Introduction to Database Systems
CSC 675/775

Project Name

Store Inventory Database

Team Details / Section 01

Arjun Sharma
Jesus Correa
Madina Ahmadzai
Nanda Pandian
Prathiba Ramesh
Purva Zinjarde

“Phase 2”

May 6, 2022

Task 1 - Create Tables, indexes and constraints using DDL

CREATE DATABASE StoreInventory;

1.

CREATE TABLE Products
(productID INT NOT NULL, productName VARCHAR (50),
productQty INT, price FLOAT, batchNO INT, manufacDate DATETIME,
expiryDate DATETIME, PRIMARY KEY (productID));

2.

CREATE TABLE Supplier
(supplierID INT NOT NULL, supplierName VARCHAR (30),
location VARCHAR (30), PRIMARY KEY (supplierID));

Creating a Non-Clustered Index on search key <location> on table Supplier.

CREATE INDEX index_location ON Supplier (location);

3.

CREATE TABLE StoreBranch
(branchCode INT NOT NULL, branchName VARCHAR (20), address VARCHAR (40),
numberOfEmp INT, activeSince INT, yearlyRevenue FLOAT,
PRIMARY KEY (branchCode));

Creating a Non-Clustered Index on the search key <numberOfEmp> on table StoreBranch.

CREATE INDEX index_numberOfEmp ON StoreBranch(numberOfEmp);

4.

CREATE TABLE Employee
(employeeID INT NOT NULL, employeeName VARCHAR (30), shiftDuration FLOAT,
wages FLOAT, designation VARCHAR (30), branchCode INT,
PRIMARY KEY (employeeID), FOREIGN KEY (branchCode) REFERENCES StoreBranch
(branchCode));

Creating View on table Employee

CREATE VIEW EmployeeDetails AS SELECT * FROM Employee;

5.

CREATE TABLE FullTime_Employee
(employeeID INT NOT NULL, incrementAmt FLOAT,
PRIMARY KEY (employeeID), FOREIGN KEY (employeeID) REFERENCES Employee
(employeeID));

6.

CREATE TABLE PartTime_Employee
(employeeID INT NOT NULL, shiftTime VARCHAR(10),
PRIMARY KEY (employeeID), FOREIGN KEY (employeeID) REFERENCES Employee
(employeeID));

7.

CREATE TABLE LogisticProvider
(logID INT NOT NULL, logName VARCHAR (30), phone VARCHAR(20),
city VARCHAR(10), address VARCHAR (40), transportMethod VARCHAR (20),
PRIMARY KEY (logID));

8.

CREATE TABLE StoreProduct
(productID INT NOT NULL, branchCode INT NOT NULL,
PRIMARY KEY (productID, branchCode),
FOREIGN KEY (productID) REFERENCES Products(productID),
FOREIGN KEY (branchCode) REFERENCES StoreBranch(branchCode));

9.

CREATE TABLE SupplierProduct
(productID INT NOT NULL, supplierID INT NOT NULL,
PRIMARY KEY (productID, supplierID),
FOREIGN KEY (productID) REFERENCES Products(productID),
FOREIGN KEY (supplierID) REFERENCES Supplier(supplierID));

10.

```
CREATE TABLE Customer
(customerID INT NOT NULL, customerName VARCHAR (30), phone VARCHAR(20),
address VARCHAR (20), PRIMARY KEY (customerID));
```

Creating View on table Customer

```
CREATE VIEW CustomerInfo AS SELECT customerID, customerName FROM Customer;
```

11.

```
CREATE TABLE CustomerProduct
( productID INT NOT NULL, customerID INT NOT NULL,
PRIMARY KEY (productID, customerID),
FOREIGN KEY (productID) REFERENCES Products(productID),
FOREIGN KEY (customerID) REFERENCES Customer(customerID));
```

12.

```
CREATE TABLE CustomerStoreBranch
( customerID INT NOT NULL, branchCode INT NOT NULL,
PRIMARY KEY (customerID , branchCode),
FOREIGN KEY (customerID) REFERENCES Customer(customerID),
FOREIGN KEY (branchCode) REFERENCES StoreBranch(branchCode));
```

13.

```
CREATE TABLE EmployeeStoreBranch
( employeeID INT NOT NULL, branchCode INT NOT NULL,
PRIMARY KEY (employeeID , branchCode),
FOREIGN KEY (employeeID) REFERENCES Employee(employeeID),
FOREIGN KEY (branchCode) REFERENCES StoreBranch(branchCode));
```

14.

```
CREATE TABLE SupplierLogProvider (
supplierID INT NOT NULL, logID INT NOT NULL,
PRIMARY KEY (supplierID , logID),
FOREIGN KEY (supplierID) REFERENCES Supplier(supplierID),
FOREIGN KEY (logID) REFERENCES LogisticProvider(logID));
```

Task 2 - Collect and Import Data into Tables

1. **Products:**

Insert Queries:

```
INSERT INTO Products
VALUES(111, "Milk", 200, 3.4, 4352376, "2022-05-03 12:34:00", "2022-05-23 12:34:00");

INSERT INTO Products
VALUES(112, "Eggs", 450, 2.6, 888798, "2022-05-01 07:05:00", "2022-06-01 12:00:00");

INSERT INTO Products
VALUES(113, "Yogurt", 600, 3.5, 723647, "2022-03-15 05:39:00", "2022-10-15 09:56:00");

INSERT INTO Products
VALUES(114, "Cereal", 300, 1.5, 76799890, "2022-01-28 01:30:00", "2023-06-28 09:30:00");

INSERT INTO Products
VALUES(115, "Apples", 270, 6.0, 87244476, "2022-02-14 02:56:00", "2022-05-28 10:02:00");

INSERT INTO Products
VALUES(116, "Onions", 200, 2.0, 87564476, "2022-02-14 02:56:00", "2022-05-28 10:02:00");

INSERT INTO Products
VALUES(117, "Ketchup", 450, 3.2, 8756471, "2022-02-14 02:56:00", "2022-05-28 10:02:00");

INSERT INTO Products
VALUES(118, "Strawberries", 200, 8.0, 1256471, "2022-02-14 02:56:00", "2022-05-28 10:02:00");

INSERT INTO Products
VALUES(119, "Tissues", 300, 1.2, 987471, "2022-02-14 02:56:00", "2022-05-28 10:02:00");

INSERT INTO Products
VALUES(120, "Chicken", 270, 11.0, 909871, "2022-02-14 02:56:00", "2022-05-28 10:02:00");
```

```
mysql> SELECT * FROM Products;
+-----------+--------------+------------+-------+----------+---------------------+---------------------+
| productID | productName  | productQty | price | batchNO  | manufacDate         | expiryDate          |
+-----------+--------------+------------+-------+----------+---------------------+---------------------+
|       111 | Milk         |        200 |   3.4 |  4352376 | 2022-05-03 12:34:00 | 2022-05-23 12:34:00 |
|       112 | Eggs         |        450 |   2.6 |   888798 | 2022-05-01 07:05:00 | 2022-06-01 12:00:00 |
|       113 | Yogurt       |        600 |   3.5 |   723647 | 2022-03-15 05:39:00 | 2022-10-15 09:56:00 |
|       114 | Cereal       |        300 |   1.5 | 76799890 | 2022-01-28 01:30:00 | 2023-06-28 09:30:00 |
|       115 | Apples       |        270 |     6 | 87244476 | 2022-02-14 02:56:00 | 2022-05-28 10:02:00 |
|       116 | Onions       |        200 |     2 | 87564476 | 2022-02-14 02:56:00 | 2022-05-28 10:02:00 |
|       117 | Ketchup      |        450 |   3.2 |  8756471 | 2022-02-14 02:56:00 | 2022-05-28 10:02:00 |
|       118 | Strawberries |        200 |     8 |  1256471 | 2022-02-14 02:56:00 | 2022-05-28 10:02:00 |
|       119 | Tissues      |        300 |   1.2 |   987471 | 2022-02-14 02:56:00 | 2022-05-28 10:02:00 |
|       120 | Chicken      |        270 |    11 |   909871 | 2022-02-14 02:56:00 | 2022-05-28 10:02:00 |
+-----------+--------------+------------+-------+----------+---------------------+---------------------+
10 rows in set (0.02 sec)
```

2. Supplier

```
INSERT INTO Supplier
VALUES(001, "Big Bob's Distributors", "San Jose");

INSERT INTO Supplier
VALUES(002, "Retail Distribution Bros", "Austin");

INSERT INTO Supplier
VALUES(003, "Transcon Trading", "Monterey");

INSERT INTO Supplier
VALUES(004, "Bluewaves Supply", "Fremont");

INSERT INTO Supplier
VALUES(005, "Bakers Inc", "Mountain View");
```

```
mysql> SELECT * FROM Supplier;
+------------+-------------------------+--------------+
| supplierID | supplierName            | location     |
+------------+-------------------------+--------------+
|          1 | Big Bob's Distributors  | San Jose     |
|          2 | Retail Distribution Bros | Austin       |
|          3 | Transcon Trading        | Monterey     |
|          4 | Bluewaves Supply        | Fremont      |
|          5 | Bakers Inc              | Mountain View |
+------------+-------------------------+--------------+
5 rows in set (0.01 sec)
```

3. StoreBranch

```
INSERT INTO StoreBranch
VALUES(1001, "Sunnyvale Branch", "123, Maple Hill, Sunnyvale", 6, 2015, 12000000);

INSERT INTO StoreBranch
VALUES(1002, "San Jose Branch", "404, Orchard Ave, San Jose", 10, 2012, 56000000);

INSERT INTO StoreBranch
VALUES(1003, "San Francisco Branch", "500, Holloway Ave, San Francisco", 16, 2018, 48000000);

INSERT INTO StoreBranch
VALUES(1004, "Palo Alto Branch", "488, Mason Bridge Rd, Palo Alto", 5, 2017, 11000000);

INSERT INTO StoreBranch
VALUES(1005, "Cupertino Branch", "990, Apple Blvd, Cupertino", 10, 2019, 51000000);
```

```
mysql> SELECT * FROM StoreBranch;
+------------+----------------------+----------------------------------+-------------+-------------+---------------+
| branchCode | branchName           | address                          | numberOfEmp | activeSince | yearlyRevenue |
+------------+----------------------+----------------------------------+-------------+-------------+---------------+
|       1001 | Sunnyvale Branch     | 123, Maple Hill Dr, Sunnyvale    |           6 |        2015 |      12000000 |
|       1002 | San Jose Branch      | 404, Orchard Ave, San Jose       |          10 |        2012 |      56000000 |
|       1003 | San Francisco Branch | 500, Holloway Ave, San Francisco |          16 |        2018 |      48000000 |
|       1004 | Palo Alto Branch     | 488, Mason Bridge Rd, Palo Alto  |           5 |        2017 |      11000000 |
|       1005 | Cupertino Branch     | 990, Apple Blvd, Cupertino       |          10 |        2019 |      51000000 |
+------------+----------------------+----------------------------------+-------------+-------------+---------------+
5 rows in set (0.01 sec)
```

4. **Employee:**

```
INSERT INTO Employee
VALUES  (2001, "Dolores Kaplan", 8.0, 15.50, "Cashier", 1001);

INSERT INTO Employee
VALUES (2002, "Carrie Allen", 8.0, 16.00, "Cashier", 1004);

INSERT INTO Employee
VALUES (2003, "Jane Alston", 7.0, 15.50, "Stocker", 1001);

INSERT INTO Employee
VALUES (2004, "Eric Foreman", 8.0, 15.00, "Stocker", 1005);

INSERT INTO Employee
VALUES (2005, "Red Foreman", 8.0, 20.00, "Manager", 1005);

INSERT INTO Employee
VALUES (2006, "Ross Geller", 6.0, 16.50, "Deli Clerk", 1002);

INSERT INTO Employee
VALUES (2007, "Chandler Bing", 8.0, 17.00, "Cashier", 1003);
```

```
mysql> select * from Employee;
+------------+----------------+---------------+-------+-------------+------------+
| employeeID | employeeName   | shiftDuration | wages | designation | branchCode |
+------------+----------------+---------------+-------+-------------+------------+
|       2001 | Dolores Kaplan |             8 |  15.5 | Cashier     |       1001 |
|       2002 | Carrie Allen   |             8 |    16 | Cashier     |       1004 |
|       2003 | Jane Alston    |             7 |  15.5 | Stocker     |       1001 |
|       2004 | Eric Foreman   |             8 |    15 | Stocker     |       1005 |
|       2005 | Red Foreman    |             8 |    20 | Manager     |       1005 |
|       2006 | Ross Geller    |             6 |  16.5 | Deli Clerk  |       1002 |
|       2007 | Chandler Bing  |             8 |    17 | Cashier     |       1003 |
+------------+----------------+---------------+-------+-------------+------------+
7 rows in set (0.00 sec)
```

5. **FullTime_Employee:**

INSERT INTO FullTime_Employee values (2001, .75);

INSERT INTO FullTime_Employee values (2002, .50);

INSERT INTO FullTime_Employee values (2003, .25);

INSERT INTO FullTime_Employee values (2004, .25);

INSERT INTO FullTime_Employee values (2005, 1.00);

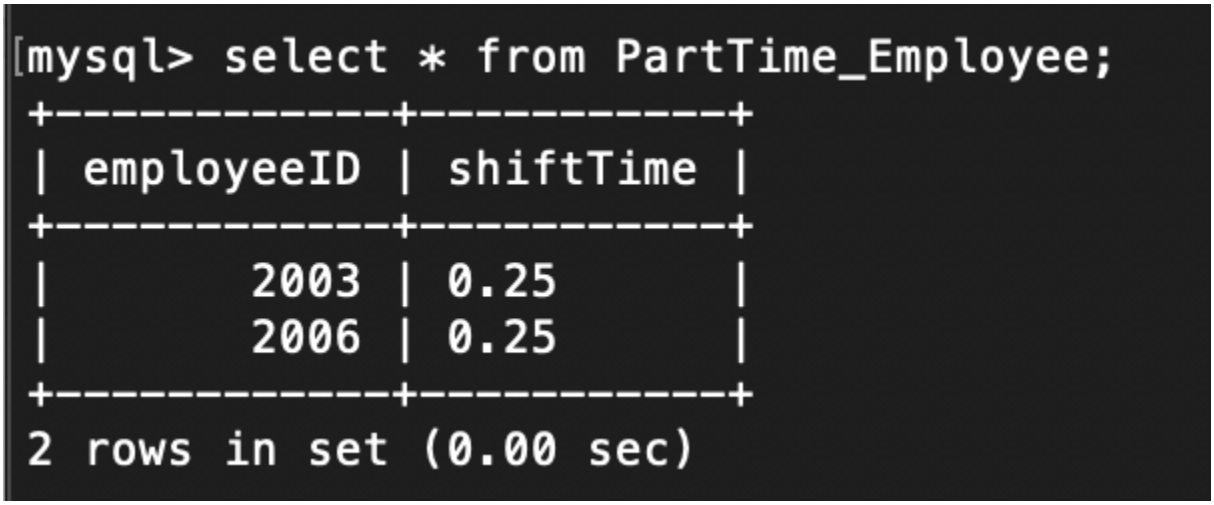
INSERT INTO FullTime_Employee values (2007, .75);

```
mysql> select * from FullTime_Employee;
+------------+--------------+
| employeeID | incrementAmt |
+------------+--------------+
|       2001 |         0.75 |
|       2002 |          0.5 |
|       2004 |         0.25 |
|       2005 |            1 |
|       2007 |         0.75 |
+------------+--------------+
5 rows in set (0.00 sec)
```

6. **PartTime_Employee:**

INSERT INTO PartTime_Employee values (2003, .25);

INSERT INTO PartTime_Employee values (2006, .25);



7. LogisticProvider:

INSERT INTO LogisticProvider
VALUES (9990, "Wholesome Hauling", "4081236676", "Nashville", "74, Emerson Blvd, Nashville", "Trucks");

INSERT INTO LogisticProvider
VALUES (9991, "Haul In A Hurry", "3136778909", "Portland", "99, Milan Dr, Portland", "Cargo Planes");

INSERT INTO LogisticProvider
VALUES (9992, "Lift And Ride", "3459008765", "Oakland", "100, Morningstar Dr, Oakland City", "Trucks");

INSERT INTO LogisticProvider
VALUES (9993, "Ship And Sort", "4569098165", "Virginia", "776, Rockford Ave, Virginia Beach", "Cargo Ships");

INSERT INTO LogisticProvider
VALUES (9994, "Fresh Start Hauling", "3139765165", "Ann Arbor", "1346, Glencoe Hill Dr, Ann Arbor", "Trucks");

```
mysql> SELECT * FROM LogisticProvider;
+-------+--------------------+------------+-----------+----------------------------------+-----------------+
| logID | logName            | phone      | city      | address                          | transportMethod |
+-------+--------------------+------------+-----------+----------------------------------+-----------------+
|  9990 | Wholesome Hauling  | 4081236676 | Nashville | 74, Emerson Blvd, Nashville      | Trucks          |
|  9991 | Haul In A Hurry    | 3136778909 | Portland  | 99, Milan Dr, Portland           | Cargo Planes    |
|  9992 | Lift And Ride      | 3459008765 | Oakland   | 100, Morningstar Dr, Oakland City | Trucks          |
|  9993 | Ship And Sort      | 4569098165 | Virginia  | 776, Rockford Ave, Virginia Beach | Cargo Ships     |
|  9994 | Fresh Start Hauling | 3139765165 | Ann Arbor | 1346, Glencoe Hill Dr, Ann Arbor  | Trucks          |
+-------+--------------------+------------+-----------+----------------------------------+-----------------+
5 rows in set (0.02 sec)
```

8. StoreProduct

INSERT INTO StoreProduct VALUES (111, 1002);

INSERT INTO StoreProduct VALUES (114, 1005);

INSERT INTO StoreProduct VALUES (113, 1004);

INSERT INTO StoreProduct VALUES (111, 1001);

INSERT INTO StoreProduct VALUES (112, 1003);

INSERT INTO StoreProduct VALUES (114, 1001);

INSERT INTO StoreProduct VALUES (115, 1003);

INSERT INTO StoreProduct VALUES (111, 1005);

INSERT INTO StoreProduct VALUES (113, 1001);

```
mysql> SELECT * FROM StoreProduct;
+-----------+------------+
| productID | branchCode |
+-----------+------------+
|       111 |       1001 |
|       113 |       1001 |
|       114 |       1001 |
|       111 |       1002 |
|       112 |       1003 |
|       115 |       1003 |
|       113 |       1004 |
|       115 |       1004 |
|       111 |       1005 |
|       114 |       1005 |
+-----------+------------+
10 rows in set (0.02 sec)
```

9. SupplierProduct

INSERT INTO SupplierProduct VALUES (113, 2);

INSERT INTO SupplierProduct VALUES (111, 5);

INSERT INTO SupplierProduct VALUES (111, 3);

INSERT INTO SupplierProduct VALUES (113, 3);

INSERT INTO SupplierProduct VALUES (114, 5);

INSERT INTO SupplierProduct VALUES (112, 5);

INSERT INTO SupplierProduct VALUES (115, 1);

INSERT INTO SupplierProduct VALUES (115, 4);

INSERT INTO SupplierProduct VALUES (113, 1);

INSERT INTO SupplierProduct VALUES (114, 1);

```
mysql> select * from SupplierProduct;
+-----------+------------+
| productID | supplierID |
+-----------+------------+
|       113 |          1 |
|       114 |          1 |
|       115 |          1 |
|       113 |          2 |
|       111 |          3 |
|       113 |          3 |
|       115 |          4 |
|       111 |          5 |
|       112 |          5 |
|       114 |          5 |
+-----------+------------+
10 rows in set (0.00 sec)
```

10. Customer:

INSERT INTO Customer
VALUES (1, “Keith McGlothlin”, “239-209-0423”, “4870 Owen Lane”);

INSERT INTO Customer
VALUES (2, "John P Clark", "407-424-8365", "81 Ocala Street");

INSERT INTO Customer
VALUES (3, "Tony Whitford", "856-689-9352", "560 Prospect Street");

INSERT INTO Customer
VALUES (4, "Thomas Rogers", "321-633-7871", "1285 Stonebrook Road");

INSERT INTO Customer
VALUES (5, "Christen Broussard", "515-320-9045", "150 Hazelwood Avenue");

```
mysql> select * from Customer;
+------------+--------------------+--------------+----------------------+
| customerID | customerName       | phone        | address              |
+------------+--------------------+--------------+----------------------+
|          1 | Keith McGlothlin   | 239-209-0423 | 4870 Owen Lane       |
|          2 | John P Clark       | 407-424-8365 | 81 Ocala Street      |
|          3 | Tony Whitford      | 856-689-9352 | 560 Prospect Street  |
|          4 | Thomas Rogers      | 321-633-7871 | 1285 Stonebrook Road |
|          5 | Christen Broussard | 515-320-9045 | 150 Hazelwood Avenue |
+------------+--------------------+--------------+----------------------+
5 rows in set (0.00 sec)
```

11. CustomerProduct:

INSERT INTO CustomerProduct VALUES (114, 3);

INSERT INTO CustomerProduct VALUES(111, 1);

INSERT INTO CustomerProduct VALUES(113, 5);

INSERT INTO CustomerProduct VALUES (113, 1);

INSERT INTO CustomerProduct VALUES (112, 2);

INSERT INTO CustomerProduct VALUES (115, 4);

INSERT INTO CustomerProduct VALUES (115, 2);

INSERT INTO CustomerProduct VALUES (112, 4);

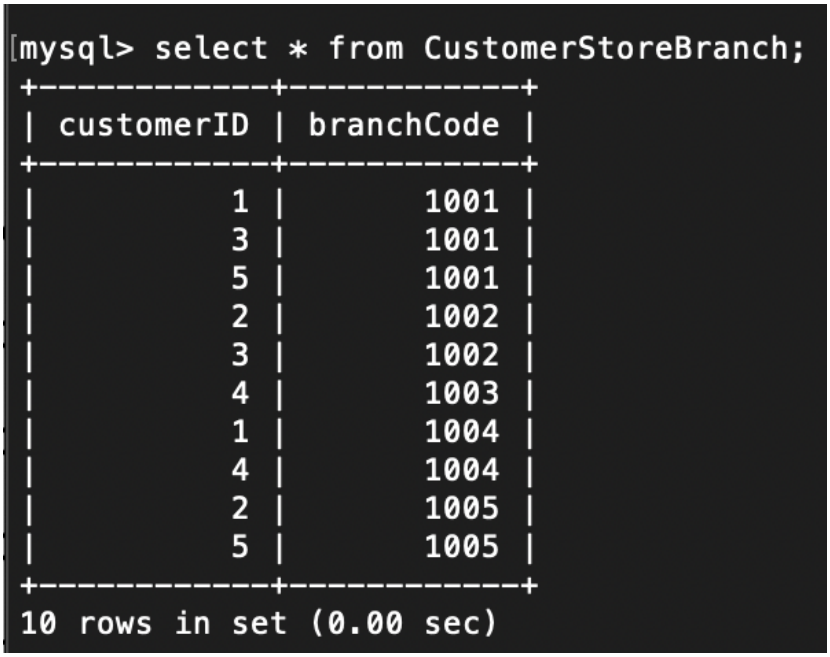
INSERT INTO CustomerProduct VALUES (111, 5);

INSERT INTO CustomerProduct VALUES (111, 3);

```
mysql> select * from CustomerProduct;
+-----------+------------+
| productID | customerID |
+-----------+------------+
|       111 |          1 |
|       113 |          1 |
|       112 |          2 |
|       115 |          2 |
|       111 |          3 |
|       114 |          3 |
|       112 |          4 |
|       115 |          4 |
|       111 |          5 |
|       113 |          5 |
+-----------+------------+
10 rows in set (0.00 sec)
```

12. CustomerStoreBranch:

```
INSERT INTO CustomerStoreBranch VALUES (3, 1001);

INSERT INTO CustomerStoreBranch VALUES (1, 1004);

INSERT INTO CustomerStoreBranch VALUES (5, 1001);

INSERT INTO CustomerStoreBranch VALUES (4, 1003);

INSERT INTO CustomerStoreBranch VALUES (2, 1005);

INSERT INTO CustomerStoreBranch VALUES (1, 1001);

INSERT INTO CustomerStoreBranch VALUES (4, 1004);

INSERT INTO CustomerStoreBranch VALUES (3, 1002);

INSERT INTO CustomerStoreBranch VALUES (5, 1005);

INSERT INTO CustomerStoreBranch VALUES (2, 1002);
```



```
mysql> select * from CustomerStoreBranch;
+------------+------------+
| customerID | branchCode |
+------------+------------+
|          1 |       1001 |
|          3 |       1001 |
|          5 |       1001 |
|          2 |       1002 |
|          3 |       1002 |
|          4 |       1003 |
|          1 |       1004 |
|          4 |       1004 |
|          2 |       1005 |
|          5 |       1005 |
+------------+------------+
10 rows in set (0.00 sec)
```

13. EmployeeStoreBranch:

INSERT INTO EmployeeStoreBranch VALUES (2001, 1001);

INSERT INTO EmployeeStoreBranch VALUES (2002, 1004);

INSERT INTO EmployeeStoreBranch VALUES (2003, 1003);

INSERT INTO EmployeeStoreBranch VALUES (2004, 1005);

INSERT INTO EmployeeStoreBranch VALUES (2005, 1002);

INSERT INTO EmployeeStoreBranch VALUES (2006, 1005);

INSERT INTO EmployeeStoreBranch VALUES (2007, 1001);

```
mysql> select * from EmployeeStoreBranch;
+------------+------------+
| employeeID | branchCode |
+------------+------------+
|       2001 |       1001 |
|       2007 |       1001 |
|       2005 |       1002 |
|       2003 |       1003 |
|       2002 |       1004 |
|       2004 |       1005 |
|       2006 |       1005 |
+------------+------------+
7 rows in set (0.00 sec)
```

14. SupplierLogProvider:

```
INSERT INTO SupplierLogProvider VALUES(1, 9993);

INSERT INTO SupplierLogProvider VALUES(4, 9991);

INSERT INTO SupplierLogProvider VALUES(3, 9990);

INSERT INTO SupplierLogProvider VALUES(5, 9994);

INSERT INTO SupplierLogProvider VALUES(3, 9993);

INSERT INTO SupplierLogProvider VALUES(2, 9991);

INSERT INTO SupplierLogProvider VALUES(2, 9994);

INSERT INTO SupplierLogProvider VALUES(1, 9990);

INSERT INTO SupplierLogProvider VALUES(3, 9992);

INSERT INTO SupplierLogProvider VALUES(4, 9991);

INSERT INTO SupplierLogProvider VALUES(5, 9993);
```

```
mysql> SELECT * FROM SupplierLogProvider;
+------------+-------+
| supplierID | logID |
+------------+-------+
|          1 |  9990 |
|          3 |  9990 |
|          4 |  9990 |
|          2 |  9991 |
|          4 |  9991 |
|          3 |  9992 |
|          1 |  9993 |
|          3 |  9993 |
|          5 |  9993 |
|          2 |  9994 |
|          5 |  9994 |
+------------+-------+
11 rows in set (0.00 sec)
```

Task3: Write SQL Queries

- At least 2 queries involving GROUP BY, HAVING with aggregate operators.
- At least 2 nested queries involving IN, EXIST, op ANY, op ALL

1. GROUP BY:

The query shows the maximum price of each productQuantity from the Products table by using a GROUP BY query.

```
SELECT P.productQty, MAX(P.price)
FROM Products P
GROUP BY P.productQty;
```

```
mysql> SELECT P.productQty, MAX(P.price) FROM Products P GROUP BY P.productQty;
+------------+--------------+
| productQty | MAX(P.price) |
+------------+--------------+
|        200 |            8 |
|        450 |          3.2 |
|        600 |          3.5 |
|        300 |          1.5 |
|        270 |           11 |
+------------+--------------+
5 rows in set (0.05 sec)
```

2. HAVING:

The query shows a result of giving average wage from selected designation, in this case cashier from Employee by using the HAVING Query.

```
SELECT designation, AVG(wages)
FROM Employee
GROUP BY designation
HAVING designation = "Cashier";
```

```
mysql> SELECT designation, AVG(wages) FROM Employee GROUP BY designation HAVING
 designation = "Cashier";
+-------------+--------------------+
| designation | AVG(wages)         |
+-------------+--------------------+
| Cashier     | 16.166666666666668 |
+-------------+--------------------+
1 row in set (0.00 sec)
```

3. ANY:

In this we are using ANY query on the Employee table and getting Employee names who have ShiftDuration of work more than 6 hours.

```
SELECT E.employeeName
FROM Employee E
WHERE E.shiftDuration > ANY
        (SELECT E2.shiftDuration
        FROM Employee E2
        WHERE E2.shiftDuration = 6);
```

```
mysql> SELECT E.employeeName FROM Employee E WHERE E.shiftDuration > ANY (SELECT E2.shiftDuration
FROM Employee E2 WHERE E2.shiftDuration = 6);
+----------------+
| employeeName   |
+----------------+
| Dolores Kaplan |
| Carrie Allen   |
| Jane Alston    |
| Eric Foreman   |
| Red Foreman    |
| Chandler Bing  |
+----------------+
6 rows in set (0.01 sec)
```

4. EXISTS:

This returns a table of Employees and the Branch where they work.

```
SELECT employeeName, branchName, Employee.branchCode
FROM Employee, StoreBranch
WHERE EXISTS
        (SELECT branchName
         FROM StoreBranch
         HAVING Employee.branchCode = StoreBranch.branchCode)
```

```
mysql> SELECT employeeName, branchName, Employee.branchCode FROM Employee, StoreBranch WHERE EXISTS
 (SELECT branchName  FROM StoreBranch  HAVING Employee.branchCode = StoreBranch.branchCode);
+----------------+----------------------+------------+
| employeeName   | branchName           | branchCode |
+----------------+----------------------+------------+
| Dolores Kaplan | Sunnyvale Branch     |       1001 |
| Carrie Allen   | Palo Alto Branch     |       1004 |
| Jane Alston    | Sunnyvale Branch     |       1001 |
| Eric Foreman   | Cupertino Branch     |       1005 |
| Red Foreman    | Cupertino Branch     |       1005 |
| Ross Geller    | San Jose Branch      |       1002 |
| Chandler Bing  | San Francisco Branch |       1003 |
+----------------+----------------------+------------+
7 rows in set (0.01 sec)
```

5. ALL:

Getting the store with the highest yearly revenue.

```
SELECT * FROM StoreBranch S
WHERE S.yearlyRevenue >= ALL
        ( SELECT S1.yearlyRevenue
        FROM StoreBranch S1)
```

```
mysql> SELECT * FROM StoreBranch S WHERE S.yearlyRevenue >= ALL (SELECT S1.yearlyRevenue FROM StoreBranch S1);
+------------+----------------+---------------------------+-------------+-------------+---------------+
| branchCode | branchName     | address                   | numberOfEmp | activeSince | yearlyRevenue |
+------------+----------------+---------------------------+-------------+-------------+---------------+
|       1002 | San Jose Branch | 404, Orchard Ave, San Jose |          10 |        2012 |      56000000 |
+------------+----------------+---------------------------+-------------+-------------+---------------+
1 row in set (0.01 sec)
```