**DBMS-Lab-Manual**

**(Solved)**

**Lab: 01**

**Question: 01**

**What are the disadvantages of file processing system?**

**Answer:**

Disadvantages of File Processing System:

**Slow access time –**

Direct access of files is very difficult and one needs to know the entire

hierarchy of folders to get to a specific file. This involves a lot of time.

**Presence of redundant data –**

The same data can be present in two or more files which takes up more disc

space.

**Inconsistent Data –**

Due to data redundancy, same data stored at different places might not match

to each other.

**Data Integrity Problems –**

The data present in the database should be consistent and correct. To achieve

this, the data should must satisfy certain constraints.

**Difficulty in recovery of corrupt data –**

Recovery or backup of lost and corrupt data is nearly impossible in case of File

Processing System.

**Lack of Atomicity –**

Operations performed in the database must be atomic i.e. either the operation

takes place as a whole or does not take place at all.

**Unauthorized Access –**

Anyone who gets access to the file can read or modify the data.

**Lab: 02**

**Exercise**

**Question 1**

**Modern School of higher education has around 2000 students and three departments. The students belong to various departments in the school. Every department has a Head of Department along with teachers. The head of department manages his departmental teachers and the students belonging to his department. The head of department also teaches the students. All the head of departments report to the principal of the school. The principal manages the departments and also teaches the students. Handling the administrative staff of the school is also the responsibility of the principal. Every individual, except the students, are the employees of the school**

**Problem**

**● Identify the entity**

**● Identify the entity’s attribute**

**● Identify the primary and foreign Keys**

**● Identify relationship between Entities**

**● Identify the Cardinality constraints**

**Answer:**

**1. Entities:**

Students

Departments

Head of Departments

Teachers

Principal

Administrative Staff

**2. Entity Attributes:**

Students (Student\_ID, Name, Phone\_no, Address)

Departments (Department\_ID, Name, E-mail)

Head of Departments (Name, E-mail)

Teachers(Teacher\_ID, Name, Phone\_no, Experience)

Principal (Name, Phone\_no)

Administrative Staff (Name, Phone\_no)

**3. Primary Key:**

Student\_Id

Department\_ID

Teacher-ID

**4. Relationship between Entities:**

The relationship between Student and Department is Many to Many.

The relationship between the Department and Head of Department is Manyto

Many.

The relationship between the Head of Department and Teachers is One to

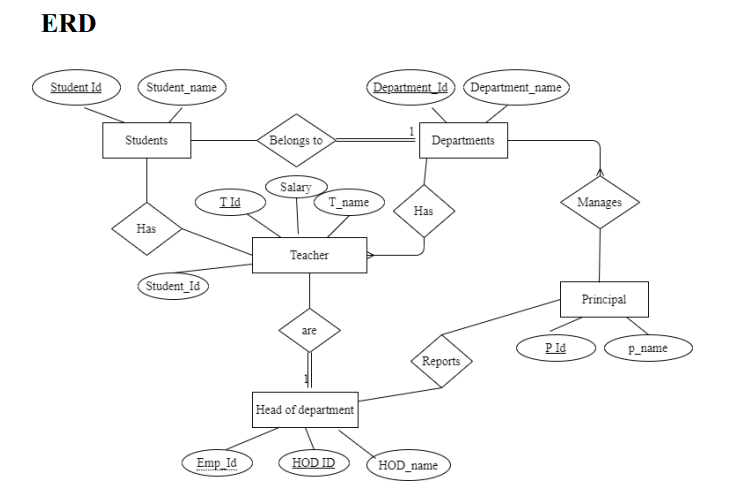
Many.

The relationship between the Head of Department and Students is One to

Many.

The relationship between the Principal and Department is One to Many.

The relationship between Administrative Staff and Principal is One to One.



**Question 2**

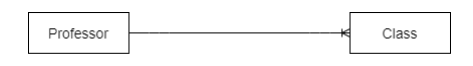
**1. a professor teaches zero, one or many classes and a class is taught by one professor**

**2. a course may generate zero, one or many classes and a class comes from one course**

**3. a class is held in one room but a room has many classes**

**Answer:**

1. A professor teaches zero, one or many classes and a class is taught by one professor



2. A course may generate zero, one or many classes and a class comes from one course



3. A class is held in one room but a room has many classes



**Lab: 03**

**Exercise**

**Question 1**

**Modern School of higher education has around 2000 students and three departments. The students belong to various departments in the school. Every department has a Head of Department along with teachers. The head of department manages his departmental teachers and the students belonging to his department. The head of department also teaches the students. All the head of departments report to the Principal of the school. The Principal manages the departments and also teaches the students. Handling the administrative staff of the school is also the responsibility of the Principal. Every individual, except the students, are the employees of the school**

* **Identify the entity**
* **Identify the entity’s attribute**
* **Identify the primary and foreign Keys**
* **Identify relationship between Entities**
* **Identify the Cardinality constraints**
* **Draw ERD**

**Answer:**

**1. Entities:**

• Students

• Departments

• Head of Departments

• Teachers

• Principal

• Administrative Staff

**2. Entity Attributes:**

• Students (Student\_ID, Name, Phone\_no, Address)

• Departments (Department\_ID, Name, E-mail)

• Head of Departments (Name, E-mail)

• Teachers (Teacher\_ID, Name, Phone\_no, Experience)

• Principal (Name, Phone\_no)

• Administrative Staff (Name, Phone\_no)

**3. Primary Key:**

• Student\_Id

• Department\_ID

• Teacher-ID

**4. Relationship between Entities:**

• The relationship between Student and Department is Many to Many.

• The relationship between the Department and Head of Department is

Manyto Many.

• The relationship between the Head of Department and Teachers is One

toMany.

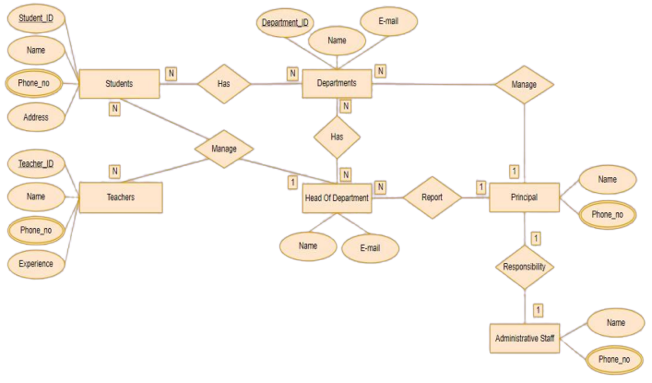
• The relationship between the Head of Department and Students is One

toMany.

• The relationship between the Principal and Department is One to Many.

• The relationship between Administrative Staff and Principal is One to One.

**5. ERD:**



**Question 2**

1. **an invoice is written by one sales rep but a sales rep writes many invoices**
2. **a vendor sells many products but a product is bought from one vendor**
3. **an invoice has one or many products and a product is found on zero, one or many invoices**

* **Identify the entity**
* **Identify the entity’s attribute**
* **Identify the primary and foreign Keys**
* **Identify relationship between Entities**
* **Identify the Cardinality constraints**
* **Draw ERD**

**Answer:**

**1. Entities:**

• Invoice

• Sales\_rep

• Vendor

• Product

**2. Entity Attributes:**

• Invoice (Invoice\_no, Invoice\_Product\_ID,Invoice\_Product\_Quantity,

Invoice\_Date, Customer\_Name, Customer\_Address)

• Sales\_rep (Sales\_rep\_ID, Sales\_rep\_Name, Sales\_rep\_Phone.no)

• Vendor (Vendor\_ID, Name, Address, Phone.no)

• Product (Product\_ID, Name, Sales\_rep\_ID, Vendor\_ID)

**3. Primary Key:**

• Invoice\_no

• Invoice\_Product\_ID

• Sales\_rep\_ID

• Vendor\_ID

• Product\_ID

Foreign Key:

• Sales\_rep\_ID

• Vendor\_ID

• Product\_ID

**4. Relationship between Entities:**

• The relationship between Invoice and Sales\_rep is One to One.

• The relationship between Sales\_rep and Invoices is One to Many.

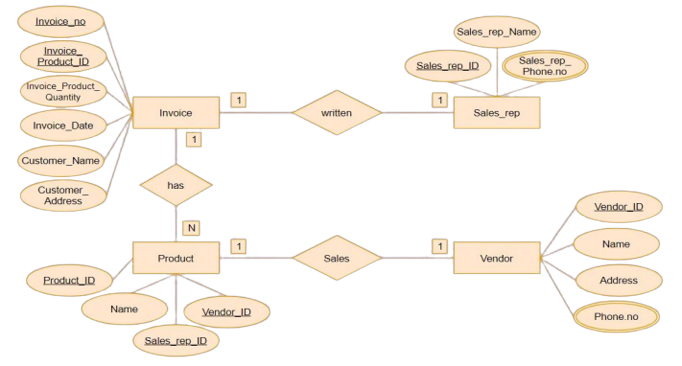
• The relationship between Vendor and Products is One to Many.

• The relationship between Product and Vendor is One to One.

• The relationship between Invoice and Products is One to Many.

• The relationship between Product and Invoices is One to Many.

**5. ERD:**



**Lab: 04**

**Exercise**

**CASE - STUDY -II**

**The Prescriptions-R-X chain of pharmacies has offered to give you a free lifetime supply of medicine if you design its database. Given the rising cost of health care, you agree. Here’s the information that you gather:**

**● Patients are identified by an SSN, and their names, addresses, and ages must be recorded.**

**● Doctors are identified by an SSN. For each doctor, the name, specialty, and years of experience must be recorded.**

**● Each pharmaceutical company is identified by name and has a phone number.**

**● For each drug, the trade name and formula must be recorded. Each drug is sold by a given pharmaceutical company, and the trade name identifies a drug uniquely from among the products of that company. If a pharmaceutical company is deleted, you need not keep track of its products any longer.**

**● Each pharmacy has a name, address, and phone number.**

**● Every patient has a primary physician. Every doctor has at least one patient.**

**● Each pharmacy sells several drugs and has a price for each. A drug could be sold at several pharmacies, and the price could vary from one pharmacy to another.**

**● Doctors prescribe drugs for patients. A doctor could prescribe one or more drugs for several patients, and a patient could obtain prescriptions from several doctors.**

**● Each prescription has a date and a quantity associated with it. You can assume that, if a doctor prescribes the same drug for the same patient more than once, only the last such prescription needs to be stored.**

**● Pharmaceutical companies have long-term contracts with pharmacies. A pharmaceutical company can contract with several pharmacies, and a pharmacy can contract with several pharmaceutical companies. For each contract, you have to store a start date, an end date, and the text of the contract.**

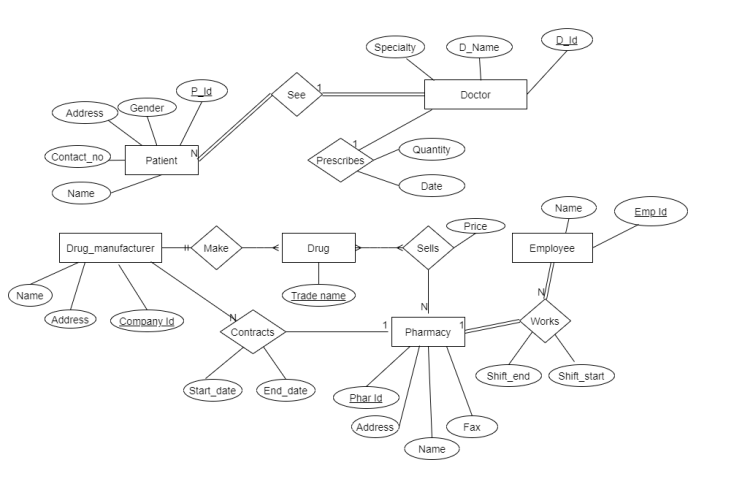
**● Pharmacies appoint a supervisor for each contract. There must always be a supervisor for each contract, but the contract supervisor can change over the lifetime of the contract.**

**Draw an ER diagram that captures the preceding information. Identify any constraints not captured by the ER diagram.**

**How would your design change if each drug must be sold at a fixed price by all pharmacies?**

**How would your design change if the design requirements change as follows: If a doctor prescribes the same drug for the same patient more than once, several such prescriptions may have to be stored?**

**ERD:**



**Lab: 05**

**Exercise**

**Consider the following schema:**

1. **Customer (cust\_id, cust\_name).**
2. **Product (prod\_code ,prod\_name, unit\_price)**
3. **Customer\_Order (order\_code, order\_date, cust\_id)**
4. **Order\_Item (order\_code, prod\_code, num\_of\_units)**
5. **Develop DDL of in SQL**
6. **Add Column contact in Customer Table**
7. **Add Column company\_name in Product Table**

**Query:**

CREATE DATABASE Lab\_05

CREATE TABLE Customer (

Customer\_id INT(10) NOT NULL,

Customer\_name CHAR(20),

PRIMARY KEY (Customer\_id)

);

CREATE TABLE Product (

Product\_code INT(10) NOT NULL,

Product\_name VARCHAR(20),

Unit\_price VARCHAR(10),

PRIMARY KEY (Product\_code)

);

CREATE TABLE Customer\_Order (

Order\_code INT(10) NOT NULL,

Order\_date DATE NOT NULL,

Customer\_id INT(10) NOT NULL,

PRIMARY KEY (Customer\_id, Order\_code)

);

CREATE TABLE Order\_Item (

Order\_code INT(10) NOT NULL,

Product\_code INT(10) NOT NULL,

Num\_of\_Units VARCHAR(20) NOT NULL,

PRIMARY KEY (Order\_code, Product\_code)

);

ALTER TABLE Customer

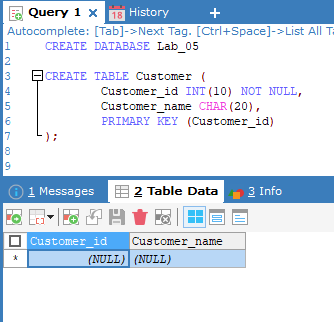
ADD Contact INT(11) NOT NULL;

ALTER TABLE Product

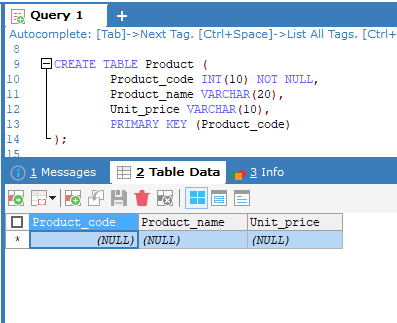
ADD Company\_Name VARCHAR(20) NOT NULL;

**Output Screenshots:**

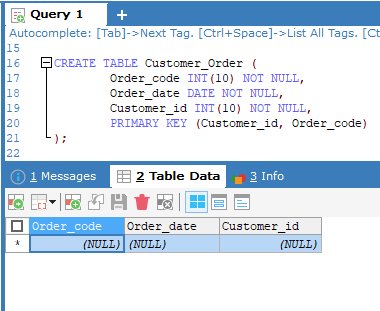
**Customer Table:**



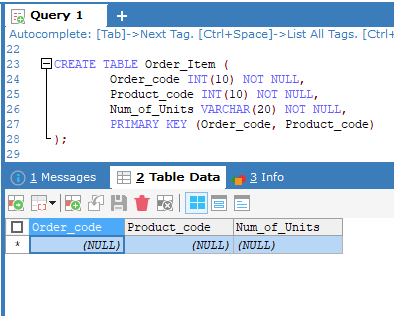
**Product Table:**



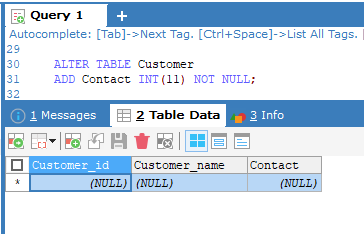
**Customer Order Table:**



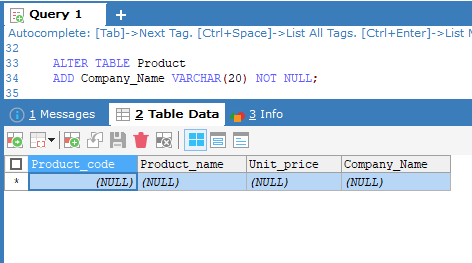
**Order Item Table:**



**Add Column Contact in Customer Table:**



**Add Column Company\_Name in Product Table:**



**Lab: 06**

**Exercise**

**Question 1**

**Insert the following data into the following table**

**Movie (id, title, year, director)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Title** | **Year** | **Director** |
| **1** | **Ben hur** | **2016** | **Thomas** |
| **2** | **Get Smart** | **2012** | **Richard Bell** |
| **3** | **Spider Man** | **2009** | **Tam Morry** |
| **4** | **Batman V/S Super Man** | **2015** | **Gerald Hond** |

**Query:**

CREATE DATABASE Lab\_06

CREATE TABLE Movie (

Id INT(10) NOT NULL,

title VARCHAR(20),

yearr INT(4) NOT NULL,

director VARCHAR(20)

);

INSERT INTO Movie (Id, title, yearr, director)

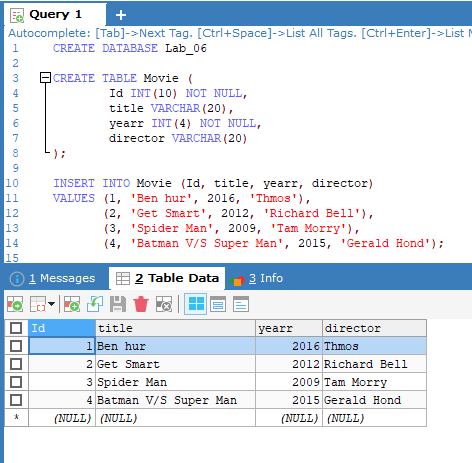
VALUES (1, 'Ben hur', 2016, 'Thmos'),

(2, 'Get Smart', 2012, 'Richard Bell'),

(3, 'Spider Man', 2009, 'Tam Morry'),

(4, 'Batman V/S Super Man', 2015, 'Gerald Hond');

**Output Screenshots:**



**Lab: 07**

**Exercise**

**Question 1**

**Grant all privilege to user which name Ali.**

**Output:**

GRANT ALL PREVILEGES ON \*.\* TO 'Ali' @ ' localhost' ;

**Question 2**

**Grant only creates permission to user which name Fahad.**

**Output:**

GRANT CREATE ON \*.\* TO 'Fahad' @ localhost';

**Lab: 08**

**Exercise**

|  |  |  |  |
| --- | --- | --- | --- |
| **Emp\_num** | **Emp\_Name** | **Emp\_Job** | **Emp\_Sal** |
| **E101** | **Salman** | **Analyst** | **6000** |
| **E102** | **Bushra** | **Programmer** | **5000** |
| **E103** | **Madiha** | **Web Designer** | **6000** |
| **E104** | **Batool** | **ERD designer** | **4000** |
| **E105** | **Hameed** | **Web Designer** | **3000** |
| **E106** | **Nini** | **Analyst** | **2500** |
| **E107** | **Imtiaz** | **Web Designer** | **6500** |
| **E108** | **Rashid** | **Programmer** | **4000** |
| **E109** | **Muzzamil** | **ERD designer** | **2000** |

1. **List all employees’ number, employee’s name and jobs from emp.**
2. **List all employees’ number, employee’s name and jobs from emp whose salaries greater than 5,000.**
3. **List all employees’ number, employee’s name and jobs from emp whose salaries less than 5,000.**
4. **List all employees’ number, employee’s name and jobs from emp whose salaries between 1,000 to 5,000.**
5. **List all employees’ number, employee’s name, jobs and salaries from emp.**

**Query:**

CREATE DATABASE Lab\_08

CREATE TABLE Employee (

Emp\_num VARCHAR(10) NOT NULL,

Emp\_Name CHAR(20),

Emp\_Job CHAR(20),

Emp\_Sal INT(10),

PRIMARY KEY (Emp\_num)

);

INSERT INTO Employee (Emp\_num, Emp\_Name, Emp\_Job, Emp\_Sal)

VALUES ('E101', 'Salman', 'Analyst', 6000),

('E102', 'Bushra', 'Programmer', 5000),

('E103', 'Madiha', 'Web Designer', 6000),

('E104', 'Batool', 'ERD designer', 4000),

('E105', 'Hameed', 'Web Designer', 3000),

('E106', 'Nini', 'Analyst', 2500),

('E107', 'Imtiaz', 'Web Designer', 6500),

('E108', 'Rashid', 'Programmer', 4000),

('E109', 'Muzzamil', 'ERD designer', 2000);

SELECT Emp\_num, Emp\_Name, Emp\_Job FROM Employee;

SELECT Emp\_num, Emp\_Name, Emp\_Job FROM Employee WHERE Emp\_Sal > 5000;

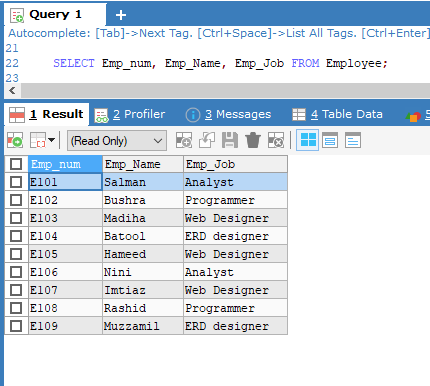
SELECT Emp\_num, Emp\_Name, Emp\_Job FROM Employee WHERE Emp\_Sal < 5000;

SELECT Emp\_num, Emp\_Name, Emp\_Job FROM Employee WHERE Emp\_Sal BETWEEN 1000 AND 5000;

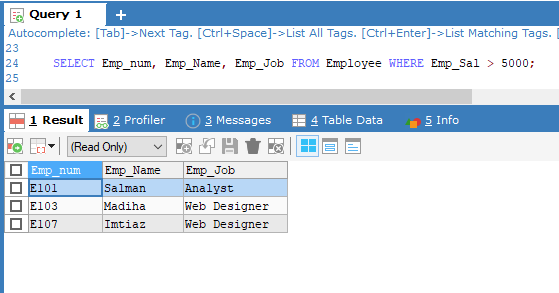
SELECT \* FROM Employee;

**Output Screenshots:**

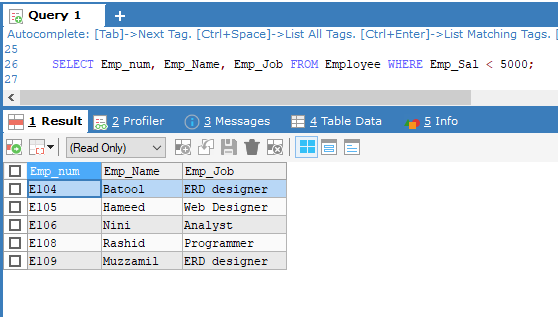
**Question: 01**



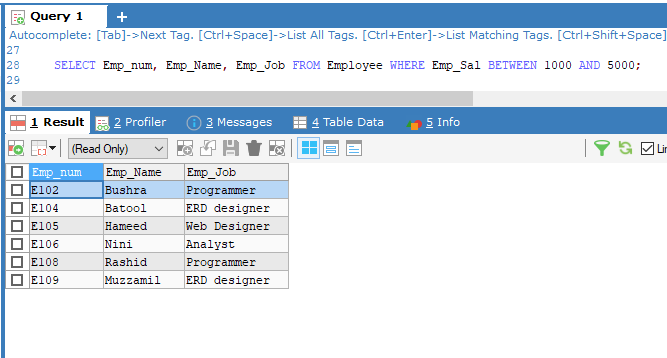
**Question: 02**



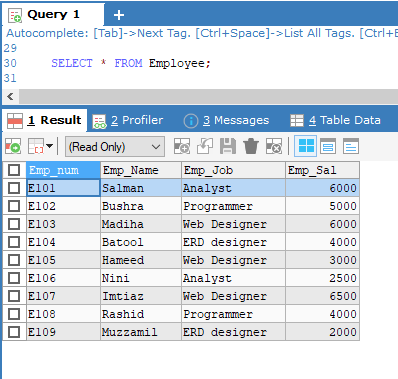
**Question: 03**



**Question: 04**



**Question: 05**



**Lab: 09**

**Exercise**

**Question No. 1**

**We have the following tables which creates statement are**

1. **create table supplier(S\_NO int identity(1,1), SUP\_ID varchar(50) primary key,SUP\_NAME varchar (50), SUP\_ADD varchar (50), SUP\_NIC int, rank int)**
2. **create table Product(Prod\_ID varchar(50) primary key,Prod\_Name varchar(50),[Price/Unit] float)**
3. **create table Orders(Ord\_ID varchar(50) primary key,Prod\_ID varchar(50),Qty int,Totalrate int,SUP\_ID varchar (50))**

**Write the Queries using Joins on above tables**

1. **List of all supplier that palced order**
2. **List of all Product that are supplied by supplier whose id is 101**
3. **Find all order(s) of product named Rice**

**Query:**

CREATE DATABASE Lab\_9

CREATE TABLE Supplier (

S\_NO INT(10),

SUP\_ID VARCHAR(50) PRIMARY KEY,

SUP\_NAME VARCHAR (50),

SUP\_ADD VARCHAR (50),

SUP\_NIC INT(10),

RANK INT(10)

);

INSERT INTO Supplier (S\_NO, SUP\_ID, SUP\_NAME, SUP\_ADD, SUP\_NIC, RANK)

VALUES (1, 101, 'John', 'Street 69, London', 0123456789, 9),

(2, 102, 'Alex', '14 Street, Scotland', 3456789012, 8),

(3, 103, 'Parker', 'L Street, London', 0237864591, 10),

(4, 104, 'Pele', 'Downtown, London', 9234567810, 8);

CREATE TABLE Product (

Prod\_ID VARCHAR(50) NOT NULL PRIMARY KEY,

Prod\_Name VARCHAR(50),

Price\_per\_Unit FLOAT

);

INSERT INTO Product (Prod\_ID, Prod\_Name, Price\_per\_Unit)

VALUE ('Prod\_1', 'Meat', 999.99),

('Prod\_2', 'Rice', 599.99),

('Prod\_3', 'Mix Fruits', 399.90),

('Prod\_4', 'Mix Vegitables', 449.99);

CREATE TABLE Orders (

Ord\_ID VARCHAR(50) NOT NULL PRIMARY KEY,

Prod\_ID VARCHAR(50) REFERENCES Product(Prod\_ID),

Qty INT,

Totalrate INT,

SUP\_ID VARCHAR(50) REFERENCES Supplier(SUP\_ID)

);

INSERT INTO Orders (Ord\_ID, Prod\_ID, Qty, Totalrate, SUP\_ID)

VALUES ('Ord\_01', 'Prod\_1', 3, 2999.97, 101),

('Ord\_02', 'Prod\_2', 5, 2999.95, 104),

('Ord\_03', 'Prod\_3', 2, 799.8, 102),

('ord\_04', 'Prod\_4', 8, 3999.92, 103);

SELECT Supplier.S\_NO, Supplier.SUP\_NAME, Supplier.SUP\_ADD, Supplier.SUP\_NIC,

Orders.Ord\_ID, Orders.Prod\_ID, Orders.Qty, Orders.Totalrate FROM Supplier

INNER JOIN Orders ON Supplier.SUP\_ID = Orders.SUP\_ID;

SELECT Orders.Prod\_ID, Supplier.SUP\_ID, Supplier.SUP\_NAME FROM Orders

INNER JOIN Supplier ON Orders.SUP\_ID = Supplier.SUP\_ID

WHERE Supplier.SUP\_ID LIKE 101;

SELECT Orders.Ord\_ID, Orders.Prod\_ID, Orders.Qty, Orders.Totalrate, Product.Prod\_ID,

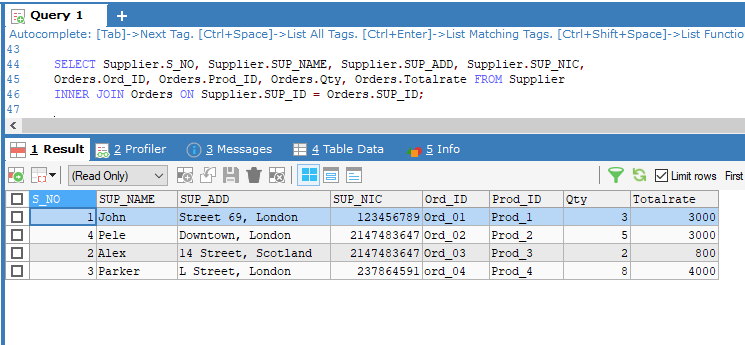
Product.Prod\_Name, Product.Price\_per\_Unit FROM Orders

INNER JOIN Product ON Orders.Prod\_ID = Product.Prod\_ID

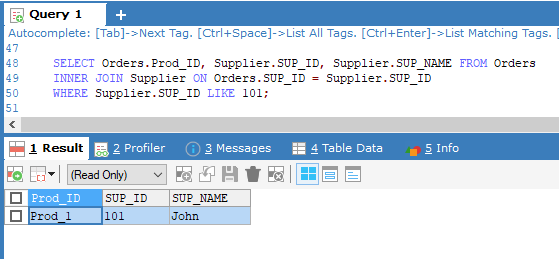
WHERE Prod\_Name = 'Rice';

**Output Screenshots:**

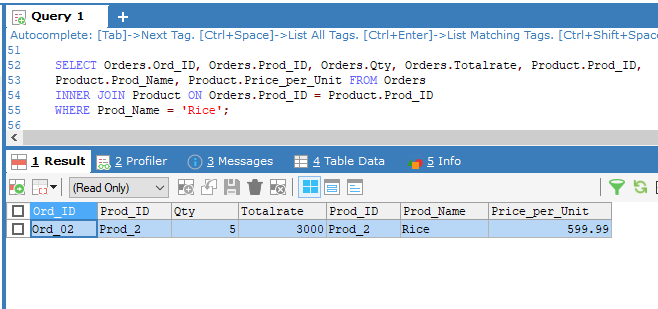
**Question: 01**



**Question: 02**



**Question: 03**



**Lab: 10**

**Exercise**

**Question No. 1**

1. **List average salary of each job.**
2. **Find average and sum of all the salaries of each job excluding clerks.**
3. **Find average and sum of the salaries of each job excluding salesmen', clerk' and 'manager'.**
4. **Find count, sum and average salaries of each job excluding salesmen', clerk' and 'manager'.**
5. **List average salary of each department.**

**Query:**

CREATE DATABASE Lab\_10

CREATE TABLE Employee (

Emp\_num VARCHAR(10) NOT NULL,

Emp\_Name CHAR(20),

Emp\_Job CHAR(20),

Emp\_Sal INT(10),

PRIMARY KEY (Emp\_num)

);

INSERT INTO Employee (Emp\_num, Emp\_Name, Emp\_Job, Emp\_Sal)

VALUES ('E101', 'Salman', 'Analyst', 6000),

('E102', 'Bushra', 'Programmer', 5000),

('E103', 'Madiha', 'Web Designer', 6000),

('E104', 'Batool', 'ERD designer', 4000),

('E105', 'Hameed', 'Web Designer', 3000),

('E106', 'Nini', 'Analyst', 2500),

('E107', 'Ajmal', 'Clerk', 2000),

('E108', 'Imtiaz', 'Web Designer', 6500),

('E109', 'Rashid', 'Programmer', 4000),

('E110', 'Haris', 'Manager', 9000),

('E111', 'Muzzamil', 'ERD designer', 2000),

('E112', 'Rashid', 'Salesmen', 5000);

SELECT Emp\_Job, AVG(Emp\_Sal) FROM Employee GROUP BY Emp\_Job;

SELECT Emp\_Job, AVG(Emp\_Sal), SUM(Emp\_Sal) FROM Employee

WHERE Emp\_Job <> 'Clerk' GROUP BY Emp\_Job;

SELECT Emp\_Job, AVG(Emp\_Sal), SUM(Emp\_Sal) FROM Employee

WHERE Emp\_Job <> 'Salesmen' AND Emp\_Job <> 'Clerk' AND Emp\_Job <> 'Manager'

GROUP BY Emp\_Job;

SELECT COUNT(\*), Emp\_Job, SUM(Emp\_Sal), AVG(Emp\_Sal) FROM Employee

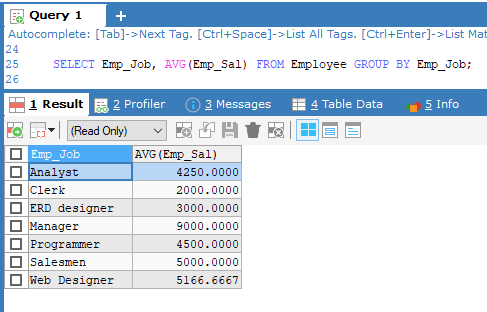
WHERE Emp\_Job <> 'Salesmen' AND Emp\_Job <> 'Clerk' AND Emp\_Job <> 'Manager'

GROUP BY Emp\_Job;

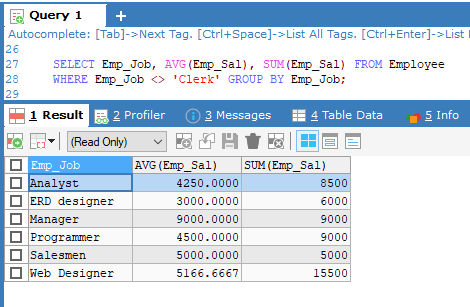
SELECT Emp\_Job, AVG(Emp\_Sal) FROM Employee GROUP BY Emp\_Job

**Output Screenshots:**

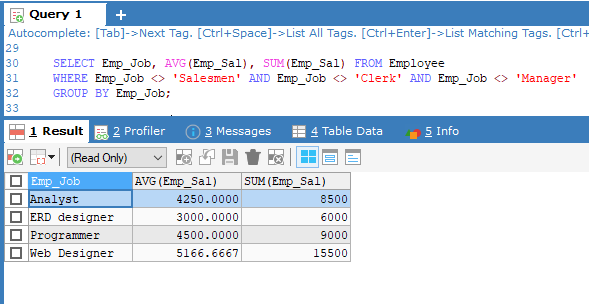
**Question: 01**



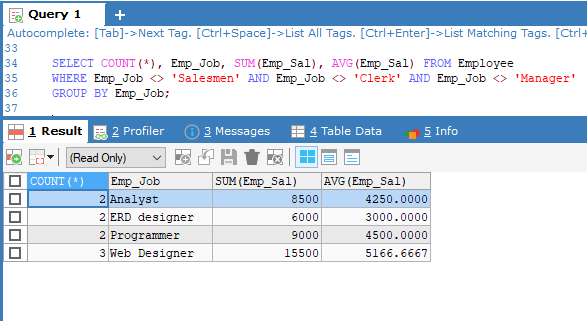
**Question: 02**



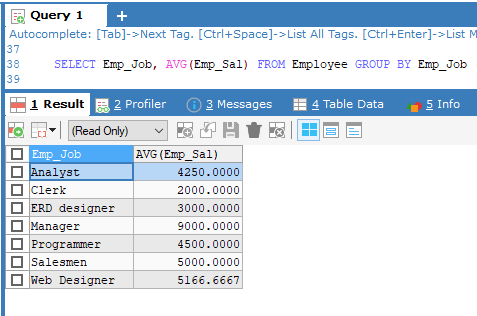
**Question: 03**



**Question: 04**



**Question: 05**



**Lab: 11**

**Exercise**

**Question No. 1**

1. **Find the names of Top 10 employees which salaries are highest.**

**Query:**

CREATE DATABASE Lab\_11

CREATE TABLE Employee (

Emp\_num VARCHAR(10) NOT NULL,

Emp\_Name CHAR(20),

Emp\_Job CHAR(20),

Emp\_Sal INT(10),

PRIMARY KEY (Emp\_num)

);

INSERT INTO Employee (Emp\_num, Emp\_Name, Emp\_Job, Emp\_Sal)

VALUES ('E101', 'Salman', 'Analyst', 6000),

('E102', 'Bushra', 'Programmer', 5000),

('E103', 'Madiha', 'Web Designer', 6000),

('E104', 'Batool', 'ERD designer', 4000),

('E105', 'Hameed', 'Web Designer', 3000),

('E106', 'Nini', 'Analyst', 2500),

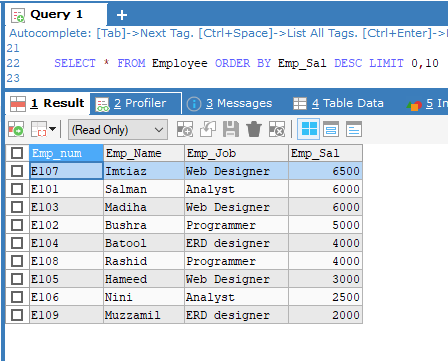
('E107', 'Imtiaz', 'Web Designer', 6500),

('E108', 'Rashid', 'Programmer', 4000),

('E109', 'Muzzamil', 'ERD designer', 2000);

SELECT \* FROM Employee ORDER BY Emp\_Sal DESC LIMIT 0,10

**Output Screenshots:**



**Lab: 12**

**Exercise**

**1-Creates a stored procedure named "SelectAllCustomers" that selects all records from the "Customers" table.**

**2-creates a stored procedure that selects Customers from a particular City from the "Customers" table**

**3- Creates a stored procedure that selects Customers from a particular City with a particular PostalCode from the "Customers" table:**

**Query:**

CREATE DATABASE Lab\_12

CREATE TABLE Customers (

customer\_ID INT(10) NOT NULL PRIMARY KEY,

fName CHAR(20),

lName CHAR(20),

address VARCHAR(50),

city CHAR(20),

postalCode VARCHAR(20),

country CHAR(20)

);

INSERT INTO Customers(customer\_ID, fName, lName, address, city, postalCode, country)

VALUES (1, 'Alfreds', 'Maria', 'Obere Str. 57', 'Berlin', '12209', 'Germany'),

(2, 'Ana', 'Trujillo', 'Avda. de la Constitución 2222', 'México D.F.', '05021', 'Mexico'),

(3, 'Antonio', 'Moreno', 'Mataderos 2312', 'México D.F.', '05023', 'Mexico'),

(4, 'Thomas', 'Hardy', '120 HanoverSq.', 'London', 'WA1 1DP', 'UK'),

(5, 'Berglunds', 'Christina', 'Berguvsvägen 8', 'Luleå', 'S-958 22', 'Sweden');

DELIMITER&&

CREATE PROCEDURE SelectAllCustomers()

BEGIN

SELECT \* FROM Customers;

END&&

DELIMITER;

CALL SelectAllCustomers();

DELIMITER&&

CREATE PROCEDURE CustomersFromParticularCity(IN particularCity CHAR(20))

BEGIN

SELECT fName, city FROM Customers WHERE city = particularCity;

END&&

DELIMITER;

CALL CustomersFromParticularCity('México D.F.');

DELIMITER&&

CREATE PROCEDURE CustomersFromParticularCityWithParticularPostalCode(

IN particularCity CHAR(20), particularPostalCode VARCHAR(20))

BEGIN

SELECT fName, city, postalCode FROM Customers

WHERE city = particularCity AND postalCode = particularPostalCode;

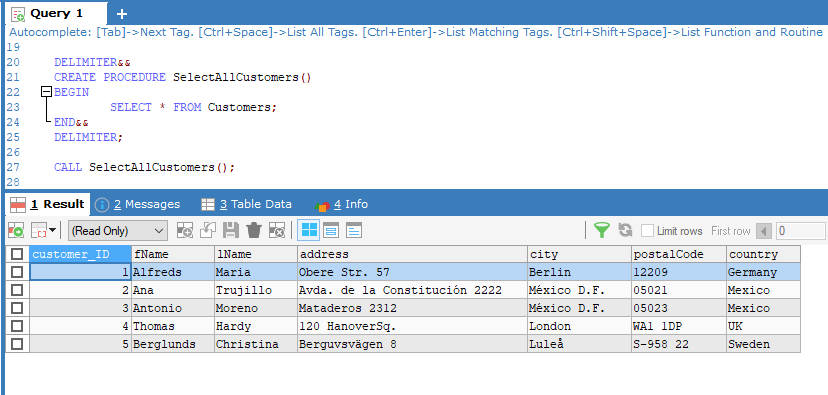
END&&

DELIMITER;

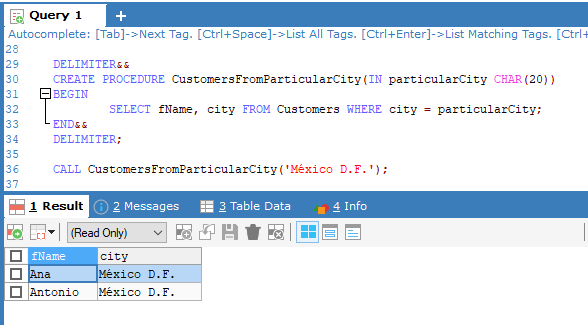
CALL CustomersFromParticularCityWithParticularPostalCode('México D.F.', '05023');

**Output Screenshots:**

**Question: 01**



**Question: 02**



**Question: 03**

