***Q1. What do you understand By Database***

***Ans. A database is an organized collection of data, so that it can be easily accessed and managed.***

***You can organize data into tables, rows, columns, and index it to make it easier to find relevant information.***

***Database handlers create a database in such a way that only one set of software program provides access of data to all the users.***

***The main purpose of the database is to operate a large amount of information by storing, retrieving, and managing data.***

***There are many dynamic websites on the World Wide Web nowadays which are handled through databases. For example, a model that checks the availability of rooms in a hotel. It is an example of a dynamic website that uses a database.***

***Q2. What is Normalization?***

***Ans.***

* ***Normalization is the process of organizing the data in the database.***
* ***Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies.***
* ***Normalization divides the larger table into smaller and links them using relationships.***
* ***The normal form is used to reduce redundancy from the database table.***

***The main reason for normalizing the relations is removing these anomalies. Failure to eliminate anomalies leads to data redundancy and can cause data integrity and other problems as the database grows. Normalization consists of a series of guidelines that helps to guide you in creating a good database structure.***

***Q3. What is Difference between DBMS and RDBMS?***

***Ans. Although DBMS and RDBMS both are used to store information in physical database but there are some remarkable differences between them.***

***The main differences between DBMS and RDBMS are given below:***

|  |  |  |
| --- | --- | --- |
| ***No*** | ***DBMS*** | ***RDBMS*** |
| ***1*** | ***DBMS stores data as file.*** | ***RDBMS stores data in tabular form.*** |
| ***2*** | ***Data elements need to access individually.*** | ***Multiple data elements can be accessed at the same time.*** |
| ***3*** | ***No relationship between data.*** | ***Data is stored in the form of tables which are related to each other.*** |
| ***4*** | ***Normalization is not present.*** | ***Normalization is present.*** |
| ***5*** | ***DBMS does not support distributed database.*** | ***RDBMS supports distributed database.*** |
| ***6*** | ***It stores data in either a navigational or hierarchical form.*** | ***It uses a tabular structure where the headers are the column names, and the rows contain corresponding values.*** |
| ***7*** | ***It deals with small quantity of data.*** | ***It deals with large amount of data.*** |
| ***8*** | ***Data redundancy is common in this model.*** | ***Keys and indexes do not allow Data redundancy.*** |
| ***9*** | ***It is used for small organization and deal with small data.*** | ***It is used to handle large amount of data.*** |
| ***10*** | ***Not all Codd rules are satisfied.*** | ***All 12 Codd rules are satisfied.*** |
| ***11*** | ***Security is less*** | ***More security measures provided.*** |
| ***12*** | ***It supports single user.*** | ***It supports multiple users.*** |
| ***13*** | ***Data fetching is slower for the large amount of data.*** | ***Data fetching is fast because of relational approach.*** |
| ***14*** | ***The data in a DBMS is subject to low security levels with regards to data manipulation.*** | ***There exists multiple levels of data security in a RDBMS.*** |
| ***15*** | ***Low software and hardware necessities.*** | ***Higher software and hardware necessities.*** |
| ***16*** | ***Examples: XML, Window Registry, Forxpro, dbaseIIIplus etc.*** | ***Examples: MySQL, PostgreSQL, SQL Server, Oracle, Microsoft Access etc.*** |

***Q4. What is MF Cod Rule of RDBMS Systems?***

***Ans. Every database has tables, and constraints cannot be referred to as a rational database system. And if any database has only relational data model, it cannot be a Relational Database System (RDBMS). So, some rules define a database to be the correct RDBMS. These rules were developed by Dr. Edgar F. Codd (E.F. Codd) in 1985, who has vast research knowledge on the Relational Model of database Systems. Codd presents his 13 rules for a database to test the concept of DBMS against his relational model, and if a database follows the rule, it is called a true relational database (RDBMS). These 13 rules are popular in RDBMS, known as Codd's 12 rules.***

***Rule 0: The Foundation Rule***

***Rule 1: Information Rule***

***Rule 2: Guaranteed Access Rule***

***Rule 3: Systematic Treatment of Null Values***

***Rule 4: Active/Dynamic Online Catalog based on the relational model***

***Rule 5: Comprehensive Data SubLanguage Rule***

***Rule 6: View Updating Rule***

***Rule 7: Relational Level Operation (High-Level Insert, Update and delete) Rule***

***Rule 8: Physical Data Independence Rule***

***Rule 9: Logical Data Independence Rule***

***Rule 10: Integrity Independence Rule***

***Rule 11: Distribution Independence Rule***

***Rule 12: Non Subversion Rule***

***Q5 What do you understand By Data Redundancy?***

***Ans . In DBMS, when the same data is stored in different tables, it causes data redundancy.***

***Sometimes, it is done on purpose for recovery or backup of data, faster access of data, or updating data easily. Redundant data costs extra money, demands higher storage capacity, and requires extra effort to keep all the files up to date.***

***Sometimes, unintentional duplicity of data causes a problem for the database to work properly, or it may become harder for the end user to access data. Redundant data unnecessarily occupy space in the database to save identical copies, which leads to space constraints, which is one of the major problems.***

***Let us understand redundancy in DBMS properly with the help of an example.***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Student\_id*** | ***Name*** | ***Course*** | ***Session*** | ***Fee*** | ***Department*** |
| ***101*** | ***Devi*** | ***B. Tech*** | ***2022*** | ***90,000*** | ***CS*** |
| ***102*** | ***Sona*** | ***B. Tech*** | ***2022*** | ***90,000*** | ***CS*** |
| ***103*** | ***Varun*** | ***B. Tech*** | ***2022*** | ***90,000*** | ***CS*** |
| ***104*** | ***Satish*** | ***B. Tech*** | ***2022*** | ***90,000*** | ***CS*** |
| ***105*** | ***Amisha*** | ***B. Tech*** | ***2022*** | ***90,000*** | ***CS*** |

***In the above example, there is a "Student" table that contains data such as "Student\_id", "Name", "Course", "Session", "Fee", and "Department". As you can see, some data is repeated in the table, which causes redundancy.***

***Q6 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.***

***Q7 What is DML Compiler in SQL?***

***Ans DML Compiler: Translates DML statements in a query language within low level instructions understandable through the query evaluation engine. Attempts to transforms users request within an equivalent and well-organized from for executing the query understandable through Data Manager, Interprets DDL statements and records them within a set of tables containing Meta data in a form that can be used through other elements of a DBMS.***

***Q8 What is SQL Key Constraints writing an Example of SQL Key Constraints***

***Ans 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 - Ensures that a column cannot have a NULL value***

***UNIQUE - Ensures that all values in a column are different***

***PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table***

***FOREIGN KEY - Prevents actions that would destroy links between tables***

***CHECK - Ensures that the values in a column satisfies a specific condition***

***DEFAULT - Sets a default value for a column if no value is specified***

***CREATE INDEX - Used to create and retrieve data from the database very quickly***

***Not null***

***CREATE TABLE TableName (ColumnName1 datatype NOT NULL, ColumnName2 datatype,…., ColumnNameN datatype);***

***UNIQUE***

***CREATE TABLE TableName (ColumnName1 datatype UNIQUE, ColumnName2 datatype,…., ColumnNameN datatype);***

***PRIMARY KEY***

***CREATE TABLE TableName (ColumnName1 datatype PRIMARY KEY, ColumnName2 datatype,…., ColumnNameN datatype);***

***FOREIGN KEY***

***CREATE TABLE tablename(ColumnName1 Datatype(SIZE) PRIMARY KEY, ColumnNameN Datatype(SIZE), FOREIGN KEY( ColumnName ) REFERENCES PARENT\_TABLE\_NAME(Primary\_Key\_ColumnName));***

***CHECK***

***CREATE TABLE TableName (ColumnName1 datatype CHECK (ColumnName1 Condition), ColumnName2 datatype,…., ColumnNameN datatype);***

***DEFAULT***

***CREATE TABLE TableName (ColumnName1 datatype DEFAULT Value, ColumnName2 datatype,…., ColumnNameN datatype);***

***CREATE INDEX***

***CREATE INDEX IndexName ON TableName (ColumnName 1);***

***Q 9. 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.***

***It is a command in Transaction Control Language that is used to mark the transaction in a table.***

***Consider you are making a very long table, and you want to roll back only to a certain position in a table then; this can be achieved using the savepoint.***

***If you made a transaction in a table, you could mark the transaction as a certain name, and later on, if you want to roll back to that point, you can do it easily by using the transaction's name.***

***Savepoint is helpful when we want to roll back only a small part of a table and not the whole table. In simple words, we can say savepoint is a bookmark in SQL.***

***CREATE TABLE student(ID INT, Name VARCHAR(20), Percentage INT, Location VARCHAR(20), DateOfBirth DATE);***

***Q10 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.***

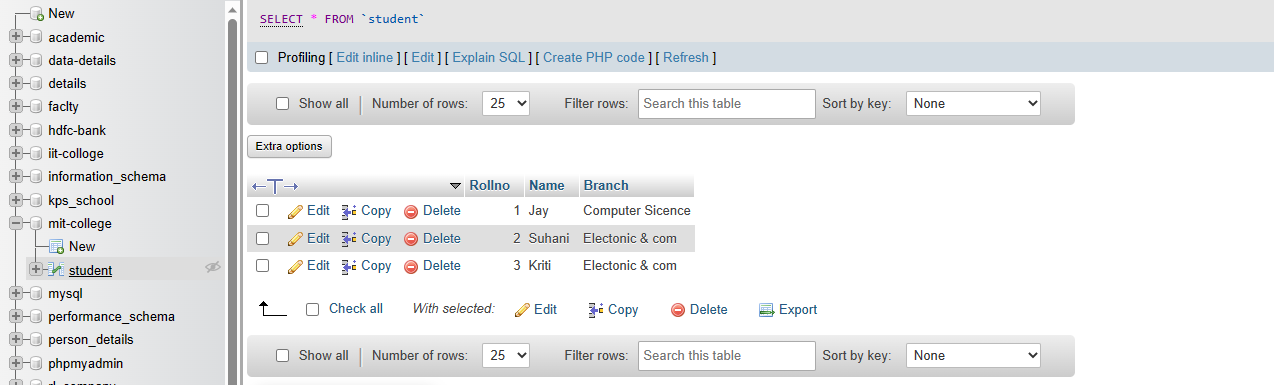
***Task***

1. ***Create Table Name : Student and Exam***

***CREATE TABLE Student( Rollno int(20)NOT NULL PRIMARY KEY , Name varchar (20) , Branch Varchar (20));***

***INSERT INTO `student` (`Rollno`, `Name`, `Branch`) VALUES ('1', 'Jay', 'Computer Sicence');***

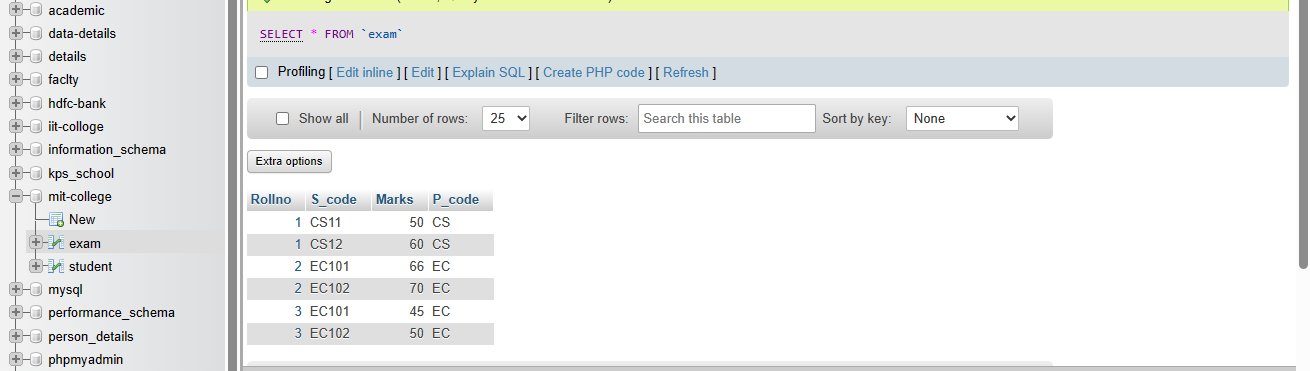
***Table 1 Student***

******

***CREATE TABLE Exam( Rollno int(20)NOT NULL , S\_code varchar (20) , Marks int (20) , P\_code varchar (20) , FOREIGN KEY (Rollno) REFERENCES student(Rollno));***

***INSERT INTO `exam` (`Rollno`, `S\_code`, `Marks`, `P\_code`) VALUES ('1', 'CS11', '50', 'CS');***

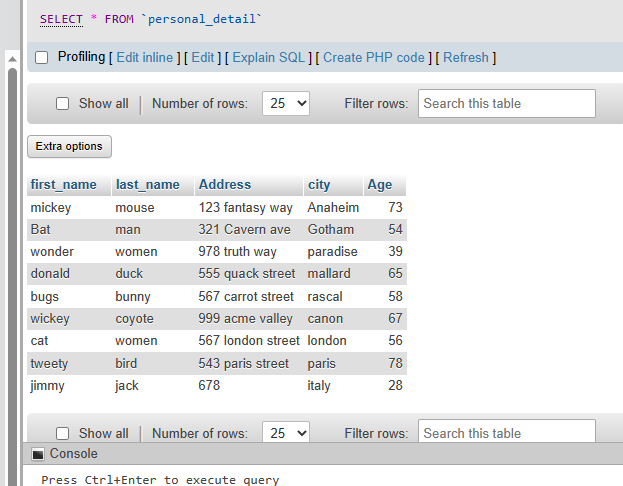
***Table 2 Exam***

******

1. ***Create table given below***

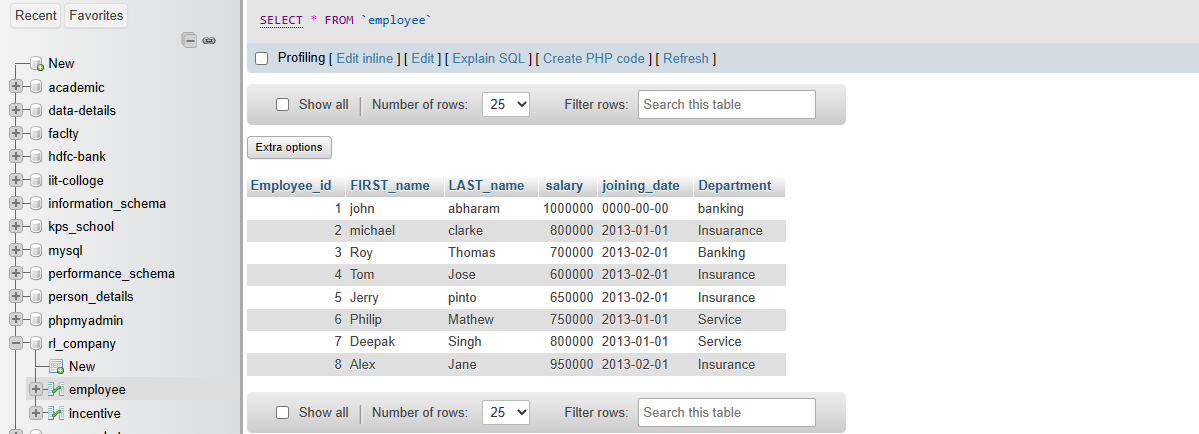
***CREATE TABLE personal\_detail(first\_name varchar(20) , last\_name varchar(20) , Address varchar(50), city varchar (20) , Age int(20));***

***INSERT INTO `personal\_detail` (`first\_name`, `last\_name`, `Address`, `city`, `Age`) VALUES ('mickey', 'mouse', '123 fantasy way', 'Anaheim', '73');***

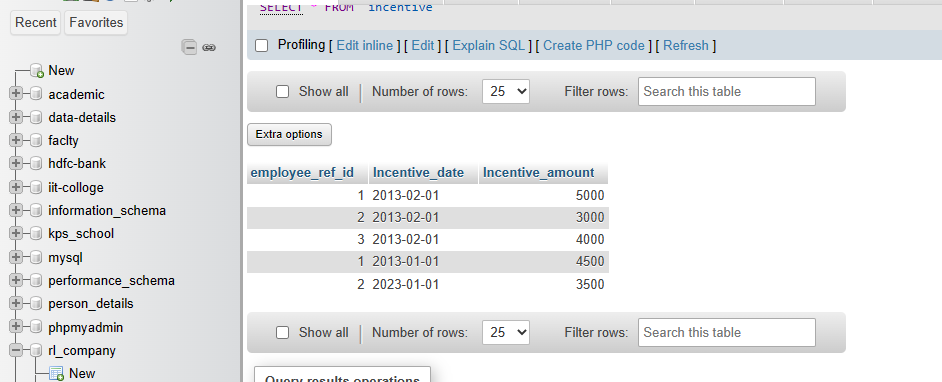
******

1. ***Create table given below: Employee and Incentive***

***Table Name: Employee***

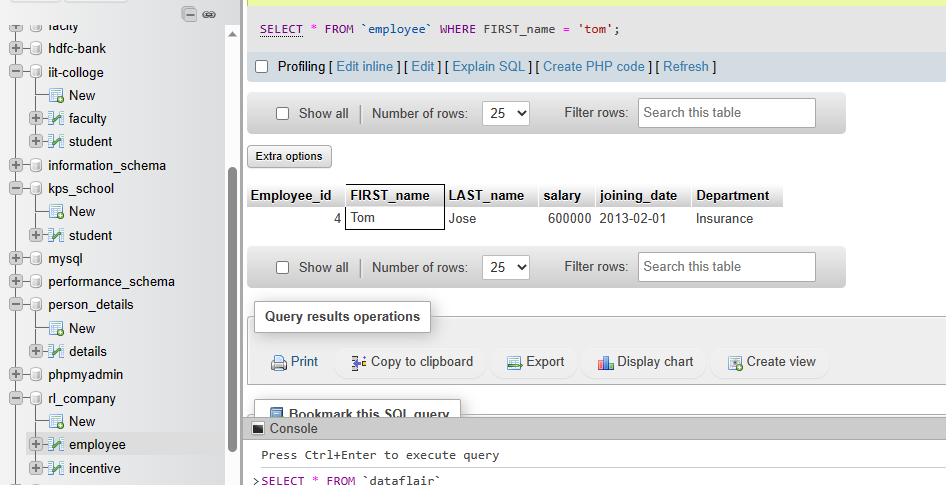
******

***Table Name: Incentive***

******

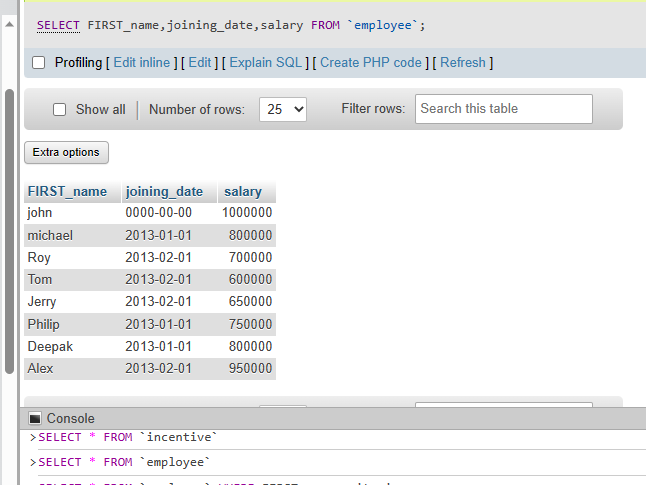
1. ***Get First\_Name from employee table using Tom name “Employee Name”.***

***SELECT \* FROM `employee` WHERE FIRST\_name = 'tom';***

******

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

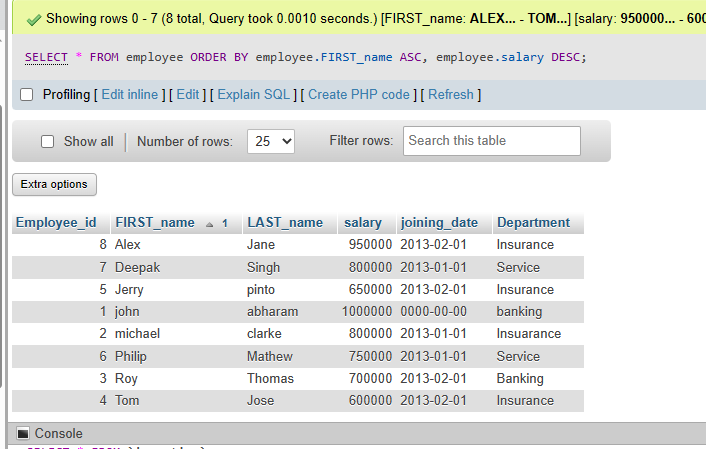
***SELECT FIRST\_name,joining\_date,salary FROM `employee`;***

******

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

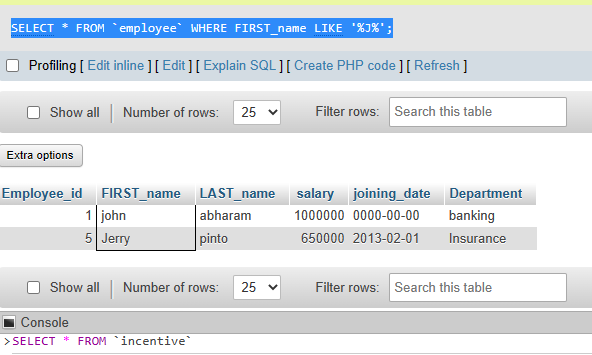
***Ascending and Salary descending?***

***SELECT \* FROM employee ORDER BY employee.FIRST\_name ASC, employee.salary DESC;***

******

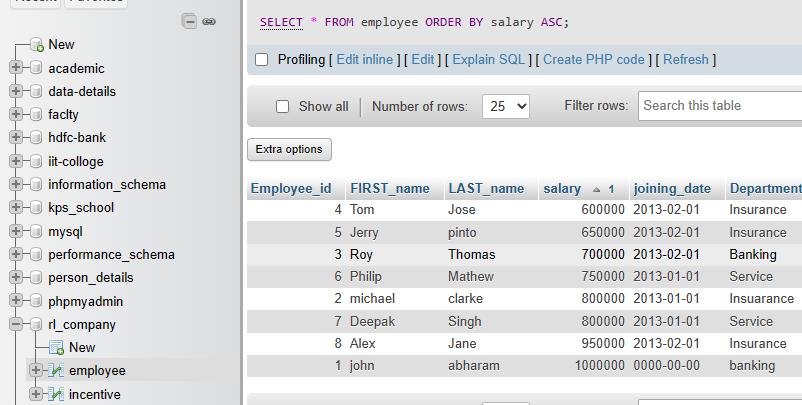
***d) Get employee details from employee table whose first name contains ‘J’.***

***SELECT \* FROM `employee` WHERE FIRST\_name LIKE '%J%';***

******

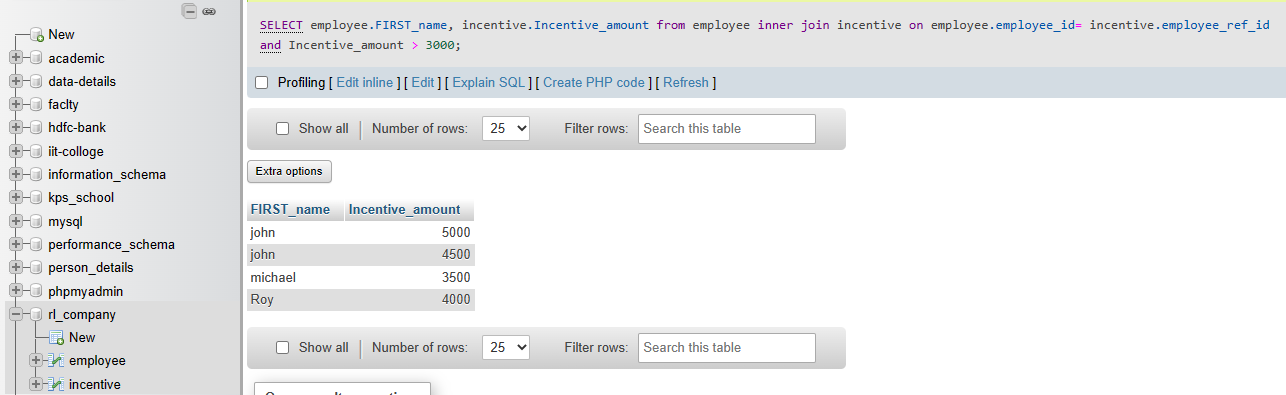
***e) Get department wise maximum salary from employee table order by salary ascending?***

***SELECT \* FROM employee ORDER BY salary ASC;***

******

***f) Select first\_name, incentive amount from employee and incentives table for those employees who have incentives and incentive amount greater than 3000***

***SELECT employee.FIRST\_name, incentive.Incentive\_amount from employee inner join incentive on employee.employee\_id= incentive.employee\_ref\_id and Incentive\_amount > 3000;***

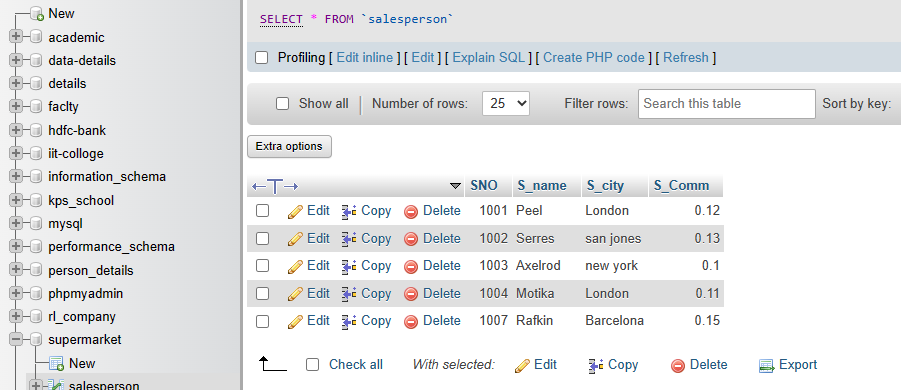
******

1. ***Create table given below: Salesperson and Customer***

***CREATE TABLE salesperson( SNO int(20) , S\_name varchar(20), S\_city varchar (20) , S\_Comm float(20) , PRIMARY KEY (SNO));***

***INSERT INTO `salesperson` (`S\_CNM`, `S\_name`, `S\_city`, `S\_Comm`) VALUES ('1001', 'PEEL', 'London', '.12');***

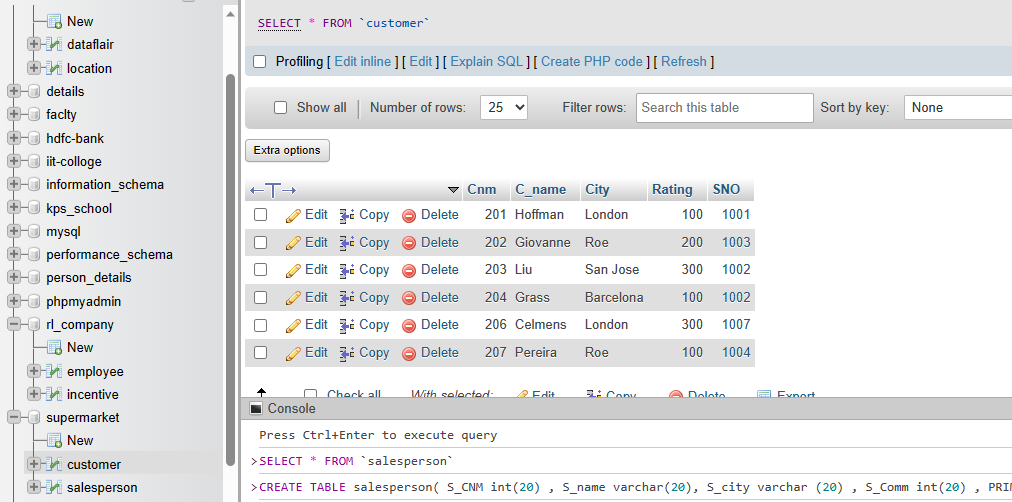
***Table 1 Salesperson***

******

***Table 2 Customer***

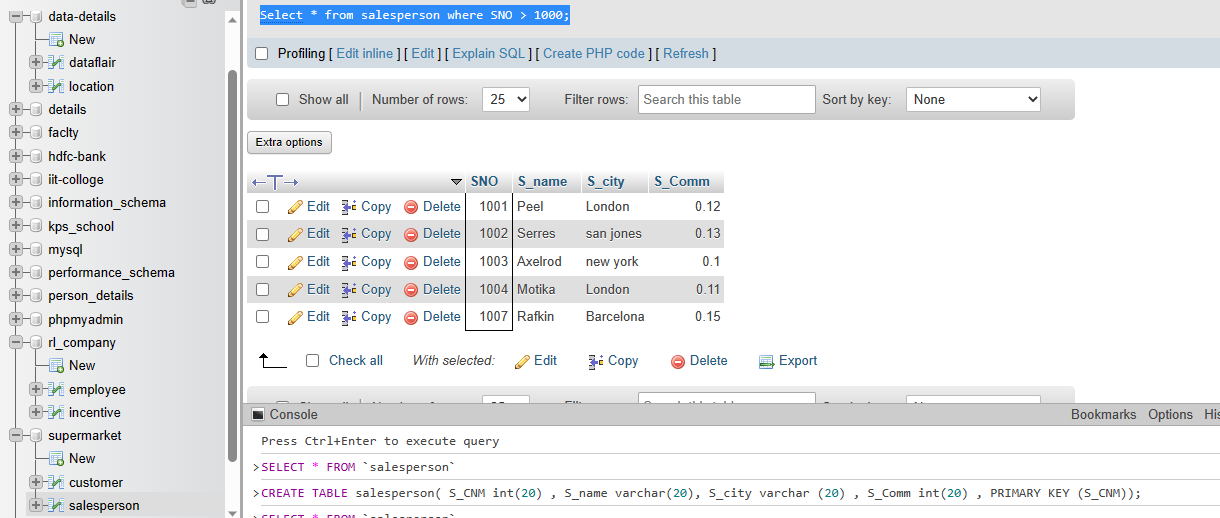
***CREATE TABLE Customer( Cnm int (20), C\_name varchar(20) , City varchar(20) , Rating int(20) , SNO int(20) , PRIMARY KEY (Cnm) ,FOREIGN KEY (SNO) REFERENCES salesperson (SNO));***

***INSERT INTO `customer` (`Cnm`, `C\_name`, `City`, `Rating`, `SNO`) VALUES ('202', 'Hoffman', 'London', '100', '1001');***

******

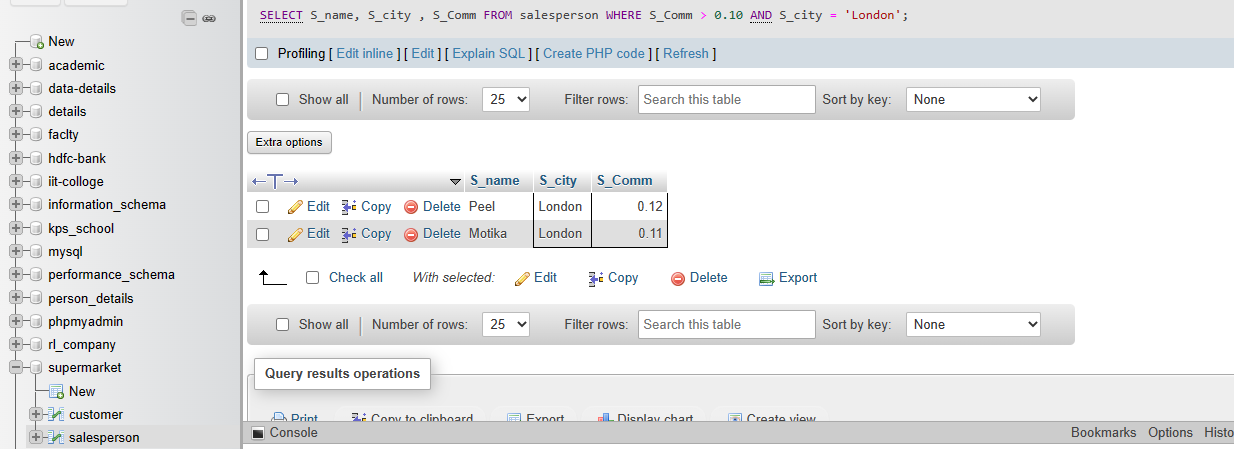
1. ***All orders for more than $1000.***

***Select \* from salesperson where SNO > 1000;***

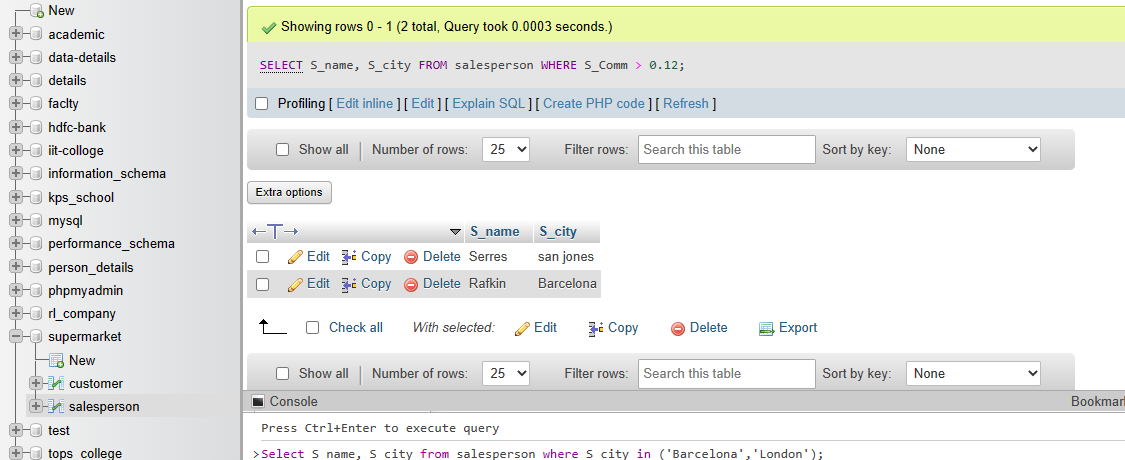
******

1. ***Names and cities of all salespeople in London with commission above 0.12***

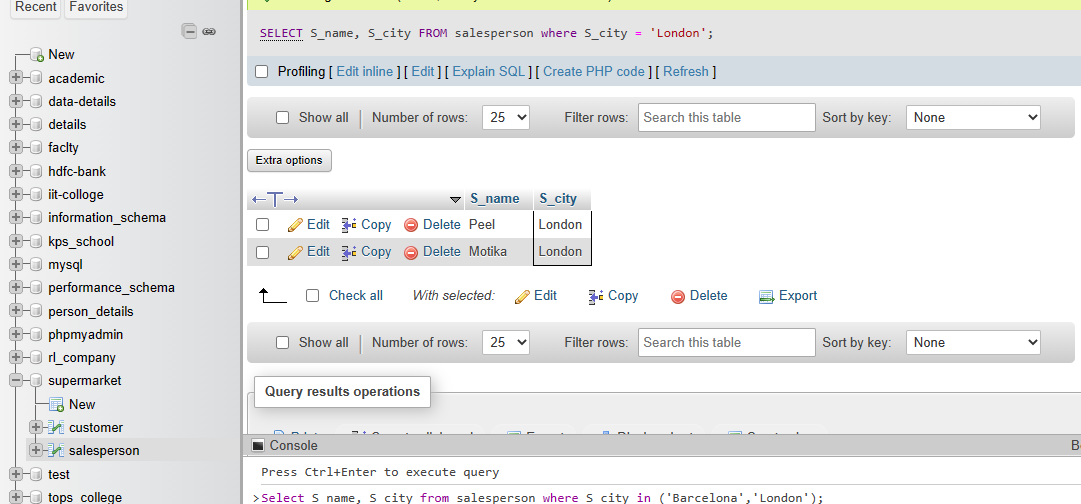
***SELECT S\_name, S\_city , S\_Comm FROM salesperson WHERE S\_Comm > 0.12 AND S\_city = 'London';***

******

***SELECT S\_name, S\_city FROM salesperson WHERE S\_Comm > 0.12;***

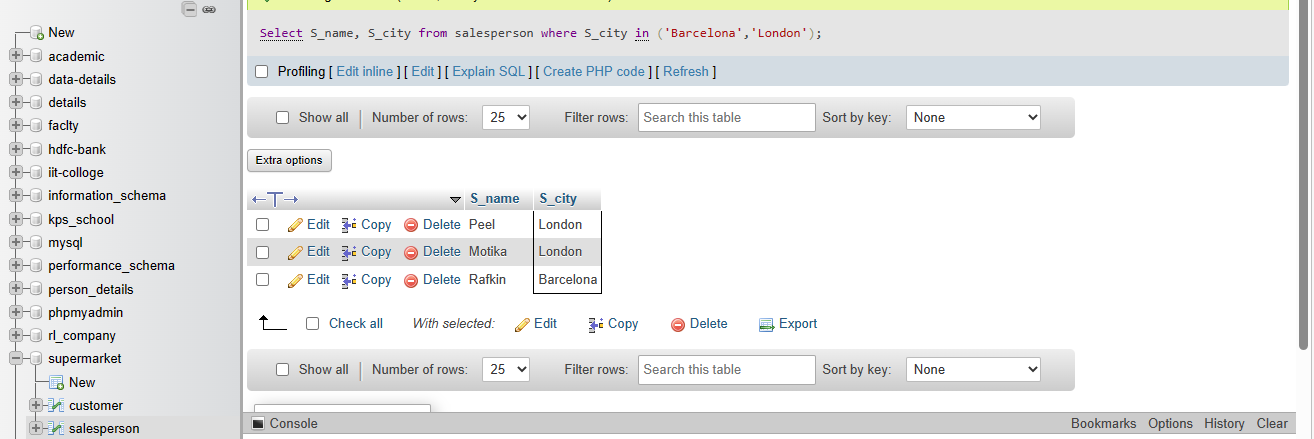
******

***SELECT S\_name, S\_city FROM salesperson where S\_city = 'London';***

******

1. ***All salespeople either in Barcelona or in London***

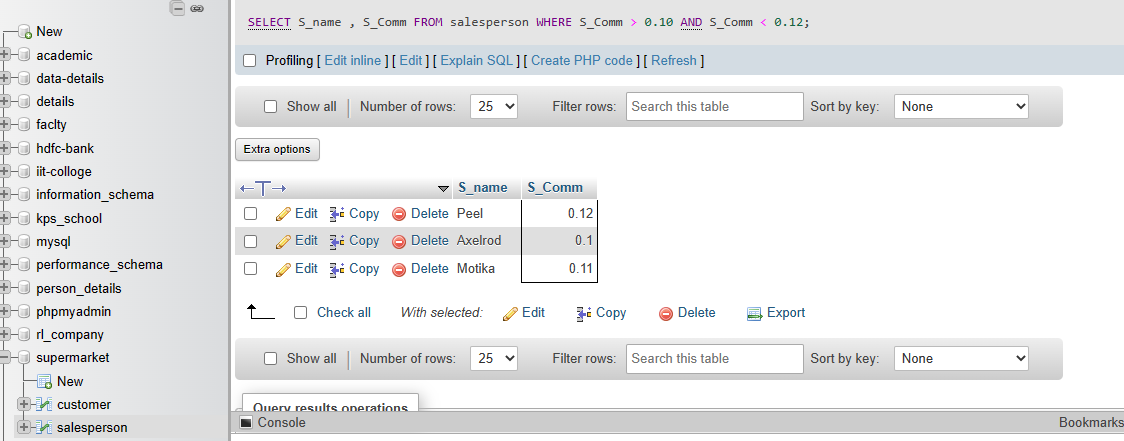
***Select S\_name, S\_city from salesperson where S\_city in ('Barcelona','London');***

******

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

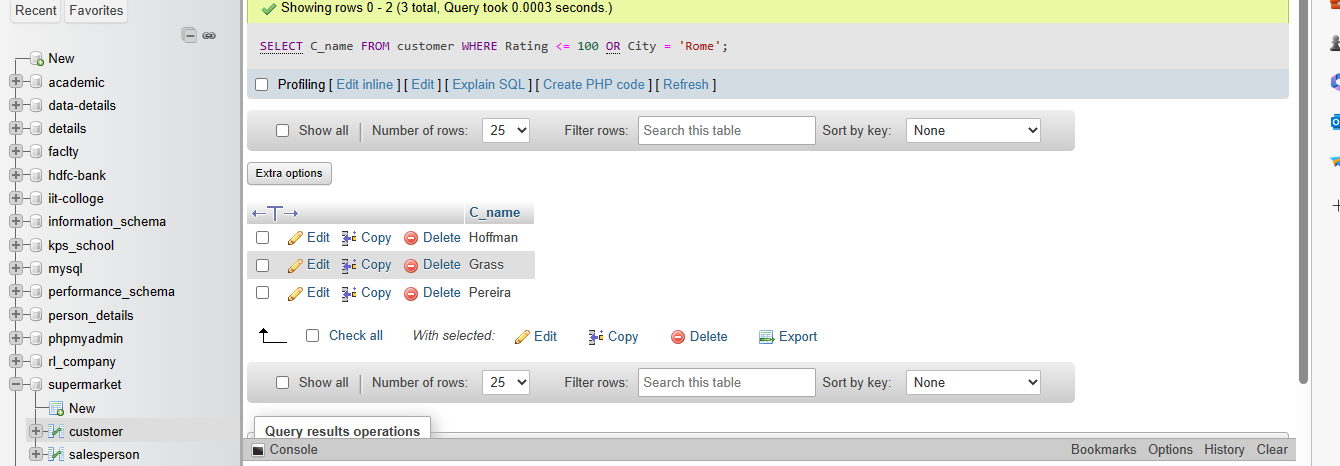
***should be excluded).***

***SELECT S\_name , S\_Comm FROM salesperson WHERE S\_Comm > 0.10 AND S\_Comm < 0.12;***

******

1. ***All customers excluding those with rating <= 100 unless they are located in Rome***

***SELECT C\_name FROM customer WHERE Rating <= 100 OR City = 'Rome';***

******