstname | birthdate  | eyecolor | incomes  | taxrate | hometown |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
|      2002 | Saurus   | Tino      | 2002-11-11 | Ruskea   | 14010.22 |    6.20 |        1 |
|      2003 | Tiainen  | Sini      | 2003-11-11 | Sininen  | 16010.32 |    7.30 |        1 |
|      2004 | Vainio   | Vilja     | 2004-11-11 | Sininen  |     0.00 |    0.00 |        3 |
|      2005 | Vainio   | Elo       | 2005-11-11 | Sininen  |     0.00 |    0.00 |        3 |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
4 rows in set (0.032 sec)
```

TASK 4 D

D. Search for students whose home town is not known (is NULL)

- The IS NULL operator is used to test for empty values (NULL values).
- In the following statement I used IS NULL operator and the results are in the screenshot.

SELECT * FROM students WHERE hometown IS NULL;

```
MariaDB [AD2008]> SELECT * FROM students WHERE hometown IS NULL;
+-----------+----------+-----------+------------+----------+----------+---------+----------+
| studentID | lastname | firstname | birthdate  | eyecolor | incomes  | taxrate | hometown |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
|      2008 | Ana      | Ruut      | 2008-11-11 | NULL     | 20010.12 |    9.90 |     NULL |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
1 row in set (0.038 sec)
```

TASK 4 E

E. Search for students whose eye color is NOT known to be 'Blue'.

- The following statements shows the students without sininen eyecolor , the results are in the screenshot

```
SELECT * FROM students WHERE eyecolor != 'sininen' OR eyecolor IS
NULL;
```

```
MariaDB [AD2008]> SELECT * FROM students WHERE eyecolor != 'sininen' OR eyecolor IS NULL;
+-----------+----------+-----------+------------+----------+----------+---------+----------+
| studentID | lastname | firstname | birthdate  | eyecolor | incomes  | taxrate | hometown |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
|      2001 | Guru     | Ken       | 2001-11-11 | Ruskea   | 12010.12 |    5.10 |        1 |
|      2002 | Saurus   | Tino      | 2002-11-11 | Ruskea   | 14010.22 |    6.20 |        1 |
|      2006 | Rahainen | Muu       | 2006-11-11 | Harmaa   | 13010.12 |    5.80 |        2 |
|      2007 | Alainen  | Kim       | 2007-11-11 | NULL     | 18010.12 |    8.80 |        2 |
|      2008 | Ana      | Ruut      | 2008-11-11 | NULL     | 20010.12 |    9.90 |     NULL |
+-----------+----------+-----------+------------+----------+----------+---------+----------+
5 rows in set (0.041 sec)
```

Task 5 [2p/2p]

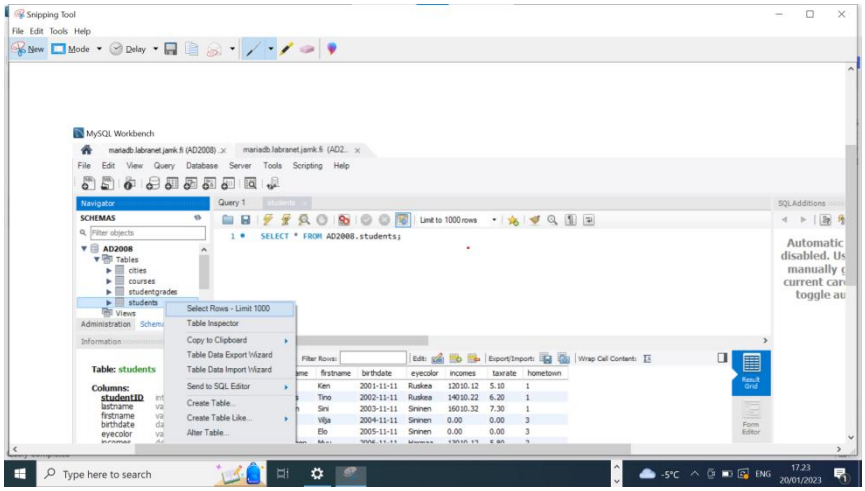
MY experiences of the mysql, MySQL Workbench and HeidiSQL programs are

Myql:

- Initially I observed login issue to labranet Maria DB through mysql command line tool.
- As it is a (CLI)command line interface I have to write commands manually and has to remember the commands.
- In CLI we have to write command to view the tables in the databases
- Cannot save tab query

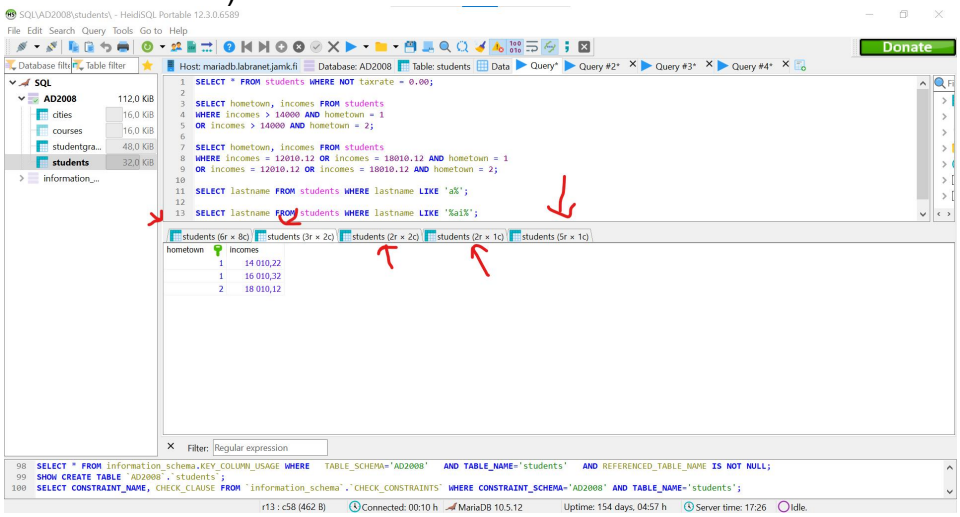
MY SQL Workbench

- It is GUI(Graphical user interface) No need to remember all commands it is comfortable to run SQL queires to work with stored data.
- I can browse database schemas, work with database objects.
- In GUI I can see what are the tables available in database directly.
- I can select the tables data by right click on the tables and then select rows-limit 1000



HeidiSQL

- It is somewhat same like MYSQL workbench but I observed that we can run many queires together and check the results of specific queiry by selecting the requied one at the result window . (the arrows in the screenshot)



- Can save the content of tab query in Heidi portable

CONDITIONS TO RUN THESE PROGRAMS:

- Login credentials to labranetmariadb is required everytime to run the programs in command line tool.
- VPN connection is also mandatory to access and run the programs in Mysqlcommand line tool, MYSQL workbench and Heidisql.

SQL QUIERIES USED IN THE ASSIGNMENTS.

TASK1 A

```
select * from cities;
```

TASK1 B

```
select cityname, population from cities order by population;
```

TASK1 C

```
select distinct eyecolor from students;

select distinct eyecolor from students where eyecolor is not NULL;
```

TASK1 D

```
select lastname, firstname, eyecolor, incomes as yearlysalary from
students;
```

TASK1 E

```
select lastname, firstname, eyecolor, incomes as yearlysalary
from students order by lastname desc, firstname desc;
```

TASK 2 A

```
select lastname, firstname from students where eyecolor = 'sininen';
```

TASK 2 B

```
select lastname, firstname, incomes from students where incomes <
14010.22;
```

TASK 2 C

```
select lastname, firstname, incomes from students where incomes <=
14010.22 order by incomes desc;
```

TASK 2 D

```
select studentID, lastname, firstname, eyecolor, hometown from
students where eyecolor = 'sininen'and hometown = 1;
```

TASK 2 E

```
select * from students where hometown = 1 or eyecolor = 'harmaa';
```

TASK 3 A

```
SELECT * FROM students WHERE taxrate != 0.00;
```

```
SELECT * FROM students WHERE NOT taxrate = 0.00;
```

TASK 3 B

```
SELECT hometown, incomes FROM students
WHERE incomes > 14000 AND hometown = 1
OR incomes > 14000 AND hometown = 2;
```

```
SELECT *FROM students
WHERE incomes > 14000 AND hometown = 1
OR incomes > 14000 AND hometown = 2;
```

TASK 3 C

```
SELECT hometown, incomes FROM students
WHERE incomes = 12010.12 OR incomes = 18010.12 AND hometown = 1
OR incomes = 12010.12 OR incomes = 18010.12 AND hometown = 2;
```

```
SELECT * FROM students
WHERE incomes = 12010.12 OR incomes = 18010.12 AND hometown = 1
OR incomes = 12010.12 OR incomes = 18010.12 AND hometown = 2;
```

TASK 3 D

```
SELECT lastname FROM students WHERE lastname LIKE 'a%';
```

```
SELECT * FROM students WHERE lastname LIKE 'a%';
```

TASK 3 E

```
SELECT lastname FROM students WHERE lastname LIKE '%ai%';
```

```
SELECT * FROM students WHERE lastname LIKE '%ai%';
```

TASK 4 A

SELECT lastname **FROM** students **WHERE** lastname **NOT LIKE** '%a%';

```
SELECT * FROM students WHERE lastname NOT LIKE '%a%';
```

TASK 4 B

SELECT taxrate from students where taxrate between 0.00 and 5.10 order by taxrate;

```
SELECT * from students where taxrate between 0.00 and 5.10 order by
taxrate;
```

TASK 4 C

SELECT * from students where taxrate IN (0.00, 6.20, 7.30);

TASK 4 D

SELECT * FROM students **WHERE** hometown **IS NULL**;

TASK 4 E

```
SELECT * FROM students WHERE eyecolor != 'sininen' OR eyecolor IS
NULL;
```