Problem Statement-

In a Superstore management system, there are various entities and their respective attributes. Let's break down the information:

1. Shops are a fundamental part of the system, and each shop is uniquely identified by a Shop_id. They have attributes such as Shop_Address, Shop_Phone, Shop_Email, and Manager_Name.

2. Employees are associated with shops, and each employee is identified by an Emp_id. Employee attributes include Emp_Name, Emp_Address, Job_Title, Joined_Date, Phone, and Salary.

3. An employee is exclusively affiliated with one shop. Customers interact with the shops, and each customer is identified by a Customer_Id. Customer details encompass Customer_Name, Customer_Address, and Phone.

4. A customer can visit only one shop at a given time but may choose to receive services from multiple employees within that shop.

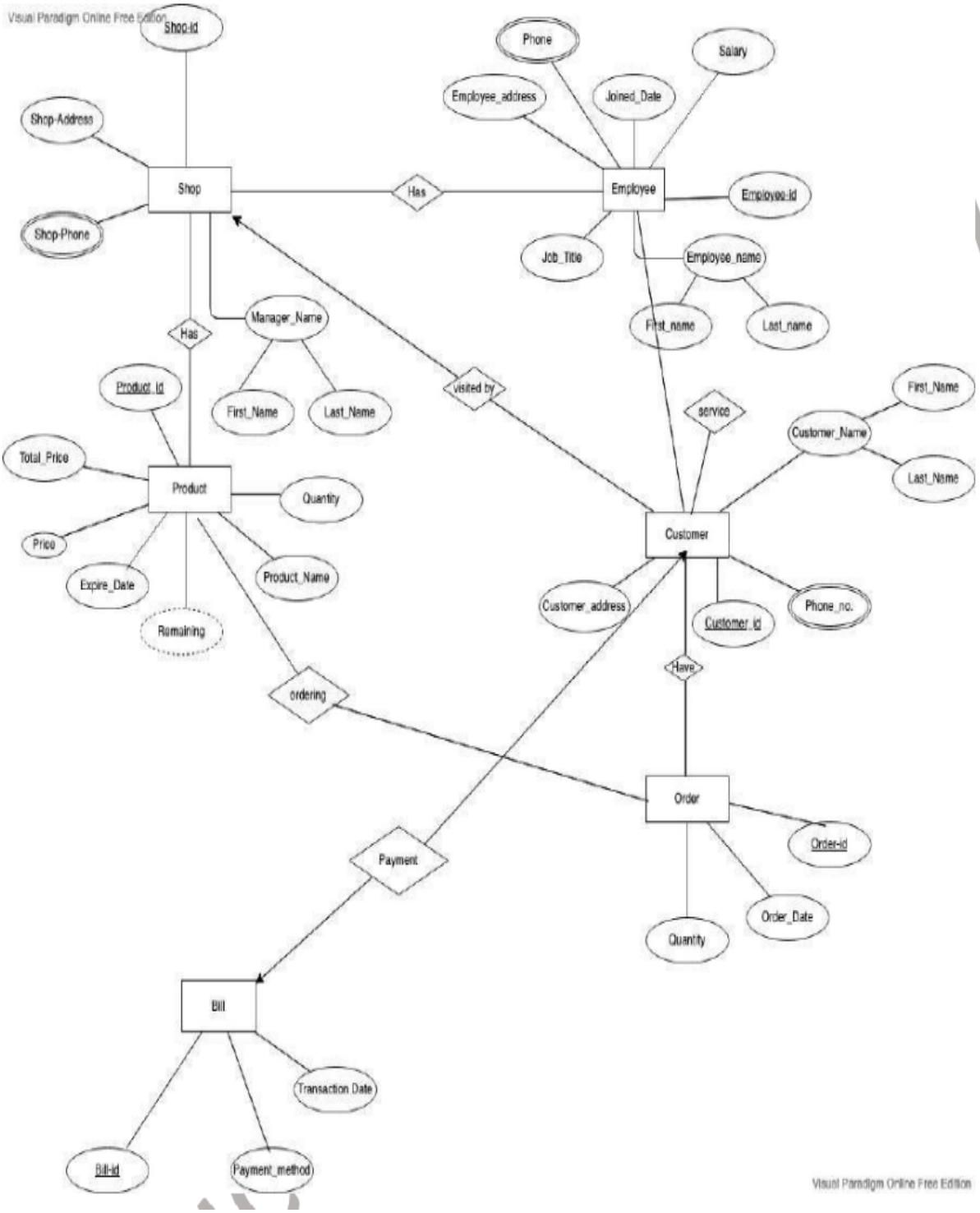
5.Products are available for sale in the shops, and each product is recognized by a Product_id. Product attributes include Product_Name, Expiry_Date, and Quantity. The products are part of the inventory.

6. Orders are placed by customers and are identified by an Order_id. Orders consist of Order_Date and Quantity. Multiple customers can place identical orders, and multiple orders can be placed by a single customer.

7. Bills are generated for transactions and are identified by Bill_id. They include attributes like Total_Bill, Payment_Method, Transaction_Date, Last_Transaction_Value, and Discount.

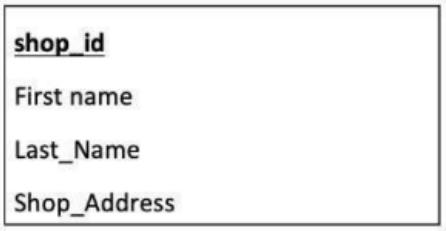
In summary, this Superstore management system involves shops, employees, customers, products, orders, and bills. Each entity has specific attributes and relationships within the system, facilitating the management of shop operations and customer interactions.

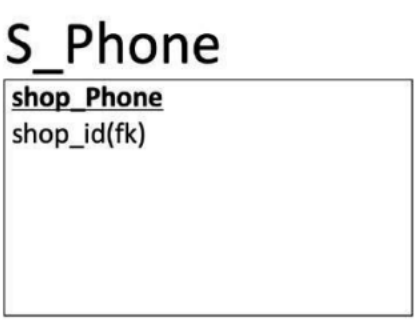
ER-Model

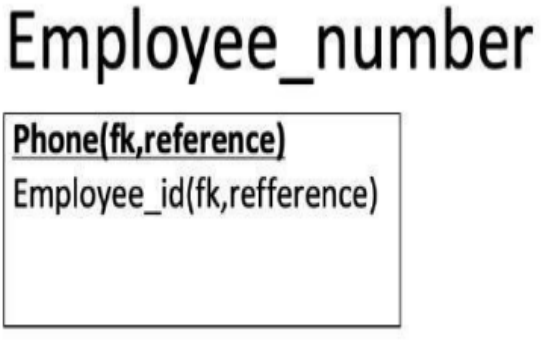

Visual Paradigm Online Free Edition
Shop-Id
Shop-Address
Shop-Phone
Shop
Has
Employee
Phone
Salary
Employee_address
Joined_Date
Employee-Id
Job_Title
Employee_name
First_name
Last_name
Has
Manager_Name
First_Name
Last_Name
Product_Id
Total_Price
Product
Quantity
Price
Expire_Date
Remaining
Product_Name
visited by
service
Customer_Name
First_Name
Last_Name
Customer
Customer_address
Customer_id
Phone_no.
Have
ordering
Order
Order-id
Order_Date
Quantity
Payment
Bill
Transaction Date
Bill-id
Payment_method
Visual Paradigm Online Free Edition

ER Diagram To Relational Model

Shop is an entity set having multivalued attributes so it will require 2 tables in the relational model. One table will contain all the simple attributes with the primary key and another table will contain the primary key and all the multivalued attributes. They are having many to many and one to many relations respectively. In many to many relation, we will make three tables and in one to many relation we make 2 tables. The one on the side of many relation will have the primary key of the other as one of the attributes. If any entity set is having one to one relation then there will be two tables and any of two will have primary key of the other. These rules are applied on the entity set of sets of our database which included Employee,Customer, Order, Bill and Product.







Customer

Customer_id
Shop_id(fk)
First_Name
Last_Name
Customer_address

Service

Employee_id(fk,pk)
Shop_id (fk,pk)

Orders

Orders_id
Orders_date
Quantity

CustomerPhone

Phone_no
Customer_id(fk)

Product

Product_id
Expire-data
Product_Quantity
Product_Name
Price
Total_Price
Shop_id(fk)

Bill

Bill_id
Payment_Method
Customer_id(fk)
Transaction_Date
Total_Bill

Functional Dependencies and Normalisation

A. **Shop**(Shop_id, First_Name,Last_Name, Shop_Address) Shop_id is the primary key and the only candidate key. It will identify all other attributes and there is no other functional dependency except this.
B. S_Phone(Shop_Phone, Shop_id) Shop_Phone is the primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this.
C. Employee(Employee_id,shop_id,First_Name,Last_Name,Sh op_Address,Salary) Employee_Id is primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this.
D. Employee_ number (phone,Employee_id). Phone is the primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this
E. Service(Employee_id, Customer_Id) Employee_Id and Customer_Id is primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this.
F. Customer(Customer_id,shop_id,First_Name,Last_Name,Custom er_address) Customer_Id is the primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this.
G. Customerphone(phone_no,Customer_id) Phone _nois the primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this.
H. Have(Order_Id, Customer_Id) Order_Id and Customer_Id is primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this.
I. Orders(Orders_id,orders_date,Quantity) Order_Id is the primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this.
J. Ordering(Order_Id, Product_Id) Order_Id and Product_Id is the primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this.

K. Product(Product_id , Expire_date date, shop_id, Product_Quantity , Product_Name , Price) Product_id is the primary key so it will identify all other attributes. Product_Name can identify Price.

So, there are 2 functional dependencies one from prime attribute (product_id) to all other remaining attributes and other from non-prime attribute(product_name) to non-prime attribute price. There is a transitive dependency present. Hence, it's not in third normal form but there is no partial dependency and hence, it's second normal form. To convert into the third normal form, we will make a separate table for functional dependencies.
1) Product(Product_id , Expire_date date, shop_id, Product_Quantity , Product_Name) Product_Id is the primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this. Hence, it's reduced to BCNF.
2) Product_price (Product_Name , Price) Product_Name primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this. Hence, it's reduced to BCNF. L) Bill(Bill_id,Payment_method,Transaction_Date,Customer_id) Bill_id is the primary key and the only candidate key. So, it will identify all other attributes and there is no other functional dependency except this. Hence, it's already normalised till BCNF.

MySql

-- Create Tables

```
CREATE TABLE shop (
    shop_id VARCHAR(8) PRIMARY KEY,
    First_Name VARCHAR(12) NOT NULL,
    Last_Name VARCHAR(10) NOT NULL,
    Shop_Address VARCHAR(40) NOT NULL);

CREATE TABLE Employee (
```

```
    Employee_id VARCHAR(8) PRIMARY KEY,
    shop_id VARCHAR(8) NOT NULL,
    First_Name VARCHAR(12) NOT NULL,
    Last_Name VARCHAR(10) NOT NULL,
    Shop_Address VARCHAR(40) NOT NULL,
    Salary INT CHECK (Salary >= 5000),
    FOREIGN KEY (shop_id) REFERENCES shop (shop_id) ON
DELETE CASCADE
);
```

```
CREATE TABLE S_Phone (
    shop_Phone VARCHAR(10) PRIMARY KEY,
    shop_id VARCHAR(8) NOT NULL,
    FOREIGN KEY (shop_id) REFERENCES shop (shop_id) ON
DELETE CASCADE
);
```

```
CREATE TABLE Employee_number (
    phone VARCHAR(10) PRIMARY KEY,
    Employee_id VARCHAR(8) NOT NULL,
    FOREIGN KEY (Employee_id) REFERENCES Employee
(Employee_id) ON DELETE CASCADE
);
```

```
CREATE TABLE Service (
    Employee_id VARCHAR(8) NOT NULL,
    shop_id VARCHAR(8) NOT NULL,
    PRIMARY KEY (Employee_id, shop_id),
    FOREIGN KEY (Employee_id) REFERENCES Employee
(Employee_id),
    FOREIGN KEY (shop_id) REFERENCES shop (shop_id)
);
```

```
CREATE TABLE Customer (
    Customer_id VARCHAR(8) PRIMARY KEY,
    shop_id VARCHAR(8) NOT NULL,
    First_Name VARCHAR(12) NOT NULL,
```

```
    Last_Name VARCHAR(10) NOT NULL,
    Customer_address VARCHAR(40) NOT NULL,
    FOREIGN KEY (shop_id) REFERENCES shop (shop_id) ON
DELETE CASCADE
);

CREATE TABLE Customerphone (
    phone_no VARCHAR(10) PRIMARY KEY,
    Customer_id VARCHAR(8) NOT NULL,
    FOREIGN KEY (Customer_id) REFERENCES Customer
(Customer_id) ON DELETE CASCADE
);

CREATE TABLE orders (
    orders_id VARCHAR(10) PRIMARY KEY,
    orders_date DATE,
    Quantity INT CHECK (Quantity >= 0)
);

CREATE TABLE Product_price (
    Product_Name VARCHAR(12) PRIMARY KEY,
    Price INT CHECK (Price >= 0)
);

CREATE TABLE Product (
    Product_id VARCHAR(10) PRIMARY KEY,
    Expire_date DATE,
    shop_id VARCHAR(8) NOT NULL,
    Product_Quantity INT CHECK (Product_Quantity >= 0),
    Product_Name VARCHAR(12) NOT NULL,
    FOREIGN KEY (Product_Name) REFERENCES Product_price
(Product_Name),
    FOREIGN KEY (shop_id) REFERENCES shop (shop_id) ON
DELETE CASCADE
);

CREATE TABLE Bill (
```

```
    bill_id VARCHAR(10) PRIMARY KEY,
    Payment_Method VARCHAR(8) NOT NULL,
    Customer_id VARCHAR(8) NOT NULL,
    Transaction_Date DATE,
    Total_Bill INT CHECK (Total_Bill > 0),
    FOREIGN KEY (Customer_id) REFERENCES Customer
(Customer_id) ON DELETE CASCADE
);
```

-- Insert Values

-- Insert values in shop

```
INSERT INTO shop (shop_id, First_Name, Last_Name, Shop_Address)
VALUES
    ('shop0001', 'Raman', 'Singh', '311-A CHAND NAGAR NEW DELHI'),
    -- (Other shop records here)
```

-- Insert values in Employee

```
INSERT INTO Employee (Employee_id, shop_id, First_Name,
Last_Name, Shop_Address, Salary) VALUES
    ('Emp1', 'shop0001', 'Ram', 'Aggarwal', '1-A CHAND NAGAR', 7000),
    -- (Other employee records here)
```

-- Insert values in Customer

```
INSERT INTO Customer (Customer_id, shop_id, First_Name,
Last_Name, Customer_address) VALUES
    ('Cus1', 'shop0001', 'Raju', 'Aggarwal', '322 CHAND NAGAR NEW
DELHI'),
    -- (Other customer records here)
```

-- Insert values in S_Phone

```
INSERT INTO S_Phone (shop_Phone, shop_id) VALUES
    ('9811223306', 'shop0001'),
    -- (Other shop phone records here)
```

-- Insert values in Employee_number

```
INSERT INTO Employee_number (phone, Employee_id) VALUES
    ('9211423305', 'Emp1'),
    -- (Other employee phone records here)

-- Insert values in Service
INSERT INTO Service (Employee_id, shop_id) VALUES
    ('Emp1', 'shop0001'),
    -- (Other service records here)

-- Insert values in Orders
INSERT INTO orders (orders_id, orders_date, Quantity) VALUES
    ('orders001', '2022-10-23', 3),
    -- (Other orders records here)

-- Insert values in Product_price
INSERT INTO Product_price (Product_Name, Price) VALUES
    ('PARLA BISCUT', 10),
    -- (Other product price records here)

-- Insert values in Bill
INSERT INTO Bill (bill_id, Payment_Method, Customer_id,
Transaction_Date) VALUES
    ('bill001', 'Cash', 'Cus1', '2022-10-23'),
    -- (Other bill records here)


-- Create and Use View
CREATE VIEW  AS
SELECT Employee_id FROM Employee WHERE Salary >= 6000;
```

```
mysql> create view veerji as
    -> select employee_id from employee where (salary>=6000);
Query OK, 0 rows affected (0.03 sec)

mysql> select * from veerji;
+-------------+
| employee_id |
+-------------+
| emp01       |
| emp02       |
| emp03       |
| emp04       |
| emp05       |
| emp06       |
| emp07       |
| emp08       |
| emp09       |
| emp10       |
+-------------+
10 rows in set (0.00 sec)

mysql>
```

-- Alter Table

-- SELECT * FROM Customer;

ALTER TABLE Customer DROP COLUMN Last_Name;

-- SELECT * FROM Customer;

```
Command Prompt - mysql -u
+-------------+---------+------------+-----------+------------------+
10 rows in set (0.00 sec)

mysql> alter table customer
    -> drop column_name last_name;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
 for the right syntax to use near 'last_name' at line 2
mysql> alter table customer
    -> drop column last_name;
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> select * from customer;
+-------------+---------+------------+------------------+
| customer_id | shop_id | first_name | customer_address |
+-------------+---------+------------+------------------+
| cus1        | shop01  | raju       | Nawada           |
| cus10       | shop10  | ankita     | Nanital          |
| cus2        | shop02  | amit       | KC Colony        |
| cus3        | shop03  | bunty      | Delhi            |
| cus4        | shop04  | roshan     | Kolkata          |
| cus5        | shop05  | anish      | Punjab           |
| cus6        | shop06  | daksh      | Rajasthan        |
| cus7        | shop07  | eshwar     | Shimla           |
| cus8        | shop08  | sanjeeta   | Bangalore        |
| cus9        | shop09  | sunita     | Haryana          |
+-------------+---------+------------+------------------+
10 rows in set (0.00 sec)

mysql>
```

-- Delete Records

-- SELECT * FROM Customer;

DELETE FROM Customer WHERE Customer_id = 'Cus4';

-- SELECT * FROM Customer;

```
mysql> delete from customer where customer_id='cus4';
Query OK, 1 row affected (0.03 sec)

mysql> select * from customer;
+-------------+---------+------------+------------------+
| customer_id | shop_id | first_name | customer_address |
+-------------+---------+------------+------------------+
| cus1        | shop01  | raju       | Nawada           |
| cus10       | shop10  | ankita     | Nanital          |
| cus2        | shop02  | amit       | KC Colony        |
| cus3        | shop03  | bunty      | Delhi            |
| cus5        | shop05  | anish      | Punjab           |
| cus6        | shop06  | daksh      | Rajasthan        |
| cus7        | shop07  | eshwar     | Shimla           |
| cus8        | shop08  | sanjeeta   | Bangalore        |
| cus9        | shop09  | sunita     | Haryana          |
+-------------+---------+------------+------------------+
9 rows in set (0.00 sec)
```

-- Update Records

-- SELECT * FROM Bill;

UPDATE Bill SET Payment_Method = 'Paytm' WHERE bill_id = 'bill003';

```
mysql> update bill
    -> set payment_method='paytm'
    -> where bill_id='bill3';
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from bill;
+---------+----------------+-------------+------------------+------------+
| bill_id | payment_method | customer_id | transaction_date | total_bill |
+---------+----------------+-------------+------------------+------------+
| bill1   | cash           | cus1        | 2023-10-23       |       NULL |
| bill10  | cash           | cus10       | 2023-10-18       |       NULL |
| bill2   | paytm          | cus2        | 2023-10-20       |       NULL |
| bill3   | paytm          | cus3        | 2023-10-21       |       NULL |
| bill5   | paytm          | cus5        | 2023-10-03       |       NULL |
| bill6   | cash           | cus6        | 2023-10-09       |       NULL |
| bill7   | cash           | cus7        | 2023-09-23       |       NULL |
| bill8   | cash           | cus8        | 2023-07-23       |       NULL |
| bill9   | cash           | cus9        | 2023-08-23       |       NULL |
+---------+----------------+-------------+------------------+------------+
9 rows in set (0.00 sec)
```

Python MySql Connectivity

Backend

```python
print("--SUPER STORE MANAGEMENT SYSTEM--")
print("Menu")
print("1. Press 1 to show details of products available.")
print("2. Press 2 to show details of Shop_id and their corresponding managers.")
print("3. Press 3 to show the average salary provided to the employees.")
print("4. Press 4 to show the number of employees working in our organization.")
print("5. Press 5 to show the ID of the employee getting the maximum salary.")
print("6. Press 6 to show the number of customers associated with us.")
print("7. Press 7 to show the details of the employees getting a salary ranging from 5000-8000.")
print("8. Press 8 to show the price of the product with the maximum price.")
print("9. Press 9 to show the details of the products between the price range 10-20.")
print("10. Press 10 to show details (like address, manager_name) of all branches of our organization.")

choice = int(input("Enter your choice: "))

if choice == 1:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
    query = "SELECT * FROM Product_price"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))

if choice == 2:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
    query = "SELECT shop_id, First_Name, Last_Name FROM shop"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))

if choice == 3:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
    query = "SELECT AVG(Salary) FROM Employee"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))

if choice == 4:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
```

```
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
    query = "SELECT COUNT(Employee_id) FROM Employee"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))

if choice == 5:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
    query = "SELECT Employee_id FROM Employee WHERE Salary=(SELECT MAX(Salary) FROM Employee)"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))

if choice == 6:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
    query = "SELECT COUNT(Customer_id) FROM Customer"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))

if choice == 7:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
    query = "SELECT * FROM Employee WHERE Salary BETWEEN 5000 AND 8000"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))

if choice == 8:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
    query = "SELECT Price FROM Product_price WHERE Price=(SELECT MAX(Price) FROM Product_price)"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))

if choice == 9:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
```

```
    query = "SELECT * FROM Product_price WHERE Price BETWEEN 10 AND 20"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))

if choice == 10:
    import pandas as pd
    from tabulate import tabulate
    import mysql.connector as c
    con = c.connect(host="localhost", user="root", password="1234", database="superstore")
    query = "SELECT * FROM shop"
    df = pd.read_sql(query, con)
    print(tabulate(df, headers='keys', tablefmt='psql', showindex=False))
```

Frontend

Press 1

```
PS C:\Users\padma\OneDrive\Desktop\dbms_project> python -u "c:\Users\padma\OneDrive\Desktop\dbms_project\shop_management.py"
---------- Shop Management System----------
Menu
1. Press 1 to show the details product available.
2. Press 2 to show details of shop_id and their corresponding managers
3. Press 3  to show average salary provided by employee
4. Press 4 to show the number of employees working in our organization
5. Press 5 to show the id of the number of employee gettting maximum salary
6. Press 6 to show the no. of customers associated with us.
7. Press 7 to show the details of employees getting salary ranging from 5000-8000.
8. Press 8 to show the price of the product with maximum price.
9. Press 9 to show details of the product between the the price 10-20
10. Press 10 to show details of all branhes of an organizarion.
Enter the choice1
c:\Users\padma\OneDrive\Desktop\dbms_project\shop_management.py:22: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database str
ing URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  df=pd.read_sql(query,con)
+----------------+---------+
| product_name   |   price |
|----------------+---------|
| almonds        |     480 |
| amchoor100g    |      40 |
| arhar500g      |      90 |
| biscuit        |      10 |
| ghee1kg        |     700 |
| maggi          |      20 |
| refined        |      80 |
| rice           |     160 |
| salt           |      30 |
| walnut         |     120 |
+----------------+---------+
PS C:\Users\padma\OneDrive\Desktop\dbms_project>
```

Press2

```
PS C:\Users\padma\OneDrive\Desktop\dbms_project> python -u "c:\Users\padma\OneDrive\Desktop\dbms_project\shop_management.py"
---------- Shop Management System----------
Menu
1. Press 1 to show the details product available.
2. Press 2 to show details of shop_id and their corresponding managers
3. Press 3  to show average salary provided by employee
4. Press 4 to show the number of employees working in our organization
5. Press 5 to show the id of the number of employee gettting maximum salary
6. Press 6 to show the no. of customers associated with us.
7. Press 7 to show the details of employees getting salary ranging from 5000-8000.
8. Press 8 to show the price of the product with maximum price.
9. Press 9 to show details of the product between the the price 10-20
10. Press 10 to show details of all branhes of an organizarion.
Enter the choice2
c:\Users\padma\OneDrive\Desktop\dbms_project\shop_management.py:29: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database string URI or sqlite
3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  df=pd.read_sql(query,con)
+-----------+-------------+-------------+
| shop_id   | first_name  | last_name   |
|-----------+-------------+-------------|
| shop01    | Raman       | Singh       |
| shop02    | Aman        | Sindhu      |
| shop03    | Simran      | Garg        |
| shop04    | Chaman      | Yadav       |
| shop05    | Anisha      | Sen         |
| shop06    | Rita        | Sharma      |
| shop07    | Naman       | Sahiwal     |
| shop08    | Viman       | Sandhu      |
| shop09    | Pradyuman   | Sidhu       |
| shop10    | Ash         | Verma       |
+-----------+-------------+-------------+
PS C:\Users\padma\OneDrive\Desktop\dbms_project>
```

Press 4

```
PS C:\Users\padma\OneDrive\Desktop\dbms_project> python -u "c:\Users\padma\OneDrive\Desktop\dbms_project\shop_management.py"
---------- Shop Management System----------------
Menu
1. Press 1 to show the details product available.
2. Press 2 to show details of shop_id and their corresponding managers
3. Press 3  to show average salary provided by employee
4. Press 4 to show the number of employees working in our organization
5. Press 5 to show the id of the number of employee gettting maximum salary
6. Press 6 to show the no. of customers associated with us.
7. Press 7 to show the details of employees getting salary ranging from 5000-8000.
8. Press 8 to show the price of the product with maximum price.
9. Press 9 to show details of the product between the the price 10-20
10. Press 10 to show details of all branhes of an organizarion.
Enter the choice4
c:\Users\padma\OneDrive\Desktop\dbms_project\shop_management.py:43: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database string URI or sqlite
3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  df=pd.read_sql(query,con)
+---------------------+
|   COUNT(employee_id) |
|---------------------|
|                  10 |
+---------------------+
PS C:\Users\padma\OneDrive\Desktop\dbms_project>
```

Press 8

```
PS C:\Users\padma\OneDrive\Desktop\dbms_project> python -u "c:\Users\padma\OneDrive\Desktop\dbms_project\shop_management.py"
---------- Shop Management System----------------
Menu
1. Press 1 to show the details product available.
2. Press 2 to show details of shop_id and their corresponding managers
3. Press 3  to show average salary provided by employee
4. Press 4 to show the number of employees working in our organization
5. Press 5 to show the id of the number of employee gettting maximum salary
6. Press 6 to show the no. of customers associated with us.
7. Press 7 to show the details of employees getting salary ranging from 5000-8000.
8. Press 8 to show the price of the product with maximum price.
9. Press 9 to show details of the product between the the price 10-20
10. Press 10 to show details of all branhes of an organizarion.
Enter the choice8
c:\Users\padma\OneDrive\Desktop\dbms_project\shop_management.py:71: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database string URI or sqlite
3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.
  df=pd.read_sql(query,con)
+---------+
|   price |
|---------|
|     700 |
+---------+
PS C:\Users\padma\OneDrive\Desktop\dbms_project>
```