Name : sutariya dhruvik

Subject: DBMS assignment

1. 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. [It is usually controlled by a **database management system (DBMS)**](https://www.oracle.com/in/database/what-is-database/) . [Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a **database system**, often shortened to just database](https://www.oracle.com/in/database/what-is-database/) .

Data within the most common types of databases in operation today is typically modelled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. [Most databases use **Structured Query Language (SQL)** for writing and querying data](https://www.oracle.com/in/database/what-is-database/) .

Databases have evolved dramatically since their inception in the early 1960s. Navigational databases such as the hierarchical database (which relied on a tree-like model and allowed only a one-to-many relationship), and the network database (a more flexible model that allowed multiple relationships), were the original systems used to store and manipulate data. Although simple, these early systems were inflexible. In the 1980s, relational databases became popular, followed by object-oriented databases in the 1990s. [More recently, NoSQL databases came about as a response to the growth of the internet and the need for faster speed and processing of unstructured data](https://www.oracle.com/in/database/what-is-database/) .

The primary differences between a database and a spreadsheet (such as Microsoft Excel) are:

* How the data is stored and manipulated
* Who can access the data
* How much data can be stored

Spreadsheets were originally designed for one user, and their characteristics reflect that. They’re great for a single user or small number of users who don’t need to do a lot of incredibly complicated data manipulation. Databases, on the other hand, are designed to hold much larger collections of organized information—massive amounts, sometimes. [Databases allow multiple users at the same time to quickly and securely access and query the data using highly complex logic and language](https://www.oracle.com/in/database/what-is-database/) .

2. What is Normalization?

Ans: A **database** is an organized collection of structured information or data, typically stored electronically in a computer system. [It is usually controlled by a **database management system (DBMS)**](https://www.oracle.com/in/database/what-is-database/) . [Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a **database system**, often shortened to just database](https://www.oracle.com/in/database/what-is-database/) .

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. [Most databases use **Structured Query Language (SQL)** for writing and querying data](https://www.oracle.com/in/database/what-is-database/) .

Databases have evolved dramatically since their inception in the early 1960s. Navigational databases such as the hierarchical database (which relied on a tree-like model and allowed only a one-to-many relationship), and the network database (a more flexible model that allowed multiple relationships), were the original systems used to store and manipulate data. Although simple, these early systems were inflexible. In the 1980s, relational databases became popular, followed by object-oriented databases in the 1990s. [More recently, NoSQL databases came about as a response to the growth of the internet and the need for faster speed and processing of unstructured data](https://www.oracle.com/in/database/what-is-database/) .

The primary differences between a database and a spreadsheet (such as Microsoft Excel) are:

* How the data is stored and manipulated
* Who can access the data
* How much data can be stored

Spreadsheets were originally designed for one user, and their characteristics reflect that. They’re great for a single user or small number of users who don’t need to do a lot of incredibly complicated data manipulation. Databases, on the other hand, are designed to hold much larger collections of organized information—massive amounts, sometimes. [Databases allow multiple users at the same time to quickly and securely access and query the data using highly complex logic and language](https://www.oracle.com/in/database/what-is-database/) .

3. What is Difference between DBMS and RDBMS?

Ans:

|  |  |  |
| --- | --- | --- |
| **o.** | **DBMS** | **RDBMS** |
| 1) | DBMS applications store **data as file**. | RDBMS applications store **data in a tabular form**. |
| 2) | In DBMS, data is generally stored in either a hierarchical form or a navigational form. | In RDBMS, the tables have an identifier called primary key and the data values are stored in the form of tables. |
| 3) | **Normalization is not** present in DBMS. | **Normalization is** present in RDBMS. |
| 4) | DBMS does **not apply any security** with regards to data manipulation. | RDBMS **defines the integrity constraint** for the purpose of ACID (Atomocity, Consistency, Isolation and Durability) property. |
| 5) | DBMS uses file system to store data, so there will be **no relation between the tables**. | in RDBMS, data values are stored in the form of tables, so a **relationship** between these data values will be stored in the form of a table as well. |
| 6) | DBMS has to provide some uniform methods to access the stored information. | RDBMS system supports a tabular structure of the data and a relationship between them to access the stored information. |
| 7) | DBMS **does not support distributed database**. | RDBMS **supports distributed database**. |
| 8) | DBMS is meant to be for small organization and **deal with small data**. it supports **single user**. | RDBMS is designed to **handle large amount of data**. it supports **multiple users**. |
| 9) | Examples of DBMS are file systems, **xml** etc. | Example of RDBMS are **mysql**, **postgre**, **sql server**, **oracle** etc. |

4. What is MF Cod Rule of RDBMS Systems?

Ans: [**Codd’s Rule for Relational DBMS** is a set of rules proposed by **E.F. Codd**, the inventor of the relational model for database management](https://www.studytonight.com/dbms/codd-rule.php) . [These rules were designed to test the concept of a DBMS against his relational model and define the qualities required for a DBMS to become a Relational Database Management System (RDBMS)](https://www.studytonight.com/dbms/codd-rule.php) . [Although there are 13 Codd’s rules, only a few commercial products follow all of them](https://www.studytonight.com/dbms/codd-rule.php) .

Here is an overview of some of the key Codd’s rules:

1. [**Rule Zero**: For a system to qualify as an RDBMS, it must be able to manage the database entirely through relational capabilities](https://www.studytonight.com/dbms/codd-rule.php) .
2. **Information Rule** : All information, including metadata, should be represented as stored data in cells of tables. [The rows and columns should be strictly unordered](https://www.studytonight.com/dbms/codd-rule.php) .
3. **Guaranteed Access**: Each unique piece of data (atomic value) should be accessible using the table name, primary key (row), and attribute (column). [Direct access via pointers is a violation of this rule](https://www.studytonight.com/dbms/codd-rule.php) .
4. **Systematic Treatment of NULL**: Null values, which can represent missing data, not applicable, or no value, should be handled consistently. [Primary keys must not be null, and expressions involving null values should yield null results](https://www.studytonight.com/dbms/codd-rule.php) .
5. **Active Online Catalog** : The database dictionary or catalog, which describes the structure of the complete database, should be stored online. [The catalog should follow the same rules as the rest of the database, and the same query language used for querying the database should also apply to the catalog](https://www.studytonight.com/dbms/codd-rule.php) .
6. **Powerful and Well-Structured Language** : A well-structured language, such as SQL, should provide various access methods to retrieve data from the database. [If a database allows access without using this language, it violates this rule](https://www.studytonight.com/dbms/codd-rule.php) .
7. [**View Updatability** : All theoretically updatable views should be updatable by the system as well](https://www.studytonight.com/dbms/codd-rule.php) .
8. **Relational-Level Operations**: The system should support relational-level operations such as insert, delete, update at each level of relations. [Set operations like union, intersection, and minus should also be supported](https://www.studytonight.com/dbms/codd-rule.php) .
9. **Physical Data Independence** : The physical storage of data should not affect the system’s ability to access and manipulate it. [Renaming or moving files supporting tables should not impact applications](https://www.studytonight.com/dbms/codd-rule.php) .
10. **Logical Data Independence** : Changes in the logical structure (table structures) of the database should not affect users’ view of the data. [For example, if a table is split into two tables, a new view should give results as the join of those two tables](https://www.studytonight.com/dbms/codd-rule.php) .
11. **Integrity Independence** : The database should enforce its own integrity constraints rather than relying on external programs. [Key and check constraints, triggers, etc., should be stored in the data dictionary](https://www.studytonight.com/dbms/codd-rule.php) .

[These are just some of Codd’s rules that define the qualities required for a DBMS to become an RDBMS](https://www.studytonight.com/dbms/codd-rule.php) . It’s worth noting that while many commercial products follow some of these rules, very few adhere to all 13 rules defined by E.F. Codd.

I hope this clarifies Codd’s Rule for RDBMS! Let me know if you have any more questions.

5. What do you understand By Data Redundancy?

Ans: [**Data redundancy** refers to the existence of **duplicate or repetitive data** within a database or storage infrastructure](https://www.talend.com/resources/what-is-data-redundancy/) . [It occurs when the same piece of data is stored in two or more separate places](https://www.talend.com/resources/what-is-data-redundancy/) . [Data redundancy can be either **intentional** or **accidental**](https://www.talend.com/resources/what-is-data-redundancy/) .

[In a database, data redundancy can arise due to various reasons such as complex processes, inefficient coding, or accidental duplication during data entry](https://www.talend.com/resources/what-is-data-redundancy/) . [For example, if a retailer accidentally enters the same product twice in their database, it results in data redundancy](https://www.talend.com/resources/what-is-data-redundancy/) . [Similarly, if a customer’s name is entered multiple times in a file storage system due to multiple purchases, it also leads to redundant data](https://www.talend.com/resources/what-is-data-redundancy/) .

[While intentional data redundancy can be used to protect data and ensure consistency by leveraging multiple occurrences of data for disaster recovery and quality checks, accidental data redundancy can cause several issues](https://www.talend.com/resources/what-is-data-redundancy/) . [Duplicate entries of the same data can lead to inconsistency, confusion, and inefficiency in data management and analysis](https://www.talend.com/resources/what-is-data-redundancy/) .

To mitigate long-term inconsistency issues caused by accidental data redundancy, it is important to reduce and track redundant data efficiently. [One approach to prevent data redundancy is through **data replication**, which involves storing the same data in multiple locations](https://www.talend.com/resources/what-is-data-redundancy/) . [Data replication helps ensure consistency and provides an extra layer of protection for backup and disaster recovery purposes](https://www.talend.com/resources/what-is-data-redundancy/) .

It’s worth noting that while some organizations intentionally incorporate data redundancy into their daily operations for specific purposes, reducing accidental data redundancy is crucial for maintaining clean and reliable databases.

I hope this clarifies the concept of data redundancy! Let me know if you have any more questions.

6. What is DDL Interpreter?

Ans: [A **DDL Interpreter** is a component of a **Database Management System (DBMS)** that processes **Data Definition Language (DDL)** statements](https://www.edureka.co/community/29972/what-is-ddl-interpreter) . [DDL statements are used to create, modify, and delete database objects such as tables, indexes, and users](https://en.wikipedia.org/wiki/Data_definition_language) . [The DDL Interpreter processes these statements and records the generated statements in the table containing metadata](https://www.edureka.co/community/29972/what-is-ddl-interpreter) .

[The DDL Interpreter is responsible for creating and modifying the database schema, which defines the structure of the database](https://www.edureka.co/community/29972/what-is-ddl-interpreter) . [The schema includes information about tables, columns, data types, constraints, and other database objects](https://www.edureka.co/community/29972/what-is-ddl-interpreter) . [The DDL Interpreter processes the DDL statements into a set of tables containing metadata that describe the database schema](https://www.edureka.co/community/29972/what-is-ddl-interpreter) .

[It’s worth noting that DDL is one of the three types of SQL commands, along with **Data Manipulation Language (DML)** and **Data Query Language (DQL)**](https://www.edureka.co/community/29972/what-is-ddl-interpreter) . [While DDL is used to define the structure of the database, DML is used to manipulate data within the database, and DQL is used to retrieve data from the database](https://www.edureka.co/community/29972/what-is-ddl-interpreter) .

I hope this clarifies what a DDL Interpreter is! Let me know if you have any more questions.

7.What is DML Compiler in SQL?

Ans:  **DML Compiler** in SQL. [However, a **Data Manipulation Language (DML)** is a subset of SQL statements that alter the information stored in tables 1](https://en.wikipedia.org/wiki/Data_manipulation_language)[2](https://learnsql.com/blog/what-is-dql-ddl-dml-in-sql/). [DML commands are used to add, modify, and delete data in a database 3](https://www.javatpoint.com/dml-commands-in-sql). Some of the most common DML commands include:

* **SELECT**: Retrieves data from one or more tables.
* **INSERT**: Adds new data to a table.
* **UPDATE**: Modifies existing data in a table.
* **DELETE**: Removes data from a table.

I hope this helps! Let me know if you have any more questions.

8. What is SQL Key Constraints writing an Example of SQL Key Constraints

Ans: [**SQL Key Constraints** are used to specify rules for the data in a table and limit the type of data that can go into a table 1](https://www.w3schools.com/sql/sql_constraints.asp). Constraints ensure 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](https://www.w3schools.com/sql/sql_constraints.asp) [1](https://www.w3schools.com/sql/sql_constraints.asp).

[SQL Key Constraints can be column level or table level](https://www.w3schools.com/sql/sql_constraints.asp) [1](https://www.w3schools.com/sql/sql_constraints.asp). Here are some of the most commonly used SQL Key Constraints:

* [**NOT NULL**: Ensures that a column cannot have a NULL value](https://www.w3schools.com/sql/sql_constraints.asp) [1](https://www.w3schools.com/sql/sql_constraints.asp).
* [**UNIQUE**: Ensures that all values in a column are different](https://www.w3schools.com/sql/sql_constraints.asp) [1](https://www.w3schools.com/sql/sql_constraints.asp).
* **PRIMARY KEY**: A combination of a NOT NULL and UNIQUE. [Uniquely identifies each row in a table](https://www.w3schools.com/sql/sql_constraints.asp) [1](https://www.w3schools.com/sql/sql_constraints.asp).
* [**FOREIGN KEY**: Prevents actions that would destroy links between tables](https://www.w3schools.com/sql/sql_constraints.asp) [1](https://www.w3schools.com/sql/sql_constraints.asp).
* [**CHECK**: Ensures that the values in a column satisfy a specific condition](https://www.w3schools.com/sql/sql_constraints.asp) [1](https://www.w3schools.com/sql/sql_constraints.asp).
* [**DEFAULT**: Sets a default value for a column if no value is specified](https://www.w3schools.com/sql/sql_constraints.asp) [1](https://www.w3schools.com/sql/sql_constraints.asp).

Here are some examples of SQL Key Constraints:

* To create a table named Employees with columns EmployeeID, LastName, FirstName, and HireDate, where EmployeeID is the primary key, use the following SQL statement:
* CREATE TABLE Employees (
* EmployeeID INT PRIMARY KEY,
* LastName VARCHAR(50) NOT NULL,
* FirstName VARCHAR(50) NOT NULL,
* HireDate DATE DEFAULT GETDATE()
* );
* To add a foreign key constraint to an existing table named Orders with columns OrderID, CustomerID, and OrderDate, where CustomerID references the Customers table, use the following SQL statement:
* ALTER TABLE Orders
* ADD FOREIGN KEY (CustomerID)
* REFERENCES Customers(CustomerID);

I hope this helps! Let me know if you have any more questions.

9. What is save Point? How to create a save Point write a Query?

Ans : [A **savepoint** is a command in SQL that is used with the **rollback** command 1](https://www.javatpoint.com/savepoint-in-sql). [It is a command in Transaction Control Language (TCL) that is used to mark the transaction in a table](https://www.javatpoint.com/savepoint-in-sql) [2](https://www.tutorialandexample.com/save-point-in-sql). [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](https://www.javatpoint.com/savepoint-in-sql) [1](https://www.javatpoint.com/savepoint-in-sql). [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](https://www.javatpoint.com/savepoint-in-sql) [1](https://www.javatpoint.com/savepoint-in-sql). 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.

[To create a savepoint in SQL, we first need to initiate the transaction by using the BEGIN / START TRANSACTION command](https://www.javatpoint.com/savepoint-in-sql) [1](https://www.javatpoint.com/savepoint-in-sql). [We will save our initiated transaction using the SAVEPOINT command along with some specific names of this savepoint](https://www.javatpoint.com/savepoint-in-sql) [1](https://www.javatpoint.com/savepoint-in-sql). Here’s an example of how to create a savepoint:

BEGIN TRANSACTION;

SAVEPOINT my\_savepoint;

[Here, we have saved the initiated transaction with the name of ‘my\_savepoint’](https://www.javatpoint.com/savepoint-in-sql) [2](https://www.tutorialandexample.com/save-point-in-sql). Then, we can perform any number of operations on the database. [If we want to roll back to this savepoint later on, we can use the ROLLBACK command along with the name of our savepoint](https://www.javatpoint.com/savepoint-in-sql) [2](https://www.tutorialandexample.com/save-point-in-sql):

ROLLBACK TO my\_savepoint;

[This will undo all changes made after the savepoint was created and restore the database to its state at that point](https://www.javatpoint.com/savepoint-in-sql) [2](https://www.tutorialandexample.com/save-point-in-sql).

I hope this helps! Let me know if you have any more questions.

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

Ans: [In SQL, a **trigger** is a database object that automatically executes a set of SQL statements when a specific event occurs on a table 1](https://www.educba.com/triggers-in-sql/)[2](https://www.geeksforgeeks.org/sql-trigger-student-database/). [Triggers are used to enforce business rules, maintain data integrity, and automate certain actions within a database 2](https://www.geeksforgeeks.org/sql-trigger-student-database/).

[Triggers are associated with a specific table and are executed in response to events such as **INSERT**, **UPDATE**, or **DELETE** operations on that table 1](https://www.educba.com/triggers-in-sql/). [There are different types of triggers based on when they are executed, such as **AFTER INSERT**, **AFTER UPDATE**, **AFTER DELETE**, **BEFORE INSERT**, **BEFORE UPDATE**, and **BEFORE DELETE** triggers 3](https://www.javatpoint.com/trigger-in-sql).

To create a trigger in SQL, you can use the CREATE TRIGGER statement followed by the trigger name, event, and table name. Here’s an example of how to create an AFTER INSERT trigger:

CREATE TRIGGER my\_trigger

AFTER INSERT ON my\_table

FOR EACH ROW

BEGIN

-- SQL statements to be executed when the trigger fires

-- You can perform various actions or validations here

-- For example, you can update other tables, send notifications, etc.

END;

In this example, my\_trigger is the name of the trigger, my\_table is the table on which the trigger is created, and AFTER INSERT specifies that the trigger should be executed after an INSERT operation on my\_table. [The FOR EACH ROW clause indicates that the trigger should be fired for each row affected by the INSERT operation](https://www.educba.com/triggers-in-sql/) [4](https://www.sqlservertutorial.net/sql-server-triggers/sql-server-create-trigger/).

Within the BEGIN and END block, you can write the SQL statements that define the actions to be performed when the trigger fires. [These statements can include any valid SQL operations such as updating other tables, inserting data into other tables, or sending notifications](https://www.educba.com/triggers-in-sql/) [4](https://www.sqlservertutorial.net/sql-server-triggers/sql-server-create-trigger/).

It’s important to note that the exact syntax and capabilities of triggers may vary depending on the specific database management system (DBMS) you are using. Therefore, it’s recommended to refer to the documentation of your DBMS for more detailed information on creating triggers.

I hope this clarifies what a trigger is and how to create one in SQL! Let me know if you have any more questions.

#programming

11**. create a table student.**

1. CREATE TABLE student(roll\_num int AUTO\_INCREMENT,

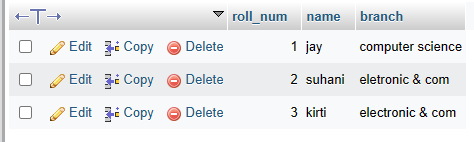
name varchar(30),

branch varchar(30),

PRIMARY KEY (roll\_num));

2. INSERT INTO student VALUES ('',"jay","computer science");

INSERT INTO student VALUES ('',"suhani","eletronic & com");



**12. Create Table Exam**

1. CREATE TABLE exam(s\_code int AUTO\_INCREMENT,

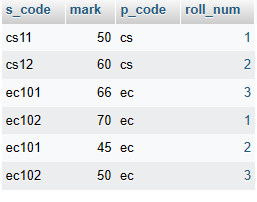
mark int,

p\_code varchar(30),

roll\_num int ,

FOREIGN KEY(roll\_num) REFERENCES student(roll\_num));

* INSERT INTO `exam` (`s\_code`, `mark`, `p\_code`, `roll\_num`) VALUES ('cs11', '50', 'cs', '1');
* INSERT INTO `exam` (`s\_code`, `mark`, `p\_code`, `roll\_num`) VALUES ('cs12', '60', 'cs', '2');
* INSERT INTO `exam` (`s\_code`, `mark`, `p\_code`, `roll\_num`) VALUES ('ec101', '66', 'ec', '3');
* INSERT INTO `exam` (`s\_code`, `mark`, `p\_code`, `roll\_num`) VALUES ('ec102', '70', 'ec', '1');
* INSERT INTO `exam` (`s\_code`, `mark`, `p\_code`, `roll\_num`) VALUES ('ec101', '45', 'ec', '2');
* INSERT INTO `exam` (`s\_code`, `mark`, `p\_code`, `roll\_num`) VALUES ('ec102', '50', 'ec', '3');



**13. create table people.**

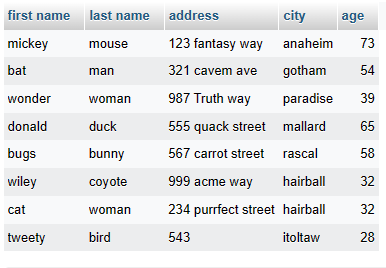
* CREATE TABLE people(first name varchar (20),

Last name varchar(20),

Address varchar (50),

city varchar(30),

age int);



**14. create table employe**

CREATE TABLE `employe` (`emp\_id` INT NOT NULL , `first name` VARCHAR(30) NOT NULL , `last name` VARCHAR(30) NOT NULL , `salary` INT NOT NULL , `joineing\_date` DATE NOT NULL , `department` VARCHAR(30));

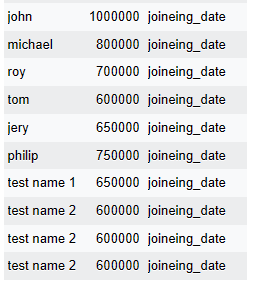
* INSERT INTO `employe` (`emp\_id`, `first name`, `last name`, `salary`, `joineing\_date`, `department`) VALUES ('1', 'john', 'abraham', '1000000', '2013-01-01', 'banking');
* INSERT INTO `employe` (`emp\_id`, `first name`, `last name`, `salary`, `joineing\_date`, `department`) VALUES ('2', 'michael', 'clarke', '800000', '2013-01-01', 'insurence');
* INSERT INTO `employe` (`emp\_id`, `first name`, `last name`, `salary`, `joineing\_date`, `department`) VALUES ('3', 'roy', 'thomas', '700000', '2013-01-01', 'banking');
* INSERT INTO `employe` (`emp\_id`, `first name`, `last name`, `salary`, `joineing\_date`, `department`) VALUES ('4', 'tom', 'jose', '600000', '2013-02-01', 'insurance');
* INSERT INTO `employe` (`emp\_id`, `first name`, `last name`, `salary`, `joineing\_date`, `department`) VALUES ('5', 'jery', 'pinto', '650000', '2013-02-01', 'insurence');
* INSERT INTO `employe` (`emp\_id`, `first name`, `last name`, `salary`, `joineing\_date`, `department`) VALUES ('6', 'philip', 'mathew', '750000', '2013-01-01', 'service');
* INSERT INTO `employe` (`emp\_id`, `first name`, `last name`, `salary`, `joineing\_date`, `department`) VALUES ('7', 'test name 1', '123’, '650000', '2013-01-01', service);
* INSERT INTO `employe` (`emp\_id`, `first name`, `last name`, `salary`, `joineing\_date`, `department`) VALUES ('8', 'test name 2', 'last name', '600000', '2013-02-01', 'insurence');



1.SELECT \* FROM `employe` WHERE 'first name' ='tom’;



2. SELECT `first name`, `salary`, 'joineing\_date' FROM `employe`;



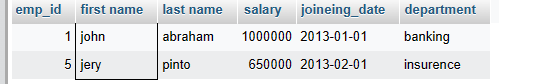
3. SELECT \* FROM `employe` ORDER BY `first name` asc;



4. SELECT \* FROM `employe` ORDER BY salary DESC;



5. SELECT \* FROM `employe` WHERE `first name` IN ('john','jery');



6. SELECT DEPARTMENT,MAX(SALARY) MAXSALARY FROM EMPLOYE GROUP BY DEPARTMENT ORDER BY MAXSALARY ASC;



7. SELECT `FIRST NAME`,INCENTIVE\_AMOUNT FROM EMPLOYE A INNER JOIN INCENTIVE B ON A.EMP\_ID=B.EMP\_REF\_ID AND INCENTIVE\_AMOUNT >3000;



**15. create table incentive**

CREATE TABLE incentive` (`emp\_ref\_id` INT NOT NULL , `incentive\_date` DATE NOT NULL , `incentive\_amount` INT NOT NULL ) ENGINE = InnoDB;

INSERT INTO `incentive` (`emp\_ref\_id`, `incentive\_date`, `incentive\_amount`) VALUES ('1', '2013-02-01', '5000'), ('2', '2013-02-01', '3000'), ('3', '2013-02-01', '4000'), ('4', '2013-01-01', '4500'), ('5', '2013-01-01', '3500');



**16. create table salesman**

CREATE TABLE`salesperson` (`sno` INT NOT NULL , `sname` VARCHAR(50) NOT NULL , `city` VARCHAR(30) NOT NULL , `comm` INT NOT NULL ) ENGINE = InnoDB;

Expand Requery Edit Bookmark Database : assinment Queried time : 17:12:14

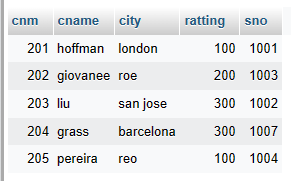
INSERT INTO `salesperson` (`sno`, `sname`, `city`, `comm`) VALUES ('1001', 'peel', 'london', '12'), ('1002', 'serres', 'san joes', '13'), ('1004', 'motika', 'london', '11'), ('1007', 'rafkin', 'barcelona', '15'), ('1003', 'axelord', 'new york', '1');



**17. create table customer.**

CREATE TABLE `customer` (`cnm` INT NOT NULL , `cname` VARCHAR(50) NOT NULL , `city` VARCHAR(30) NOT NULL , `ratting` INT NOT NULL , `sno` INT NOT NULL ) ENGINE = InnoDB;

INSERT INTO `customer` (`cnm`, `cname`, `city`, `ratting`, `sno`) VALUES ('201', 'hoffman', 'london', '100', '1001'), ('202', 'giovanee', 'roe', '200', '1003'), ('203', 'liu', 'san jose', '300', '1002'), ('204', 'grass', 'barcelona', '300', '1007'), ('205', 'pereira', 'reo', '100', '1004');



1. select \* from order where amount> 1000;

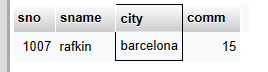
2. SELECT sname,city from salesperson where city='london' and comm>0.12;

3. All salespeople either in Barcelona or in London.

SELECT \* FROM salesperson WHERE city='barcelona';

SELECT \* FROM salesperson WHERE city=london;





4. SELECT \*FROM salesperson WHERE (comm > 10 AND comm< 12);



5. SELECT \*FROM customer WHERE ratting>100 OR city='rome';

