**MODULE: 5 (Database)**

 What do you understand By Database

Ans: A database is **an organized collection of structured information, or data, typically stored electronically in a computer system**. A database is usually controlled by a database management system (DBMS).

 What is Normalization?

Ans: **Normalization** is a database design technique that reduces data redundancy and eliminates undesirable characteristics like Insertion, Update and Deletion Anomalies. Normalization rules divides larger tables into smaller tables and links them using relationships. The purpose of Normalisation in SQL is to eliminate redundant (repetitive) data and ensure data is stored logically.

 What is Difference between DBMS and RDBMS?

### Ans: Difference between RDBMS and DBMS

|  |  |
| --- | --- |
| RDBMS | DBMS |
| Data stored is in table format | Data stored is in the file format |
| Multiple data elements are accessible together | Individual access of data elements |
| Data in the form of a table are linked together | No connection between data |
| Normalisation is not achievable | There is normalisation |
| Support distributed database | No support for distributed database |
| Data is stored in a large amount | Data stored is a small quantity |
| Here, redundancy of data is reduced with the help of key and indexes in RDBMS | Data redundancy is common |
| RDBMS supports multiple users | DBMS supports a single user |
| It features multiple layers of security while handling data | There is only low security while handling data |
| The software and hardware requirements are higher | The software and hardware requirements are low |
| Oracle, SQL Server. | XML, Microsoft Access. |

 What is MF Cod Rule of RDBMS Systems?

Ans: **Codd's twelve rules**[[1]](https://en.wikipedia.org/wiki/Codd%27s_12_rules#cite_note-coddsrules-1) are a set of thirteen rules ([numbered zero to twelve](https://en.wikipedia.org/wiki/Zero-based_numbering)) proposed by [Edgar F. Codd](https://en.wikipedia.org/wiki/Edgar_F._Codd), a pioneer of the [relational model](https://en.wikipedia.org/wiki/Relational_model) for [databases](https://en.wikipedia.org/wiki/Database), designed to define what is required from a [database management system](https://en.wikipedia.org/wiki/Database_management_system) in order for it to be considered *relational*, i.e., a [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS).[[2]](https://en.wikipedia.org/wiki/Codd%27s_12_rules#cite_note-2)[[3]](https://en.wikipedia.org/wiki/Codd%27s_12_rules#cite_note-3) They are sometimes referred to as "Codd's Twelve Commandments".

**Rule 0:** The *foundation rule*:

For any system that is advertised as, or claimed to be, a relational data base management system, that system must be able to manage data bases entirely through its relational capabilities.

**Rule 1:** The *information rule*:

All information in a relational data base is represented explicitly at the logical level and in exactly one way – by values in tables.

**Rule 2:** The *guaranteed access rule*:

Each and every datum (atomic value) in a relational data base is guaranteed to be logically accessible by resorting to a combination of table name, primary key value and column name.

**Rule 3:** *Systematic treatment of null values*:

Null values (distinct from the empty character string or a string of blank characters and distinct from zero or any other number) are supported in fully relational DBMS for representing missing information and inapplicable information in a systematic way, independent of data type.

**Rule 4:** *Dynamic* [*online*](https://en.wikipedia.org/wiki/Online)[*catalog*](https://en.wikipedia.org/wiki/Database_catalog) *based on the relational model*:

The data base description is represented at the logical level in the same way as ordinary data, so that authorized users can apply the same relational language to its interrogation as they apply to the regular data.

**Rule 5:** The *comprehensive data sublanguage rule*:

A relational system may support several languages and various modes of terminal use (for example, the fill-in-the-blanks mode). However, there must be at least one language whose statements are expressible, per some well-defined syntax, as character strings and that is comprehensive in supporting all of the following items:

1. Data definition.
2. View definition.
3. Data manipulation (interactive and by program).
4. Integrity constraints.
5. Authorization.
6. Transaction boundaries (begin, commit and rollback).

**Rule 6:** The [*view*](https://en.wikipedia.org/wiki/View_(SQL)) *updating rule*:

All views that are theoretically updatable are also updatable by the system.

**Rule 7:** Relational Operations Rule / *Possible for high-level insert, update, and delete*:

The capability of handling a base relation or a derived relation as a single operand applies not only to the retrieval of data but also to the insertion, update and deletion of data.

**Rule 8:** *Physical data independence*:

Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representations or access methods.

**Rule 9:** *Logical data independence*:

Application programs and terminal activities remain logically unimpaired when information-preserving changes of any kind that theoretically permit unimpairment are made to the base tables.

**Rule 10:** *Integrity independence*:

Integrity constraints specific to a particular relational data base must be definable in the relational data sublanguage and storable in the catalog, not in the application programs.

**Rule 11:** *Distribution independence*:

The end-user must not be able to see that the data is distributed over various locations. Users should always get the impression that the data is located at one site only.

**Rule 12:** The *nonsubversion rule*:

If a relational system has a low-level (single-record-at-a-time) language, that low level cannot be used to subvert or bypass the integrity rules and constraints expressed in the higher level relational language (multiple-records-at-a-time).

 What do you understand By Data Redundancy?

Ans: Data redundancy **occurs when the same piece of data exists in multiple places**, whereas data inconsistency is when the same data exists in different formats in multiple tables. Unfortunately, data redundancy can cause data inconsistency, which can provide a company with unreliable and/or meaningless information.

 What is DDL Interpreter?

Ans: DDL Interpreter DDL expands to Data Definition Language. DDL Interpreter as the name suggests **interprets the DDL statements such as schema definition statements like create, delete, etc**. The result of this interpretation is a set of a table that contains the meta-data which is stored in the data dictionary.

 What is DML Compiler in SQL?

Ans: A data manipulation language (DML) is **a computer programming language used for adding (inserting), deleting, and modifying (updating) data in a database**. A DML is often a sublanguage of a broader database language such as SQL, with the DML comprising some of the operators in the language.

 What is SQL Key Constraints writing an Example of SQL Key Constraints

Ans: SQL constraints are used to specify rules for data in a table.

SQL constraints are used to specify rules for the data in a table.

Constraints are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the table. If there is any violation between the constraint and the data action, the action is aborted.

Constraints can be column level or table level. Column level constraints apply to a column, and table level constraints apply to the whole table.

The following constraints are commonly used in SQL:

* [NOT NULL](https://www.w3schools.com/sql/sql_notnull.asp) - Ensures that a column cannot have a NULL value
* [UNIQUE](https://www.w3schools.com/sql/sql_unique.asp) - Ensures that all values in a column are different
* [PRIMARY KEY](https://www.w3schools.com/sql/sql_primarykey.asp) - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
* [FOREIGN KEY](https://www.w3schools.com/sql/sql_foreignkey.asp) - Prevents actions that would destroy links between tables
* [CHECK](https://www.w3schools.com/sql/sql_check.asp) - Ensures that the values in a column satisfies a specific condition
* [DEFAULT](https://www.w3schools.com/sql/sql_default.asp) - Sets a default value for a column if no value is specified
* [CREATE INDEX](https://www.w3schools.com/sql/sql_create_index.asp) - Used to create and retrieve data from the database very quickly

 What is save Point? How to create a save Point write a Query?

Ans:  Savepoint is a command in SQL that is used with the rollback command.

To use the TCL commands in SQL, we first need to initiate the transaction by using the BEGIN / START TRANSACTION command.

1. mysql> START TRANSACTION;

We will save our initiated transaction using the SAVEPOINT command along with some specific names of this savepoint.

1. mysql> SAVEPOINT ini;

Here, we have saved the initiated transaction with the name of 'ini'.

 What is trigger and how to create a Trigger in SQL?

Ans: A trigger is a set of SQL statements that reside in system memory with unique names. It is a specialized category of stored procedure that is called automatically when a database server event occurs. Each trigger is always associated with a table.

A **trigger is called a special procedure** because it cannot be called directly like a stored procedure. The key distinction between the trigger and procedure is that a trigger is called automatically when a data modification event occurs against a table. A stored procedure, on the other hand, must be invoked directly.

The following are the main characteristics that distinguish triggers from stored procedures:

* We cannot manually execute/invoked triggers.
* Triggers have no chance of receiving parameters.
* A transaction cannot be committed or rolled back inside a trigger.

## Syntax of Trigger

We can create a trigger in [SQL Server](https://www.javatpoint.com/sql-server-tutorial) by using the **CREATE TRIGGER** statement as follows:

1. CREATE TRIGGER schema.trigger\_name
2. ON table\_name
3. AFTER  {INSERT, UPDATE, DELETE}
4. [NOT FOR REPLICATION]
5. AS
6. {SQL\_Statements}

The parameter descriptions of this syntax illustrate below:

* **schema:** It is an optional parameter that defines which schema the new trigger belongs to.
* **trigger\_name:** It is a required parameter that defines the name for the new trigger.
* **table\_name:** It is a required parameter that defines the table name to which the trigger applies. Next to the table name, we need to write the AFTER clause where any events like INSERT, UPDATE, or DELETE could be listed.
* **NOT FOR REPLICATION:** This option tells that [SQL](https://www.javatpoint.com/sql-tutorial) Server does not execute the trigger when data is modified as part of a replication process.

Task

1. Create Table Name : Student and Exam

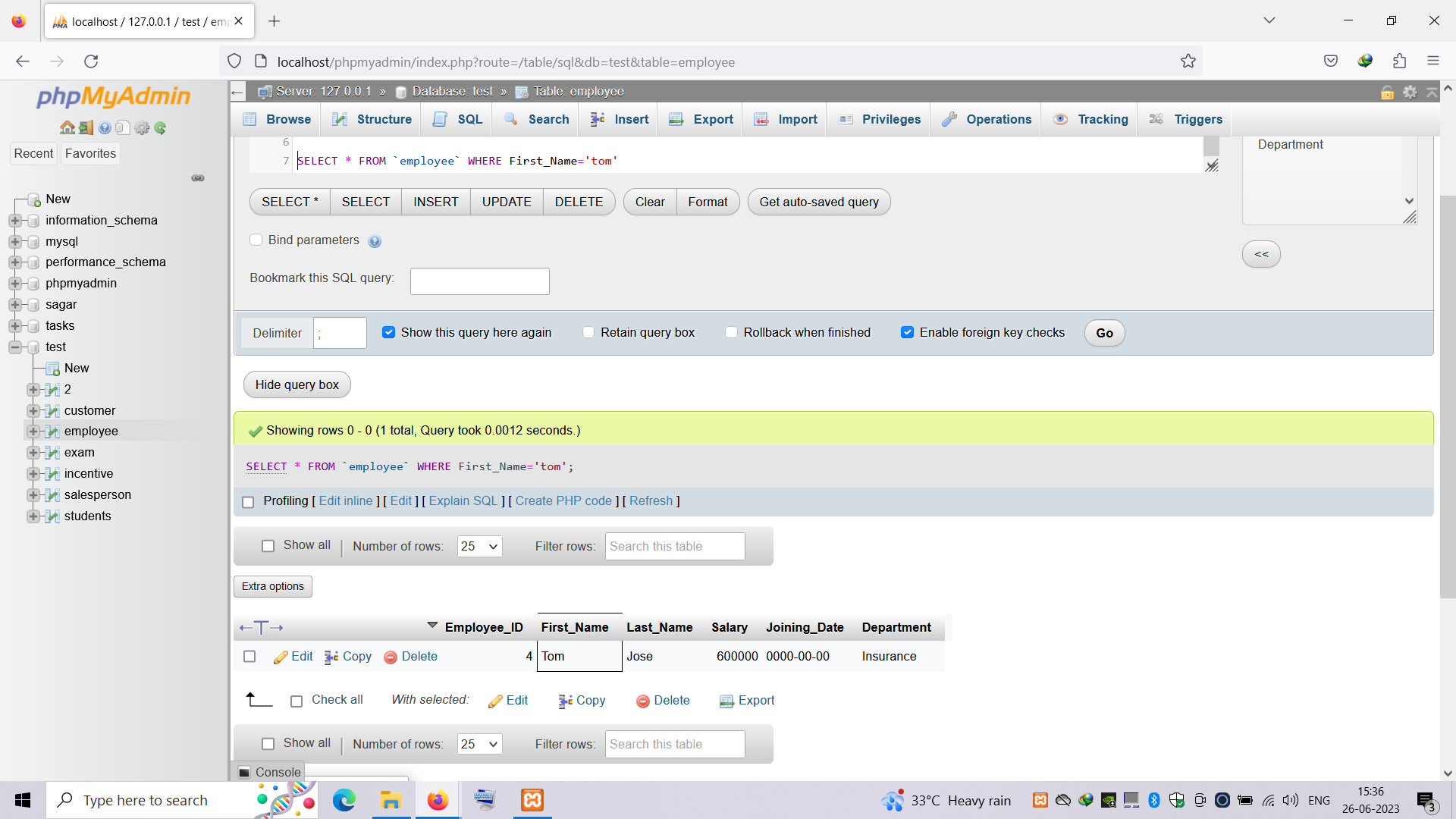
2. Create table given below

3. Create table given below: Employee and Incentive

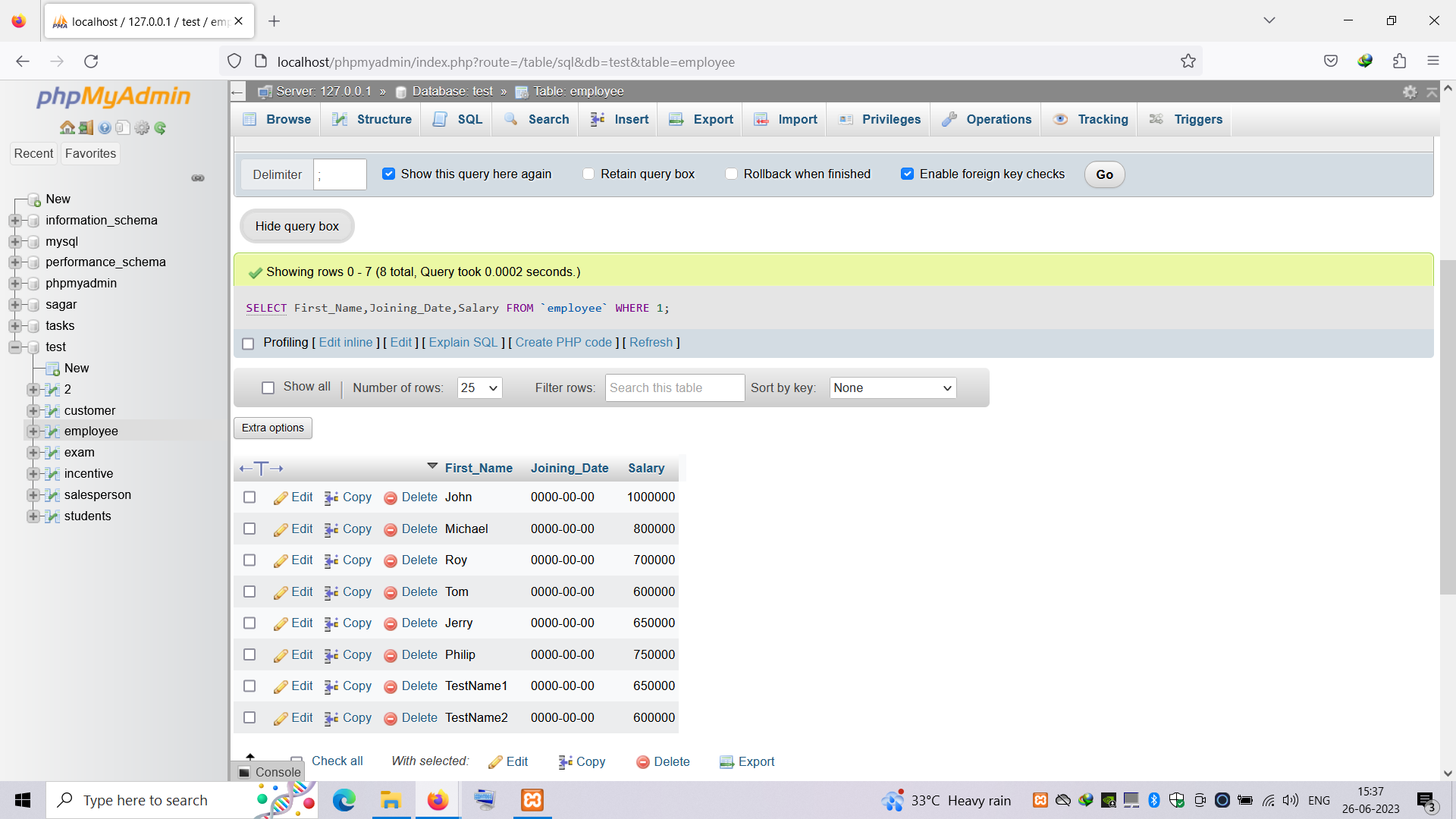
Table Name: Employee

Table Name: Incentive

Get First\_Name from employee table using Tom name “Employee Name”.

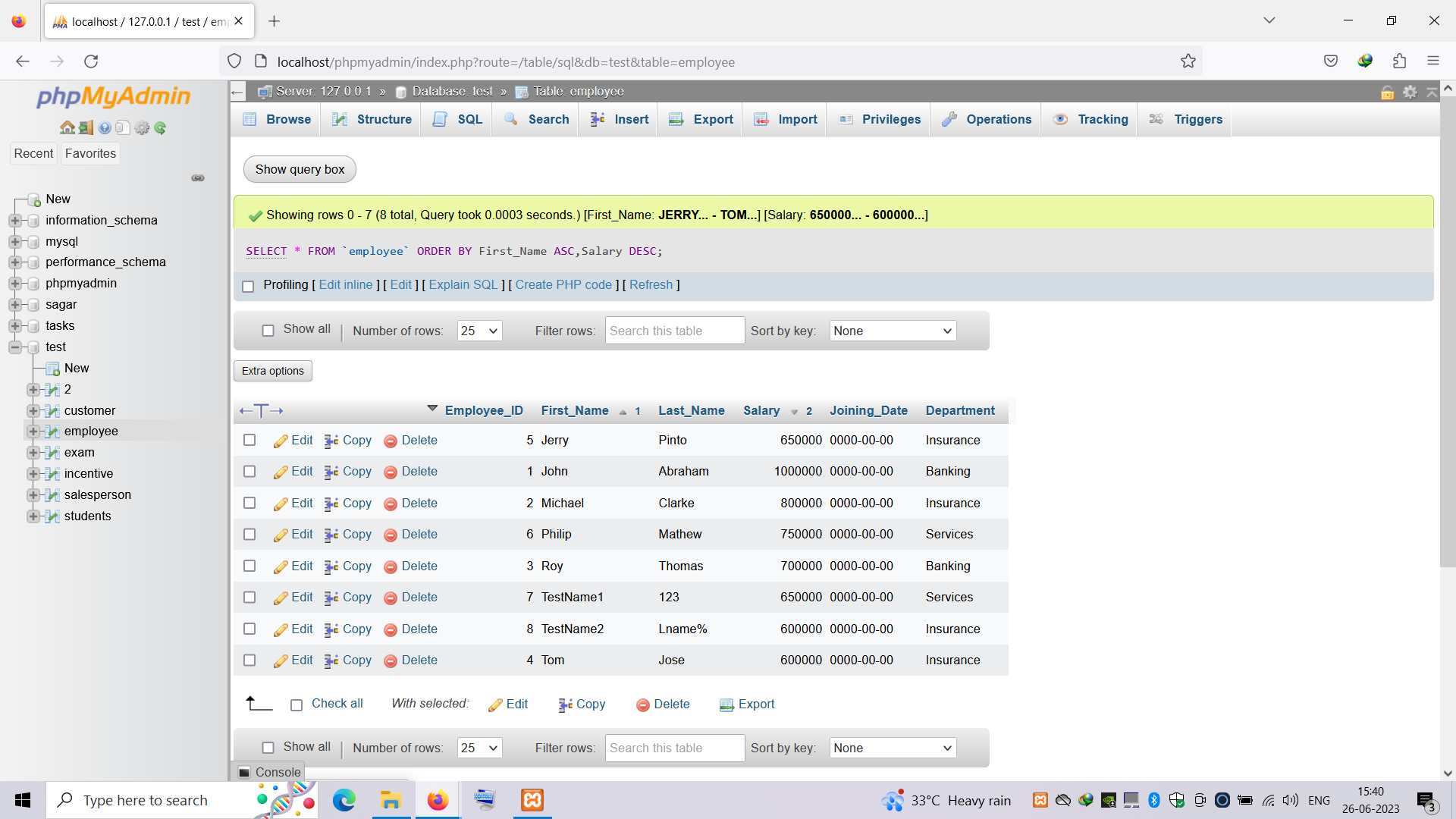


Get FIRST\_NAME, Joining Date, and Salary from employee table.

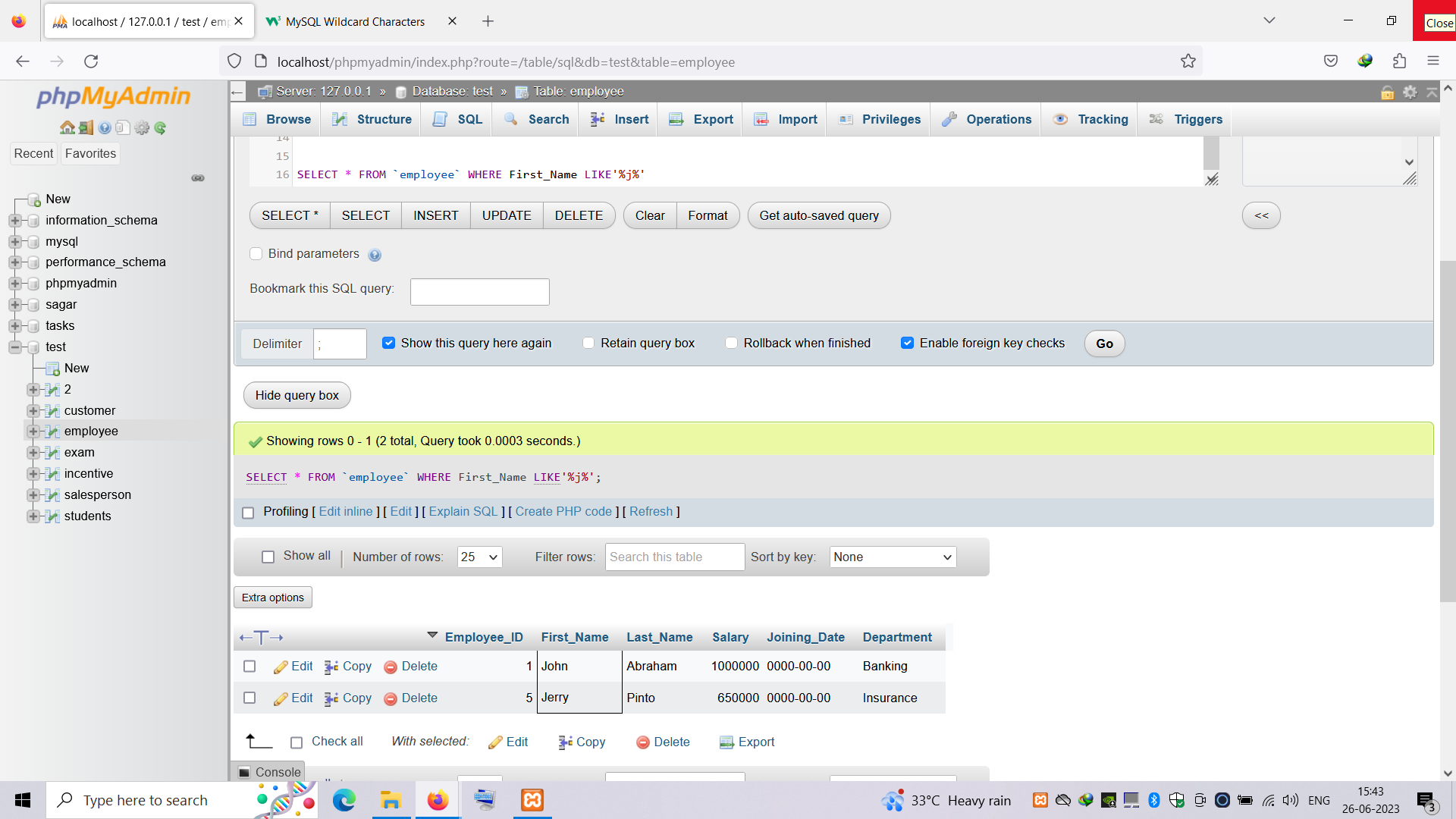


c) Get all employee details from the employee table order by First\_Name

Ascending and Salary descending?

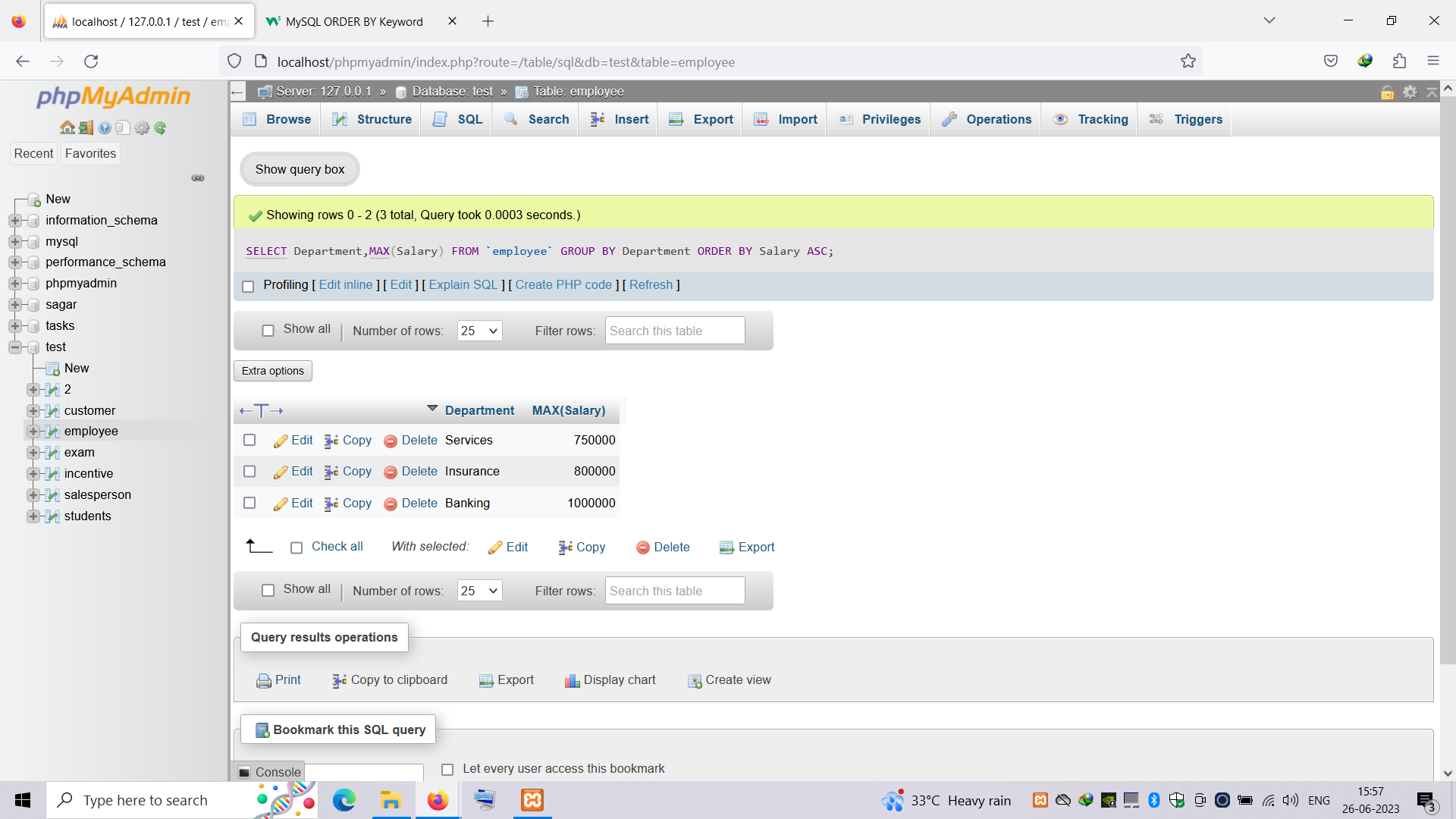


d) Get employee details from employee table whose first name contains ‘J’.



e) Get department wise maximum salary from employee table order by salary

ascending?

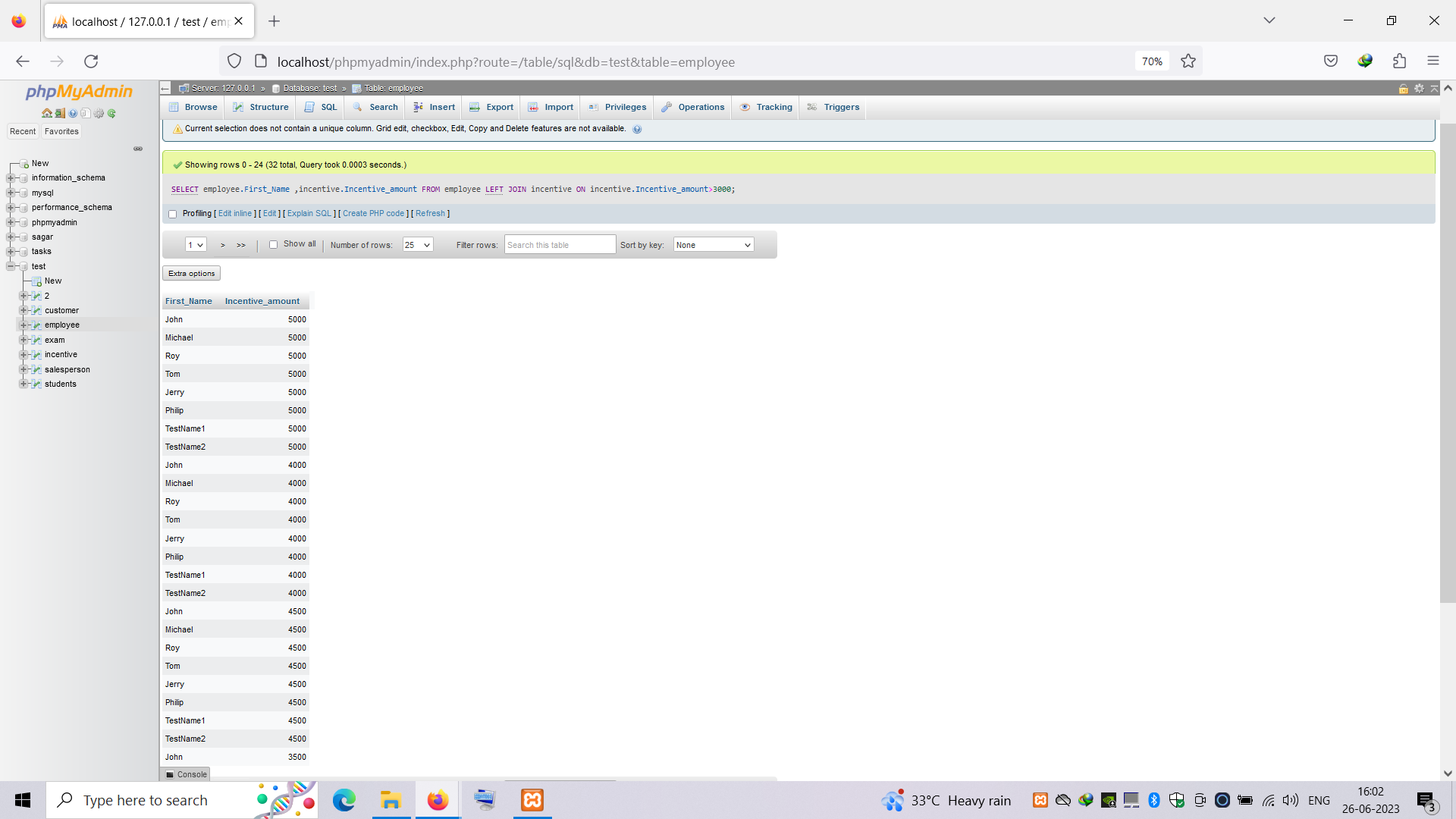


f) Select first\_name, incentive amount from employee and incentives table for

those employees who have incentives and incentive amount greater than 3000

g) Create After Insert trigger on Employee table which insert records in view

table

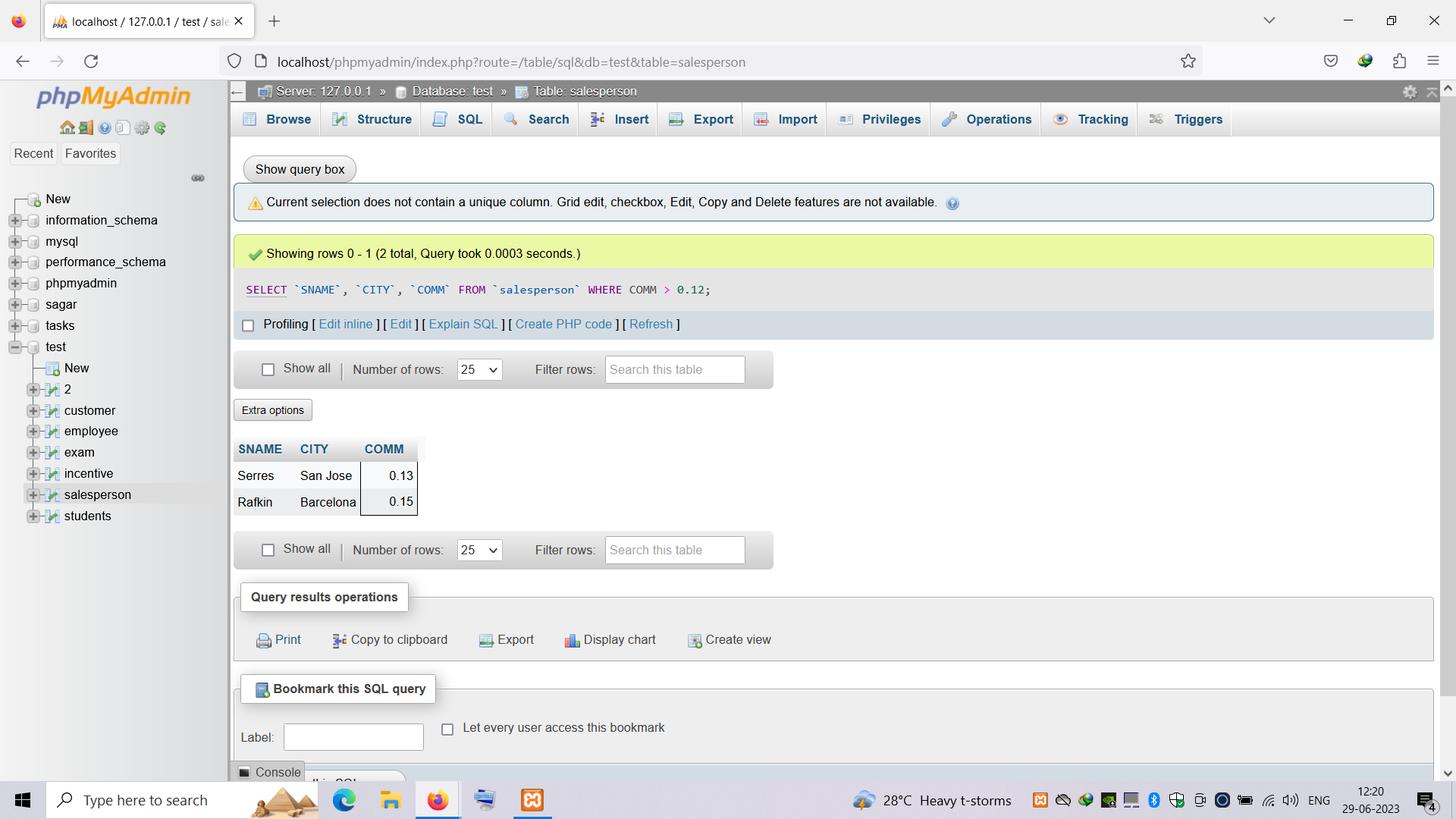


4. Create table given below: Salesperson and Customer

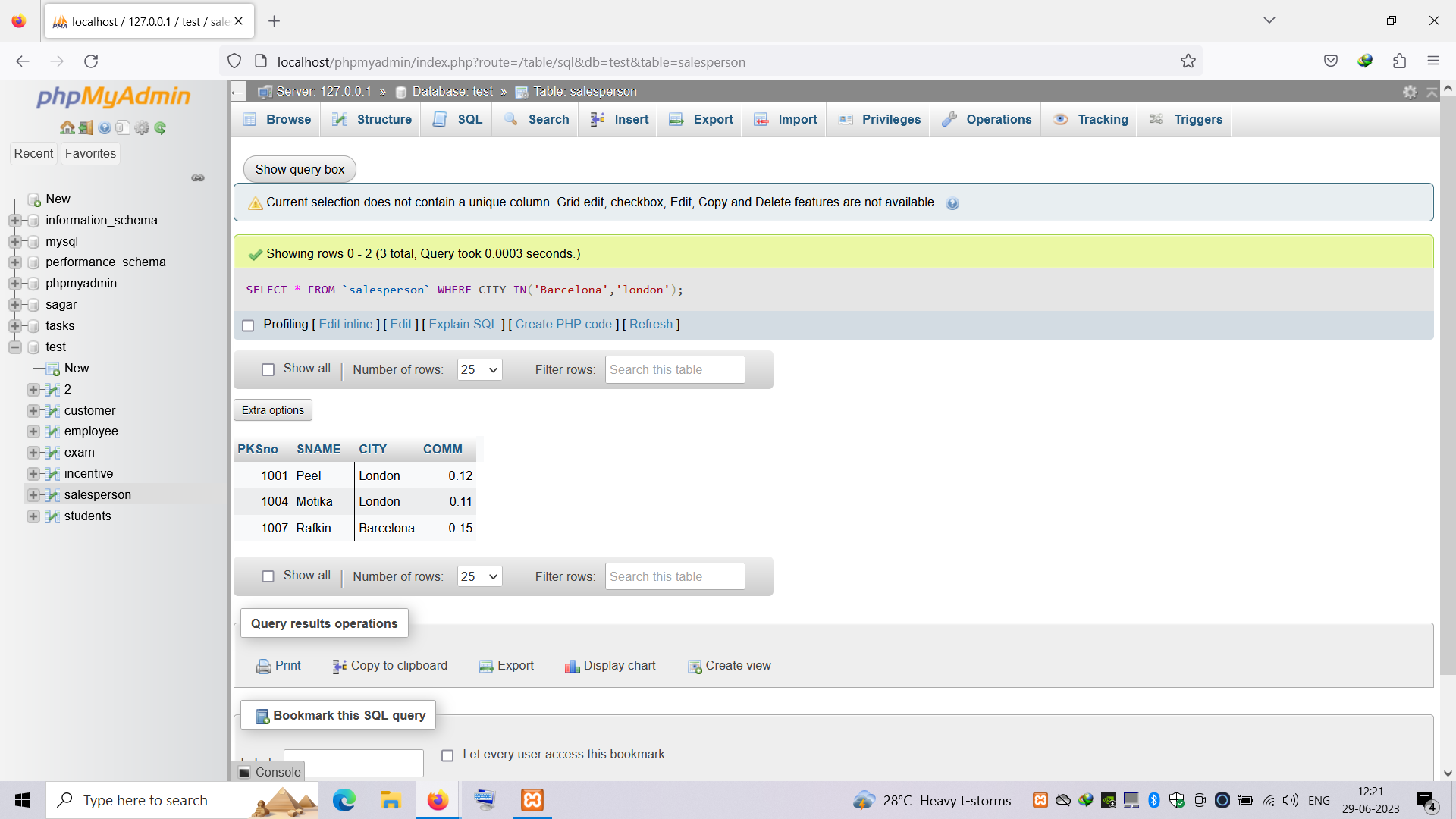
Retrieve the below data from above table

a) All orders for more than $1000.

b) Names and cities of all salespeople in London with commission above 0.12

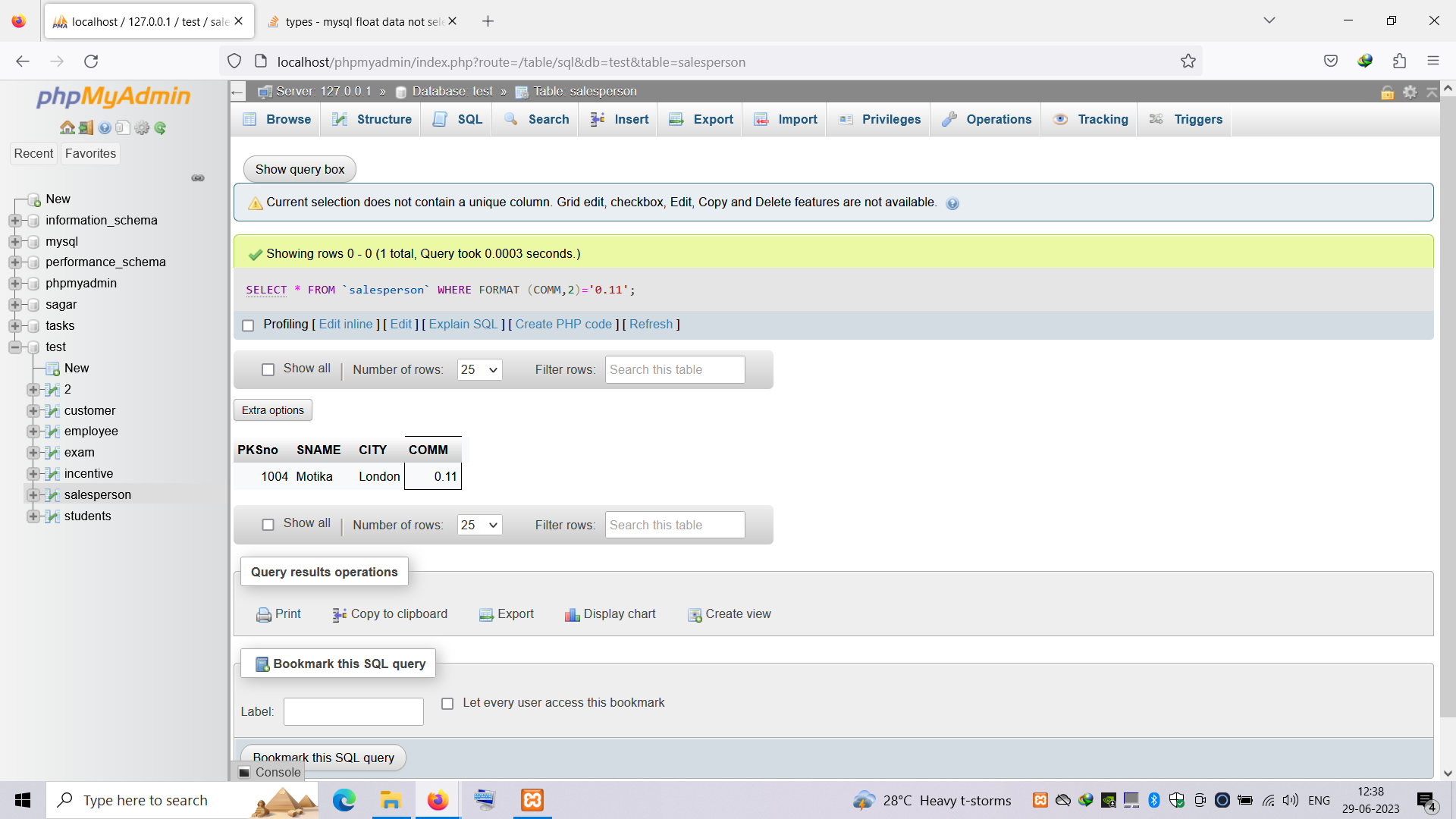


c) All salespeople either in Barcelona or in London



d) All salespeople with commission between 0.10 and 0.12. (Boundary values

should be excluded).



e) All customers excluding those with rating <= 100 unless they are located in

Roe

