DEERWALK INSTITUTE OF TECHNOLOGY

Tribhuvan University

Faculties of Computer Science



Bachelors of Science in Computer Science and Information Technology (B.Sc. CSIT)

Course: Theory of Computation (CSC 262)

Class of 2027/Semester: IV

A Lab Report On:

**IMPLEMENTATION OF**

**Submitted by:**

Name: Shaurav Bhandari

Roll No: 1333

**Submitted to:**

Er Satya Ram Suwal

Faculty, Department of Computer Science

**Definition of tables and insertion of all data for First part:**

1. **Salesman Table Definition:**

**Code:**

CREATE TABLE salesman (

salesman\_id **INT** PRIMARY KEY,

name VARCHAR(100),

city VARCHAR(100),

commission DECIMAL(4,2)

);

1. **Customer Table Definition:**

**Code:**

CREATE TABLE customer (

customer\_id **INT** PRIMARY KEY,

customer\_name VARCHAR(100),

city VARCHAR(100),

grade **INT**,

salesman\_id **INT**,

FOREIGN KEY (salesman\_id) REFERENCES salesman(salesman\_id)

);

1. **Order Table Definition:**

**Code:**

CREATE TABLE orders (

order\_no **INT** PRIMARY KEY,

purch\_amt DECIMAL(10,2),

order\_date DATE,

customer\_id **INT**,

salesman\_id **INT**,

FOREIGN KEY (customer\_id) REFERENCES customer(customer\_id),

FOREIGN KEY (salesman\_id) REFERENCES salesman(salesman\_id)

);

1. **Insertion in Salesman:**

**Code:**

INSERT INTO salesman VALUES

(5001, ***'James Hoog'***, ***'New York'***, 0.15),

(5002, ***'Nail Knite'***, ***'Paris'***, 0.13),

(5005, ***'Pit Alex'***, ***'London'***, 0.11),

(5006, ***'Mc Lyon'***, ***'Paris'***, 0.14),

(5003, ***'Lauson Hen'***, ***'London'***, 0.12),

(5007, ***'Paul Adam'***, 'Rome', 0.13);

1. **Insertion in Customer:**

**Code:**

INSERT INTO customer VALUES

(3002, ***'Nick Rimando'***, ***'New York'***, 100, 5001),

(3005, ***'Graham Zusi'***, ***'California'***, 200, 5002),

(3001, ***'Brad Guzan'***, ***'London'***, 200, 5005),

(3004, ***'Fabian Johns'***, ***'Paris'***, 300, 5006),

(3007, ***'Brad Davis'***, ***'New York'***, 200, 5001),

(3009, ***'Geoff Camero'***, ***'Berlin'***, 100, 5003),

(3008, ***'Julian Green'***, ***'London'***, 300, 5002),

(3003, ***'Jozy Altidor'***, ***'Moncow'***, 200, 5007);

1. **Insertion in Order:**

**Code:**

INSERT INTO orders VALUES

(70001, 150.5, ***'2016-10-05'***, 3005, 5002),

(70009, 270.65, ***'2016-09-10'***, 3001, 5005),

(70002, 65.26, ***'2016-10-05'***, 3002, 5001),

(70004, 110.5, ***'2016-08-17'***, 3009, 5003),

(70007, 948.5, ***'2016-09-10'***, 3007, 5001),

(70005, 2400.6, ***'2016-07-27'***, 3007, 5001),

(70008, 5760.0, ***'2016-09-10'***, 3002, 5001),

(70010, 1983.43, ***'2016-10-10'***, 3004, 5006),

(70003, 2480.4, ***'2016-10-10'***, 3009, 5003),

(70012, 250.45, ***'2016-06-27'***, 3008, 5002),

(70011, 75.29, ***'2016-08-17'***, 3003, 5007);

1. **Definition of item\_mast:**

**Code:**

CREATE TABLE item\_mast (

pro\_id **INT** PRIMARY KEY,

pro\_name VARCHAR(50),

pro\_price DECIMAL(10, 2),

pro\_com **INT**

);

1. **Insertions:**

**Code:**

INSERT INTO item\_mast (pro\_id, pro\_name, pro\_price, pro\_com) VALUES

(101, ***'Mother Board'***, 3200.00, 15),

(102, ***'Key Board'***, 450.00, 16),

(103, ***'ZIP drive'***, 250.00, 14),

(104, ***'Speaker'***, 550.00, 16),

(105, ***'Monitor'***, 5000.00, 11),

(106, ***'DVD drive'***, 900.00, 12),

(107, ***'CD drive'***, 800.00, 12),

(108, ***'Printer'***, 2600.00, 13),

(109, ***'Refill cartridge'***, 350.00, 13),

(110, ***'Mouse'***, 250.00, 12);

**Queries:**

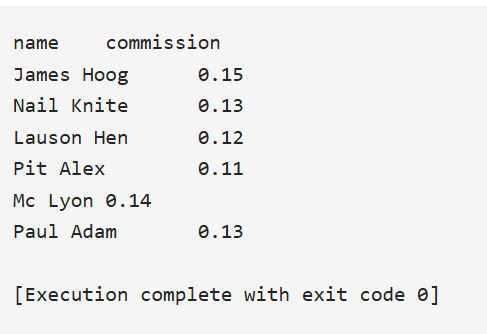
1. **Display Name and commission of all salesmans:**

**Code:**

**SELECT** name, commission

**FROM** salesman;

**Output:**

****

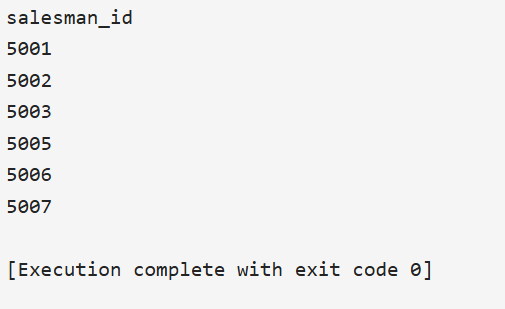
1. **Retrieve salesman id of all salesmen from orders table without any repeats:**

**Code:**

**SELECT** **DISTINCT** salesman\_id

**FROM** orders;

**Output:**

****

1. **Display names and city of salesman, who belongs to the city of Paris.**

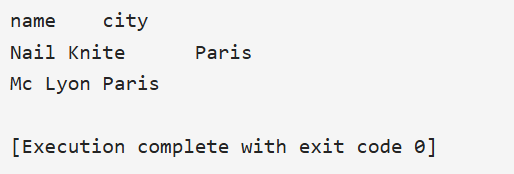
**Code:**

**SELECT** name, city

**FROM** salesman

**WHERE** city = 'Paris';

**Output:**

****

1. **Display all the information for those customers with a grade of 200.**

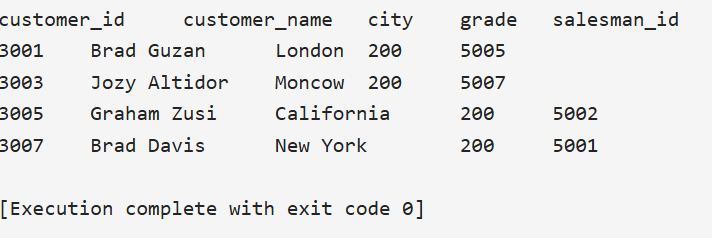
**Code:**

**SELECT** \*

**FROM** customer

**WHERE** grade = 200;

**Output:**

****

1. **Display the order number, order date and the purchase amount for order(s) which will be delivered by the salesman with ID 5001.**

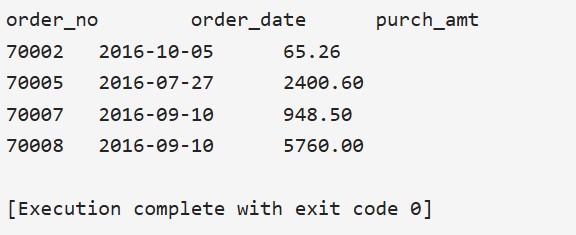
**Code:**

**SELECT** order\_no, order\_date, purch\_amt

**FROM** orders

**WHERE** salesman\_id = 5001;

**Output:**

****

**Definition of tables and insertion of all data Second Part:**

**Code:**

CREATE TABLE nobel\_win (

year **INT**,

subject VARCHAR(100),

winner VARCHAR(255),

country VARCHAR(100),

category VARCHAR(100)

);

**Insertions:**

INSERT INTO nobel\_win (year, subject, winner, country, category) VALUES

(1970, ***'Physics'***, ***'Hannes Alfven'***, ***'Sweden'***, ***'Scientist'***),

(1970, ***'Physics'***, ***'Louis Neel'***, ***'France'***, ***'Scientist'***),

(1970, ***'Chemistry'***, ***'Luis Federico Leloir'***, ***'France'***, ***'Scientist'***),

(1970, ***'Physiology'***, ***'Ulf von Euler'***, ***'Sweden'***, ***'Scientist'***),

(1970, ***'Physiology'***, ***'Bernard Katz'***, ***'Germany'***, ***'Scientist'***),

(1970, ***'Literature'***,***'Aleksandr Solzhenitsyn'***,***'Russia'***, ***'Linguist'***),

(1970, ***'Economics'***, ***'Paul Samuelson'***,'USA', ***'Economist'***),

(1970, ***'Physiology'***, ***'Julius Axelrod'***,'USA', ***'Scientist'***),

(1971, ***'Physics'***, ***'Dennis Gabor'***,***'Hungary'***, ***'Scientist'***),

(1971, ***'Chemistry'***, ***'Gerhard Herzberg'***,***'Germany'***, ***'Scientist'***),

(1971, ***'Peace'***, ***'Willy Brandt'***,***'Germany'***, ***'Chancellor'***),

(1971, ***'Literature'***, ***'Pablo Neruda'***,***'Chile'***, ***'Linguist'***),

(1971, ***'Economics'***, ***'Simon Kuznets'***,***'Russia'***, ***'Economist'***),

(1978, ***'Peace'***, ***'Anwar al-Sadat'***,***'Egypt'***, ***'President'***),

(1978, ***'Peace'***, ***'Menachem Begin'***,***'Israel'***, ***'Prime Minister'***),

(1987, ***'Chemistry'***, ***'Donald J. Cram'***,'USA', ***'Scientist'***),

(1987, ***'Chemistry'***, ***'Jean-Marie Lehn'***,***'France'***, ***'Scientist'***),

(1987, ***'Physiology'***,***'Susumu Tonegawa'***,***'Japan'***, ***'Scientist'***),

(1987, ***'Physics'***, ***'Johannes Georg Bednorz'***,***'Germany'***, ***'Scientist'***),

(1987, ***'Literature'***, ***'Joseph Brodsky'***,***'Russia'***, ***'Linguist'***),

(1987, ***'Economics'***, ***'Robert Solow'***,'USA', ***'Economist'***),

(1994, ***'Economics'***, ***'Reinhard Selten'***,***'Germany'***, ***'Economist'***),

(1994, ***'Peace'***, ***'Yitzhak Rabin'***,***'Israel'***, ***'Prime Minister'***),

(1994, ***'Literature'***,***'Kenzaburo Oe'***,***'Japan'***, ***'Linguist'***);

1. **Show the winner of the 1971 prize for Literature.**

**Code:**

**SELECT** winner

**FROM** nobel\_win

**WHERE** **year** = 1971 **AND** subject = 'Literature';

**Output:**

****

1. **Show all the details of the winners with first name Louis.**

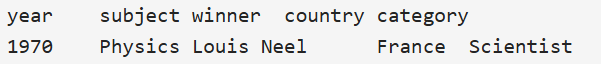
**Code:**

**SELECT** \*

**FROM** nobel\_win

**WHERE** winner **LIKE** 'Louis%';

**Output:**

****

1. **Show all the winners in Physics for 1970 together with the winner of Economics for 1971.**

**Code:**

**SELECT** \* **FROM** nobel\_win

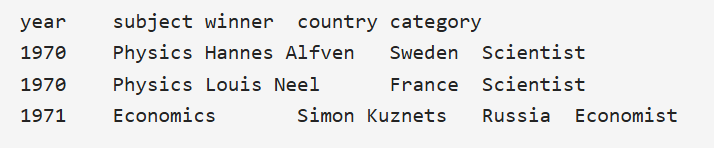
**WHERE** subject = 'Physics' **AND** **year** = 1970

**UNION**

**SELECT** \* **FROM** nobel\_win

**WHERE** subject = 'Economics' **AND** **year** = 1971;

**Output:**

****

1. **Show all the winners of Nobel prize in the year 1970 except the subject Physiology and Economics.**

**Code:**

**SELECT** \* **FROM** nobel\_win

**WHERE** **year** = 1970

**AND** subject **NOT** **IN** ('Physiology','Economics');

**Output:**

****

1. **Find all the details of the Nobel winners for the subject not started with the letter 'P' and arranged the list as the most recent comes first, then by name in order.**

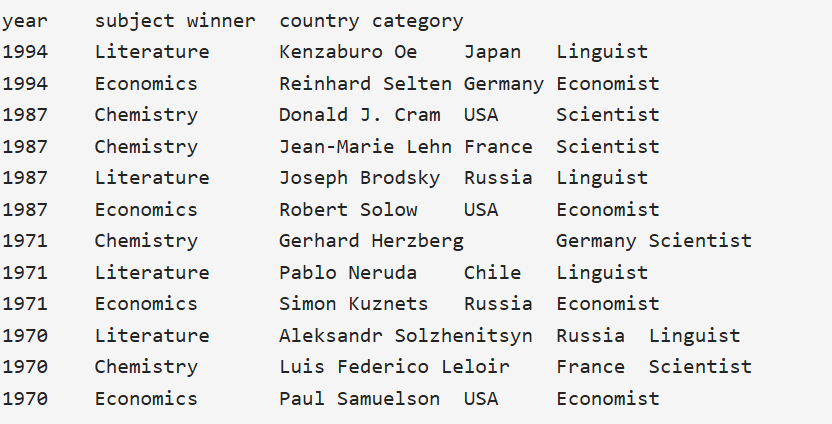
**Code:**

**SELECT** \* **FROM** nobel\_win

**WHERE** subject **NOT** **LIKE** 'P%'

**ORDER** **BY** **year** **DESC**, winner;

**Output:**

****

1. **Find the name and price of the cheapest item(s).**

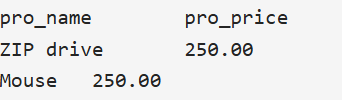
**Code:**

**SELECT** pro\_name, pro\_price

**FROM** item\_mast

**WHERE** pro\_price = (**SELECT** **MIN**(pro\_price) **FROM** item\_mast);

**Output:**

****

1. **Display all the customers, who are either belongs to the city New York or not had a grade above 100.**

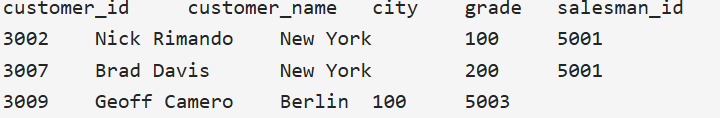
**Code:**

**SELECT** \*

**FROM** customer

**WHERE** city = 'New York' **OR** grade <= 100;

**Output:**

****

1. **Find those salesmen with all information who gets the commission within a range of 0.12 and 0.14.**

**Code:**

**SELECT** salesman\_id, name, city, commission

**FROM** salesman

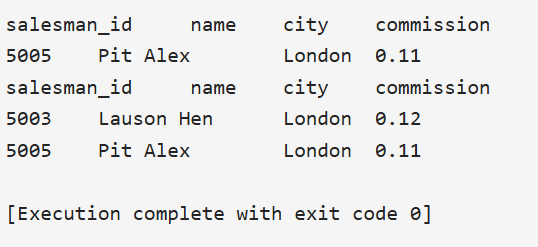
**WHERE** commission > 0.10 **AND** commission < 0.12;

**SELECT** salesman\_id, name, city, commission

**FROM** salesman

**WHERE** commission **BETWEEN** 0.10 **AND** 0.12;

**Output:**

****

1. **Find all those customers with all information whose names are ending with the letter 'n'.**

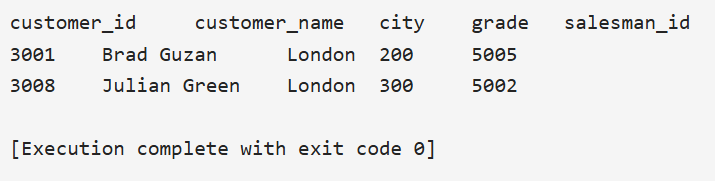
**Code:**

**SELECT** \*

**FROM** customer

**WHERE** customer\_name **LIKE** '%n';

**Output:**

****

1. **Find those salesmen with all information whose name containing the 1st character is 'N' and the 4th character is 'l' and rests may be any character.**

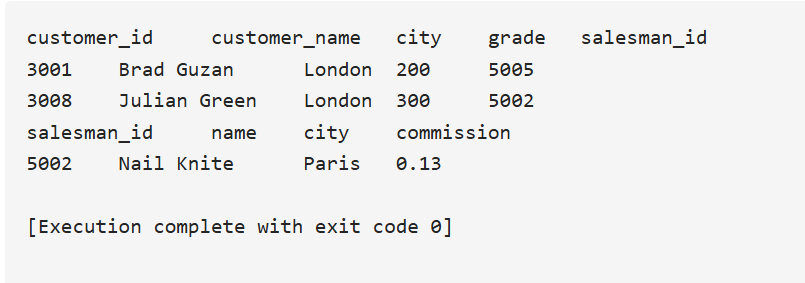
**Code:**

**SELECT** \*

**FROM** salesman

**WHERE** name **LIKE** 'N\_\_l%';

**Output:**

****

1. **Find that customer with all information who does not get any grade except NULL.**

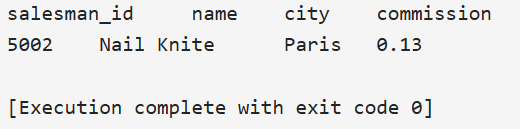
**Code:**

**SELECT** \*

**FROM** customer

**WHERE** grade **IS** **NULL**;

**Output:**

****

1. **Find the total purchase amount of all orders.**

**Code:**

**SELECT** **SUM**(purch\_amt)

**From** orders;

**Output:**

****

1. **Find the number of salesman currently listing for all of their customers.**

**Code:**

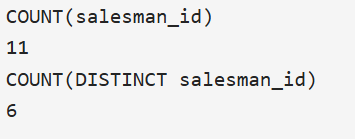
**SELECT** **COUNT**(salesman\_id)

**FROM** orders;

**SELECT** **COUNT**(**DISTINCT** salesman\_id)

**FROM** orders;

**Output:**

****

1. **Find the highest grade for each of the cities of the customers.**

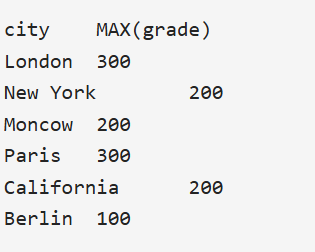
**Code:**

**SELECT** city, **MAX**(grade)

**FROM** customer

**GROUP** **BY** city;

**Output:**

****

1. **Find the highest purchase amount ordered by the each customer with their ID and highest purchase amount.**

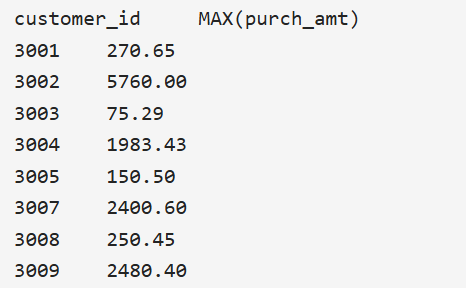
**Code:**

**SELECT** customer\_id, **MAX**(purch\_amt)

**FROM** orders

**GROUP** **BY** customer\_id;

**Output:**

****

1. **Find the highest purchase amount ordered by the each customer on a particular date with their ID, order date and highest purchase amount.**

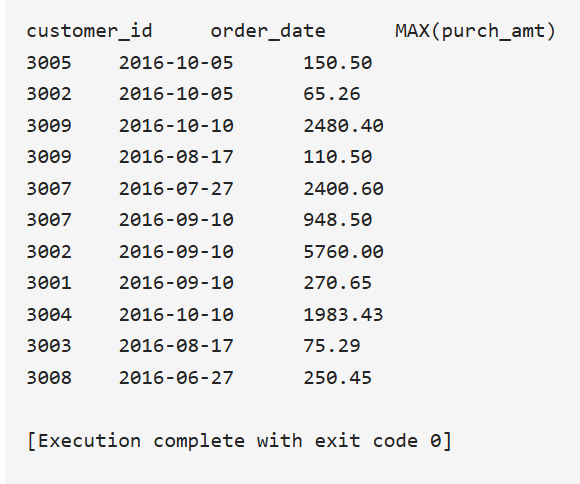
**Code:**

**SELECT** customer\_id, order\_date, **MAX**(purch\_amt)

**FROM** orders

**GROUP** **BY** customer\_id, order\_date;

**Output:**

****

1. **Find the highest purchase amount on a date '2012-08-17' for each salesman with their ID.**

**Code:**

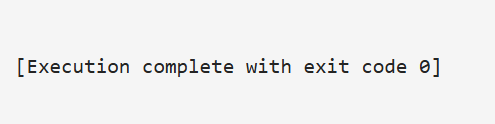
**SELECT** salesman\_id, **MAX**(purch\_amt)

**FROM** orders

**WHERE** order\_date = '2012-08-17'

**GROUP** **BY** salesman\_id;

**Output:**

****

1. **Find the highest purchase amount with their customer ID and order date, for only those customers who have the highest purchase amount in a day is more than 2000.**

**Code:**

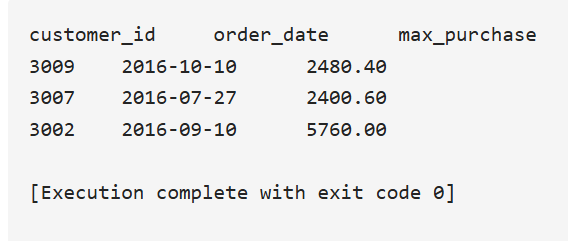
**SELECT** customer\_id, order\_date, **MAX**(purch\_amt) **AS** max\_purchase

**FROM** orders

**GROUP** **BY** customer\_id, order\_date

**HAVING** **MAX**(purch\_amt) > 2000.00;

**Output:**

****

1. **Write a SQL statement that counts all orders for a date August 17th, 2012.**

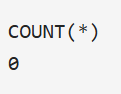
**Code:**

**SELECT** **COUNT**(\*)

**FROM** orders

**WHERE** order\_date = '2012-08-17';

**Output:**

****

**PRACTICE 2 FOR NESTED QUERIES:**

1. **Find the name and city of those customers and salesmen who lives in the same city.**

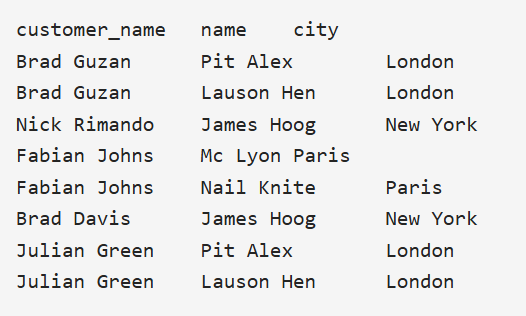
**Code:**

**SELECT** **C**.customer\_name, **S**.name, **S**.city

**FROM** salesman **S**

**JOIN** customer **C** **ON** **S**.city = **C**.city;

**Output:**

****

1. **Find the names of all customers along with the salesmen who works for them.**

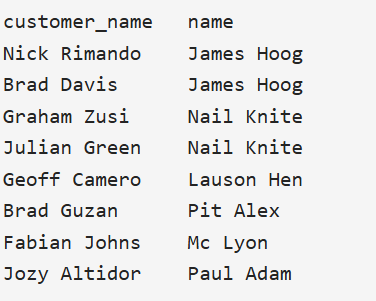
**Code:**

**SELECT** **c**.customer\_name, **s**.name

**FROM** customer **c**

**JOIN** salesman **s** **ON** **s**.salesman\_id = **c**.salesman\_id;

**Output:**

****

1. **Display all those orders by the customers not located in the same cities where their salesmen live.**

**Code:**

**SELECT** o.order\_no, **c**.customer\_name, **c**.customer\_id, o.salesman\_id

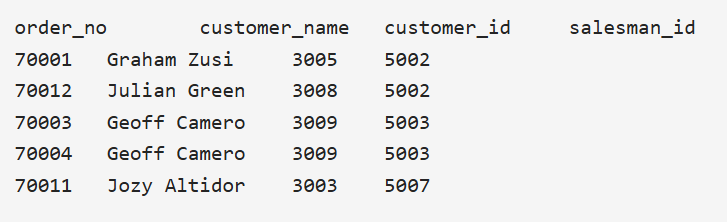
**FROM** orders o

**JOIN** customer **c** **ON** o.customer\_id = **c**.customer\_id

**JOIN** salesman **s** **ON** o.salesman\_id = **s**.salesman\_id

**WHERE** **c**.city <> **s**.city;

**Output:**

****

1. **Display all the orders issued by the salesman 'Paul Adam' from the orders table.**

**Code:**

**SELECT** \*

**FROM** orders

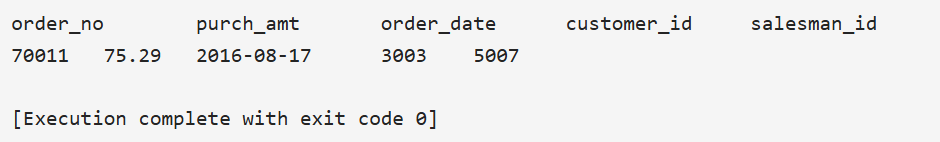
**WHERE** salesman\_id =

(**SELECT** salesman\_id

**FROM** salesman

**WHERE** name = 'Paul Adam');

**Output:**

****

1. **Display all the orders which values are greater than the average order value for 10th October 2012**

**Code:**

**SELECT** \*

**FROM** orders

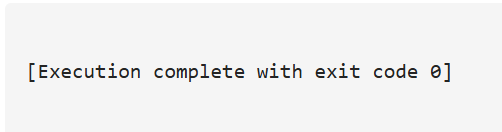
**WHERE** purch\_amt >

(**SELECT** **AVG**(purch\_amt)

**FROM** orders

**WHERE** order\_date = '2012-10-10');

**Output:**

****

1. **Find all orders attributed to salesmen in Paris**

**Code:**

**SELECT** \*

**FROM** orders

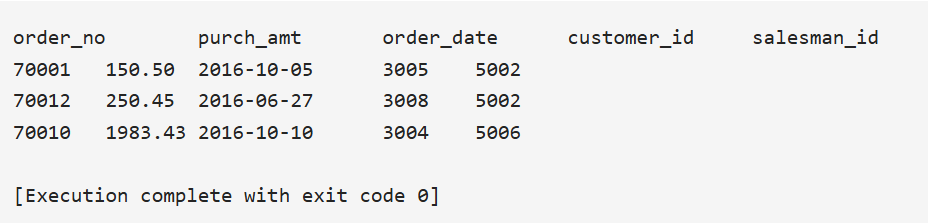
**WHERE** salesman\_id **IN**

(**SELECT** salesman\_id

**FROM** salesman

**WHERE** city = 'Paris');

**Output:**

****

1. **Extract the data from the orders table for the salesman who earned the maximum commission**

**Code:**

**SELECT** order\_no, purch\_amt, order\_date, salesman\_id

**FROM** orders

**WHERE** salesman\_id **IN** (

**SELECT** salesman\_id

**FROM** salesman

**WHERE** commission = (

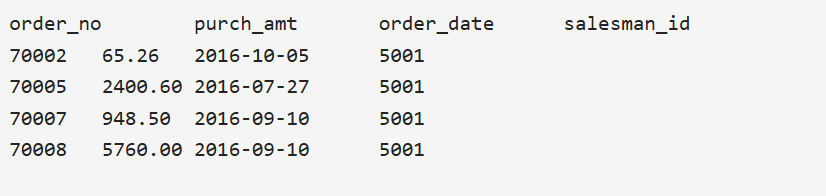
**SELECT** **MAX**(commission)

**FROM** salesman

)

);

**Output:**

****

1. **Find the name and ids of all salesmen who had more than one customer.**

**Code:**

**SELECT** salesman\_id, name

**FROM** salesman **AS** **a**

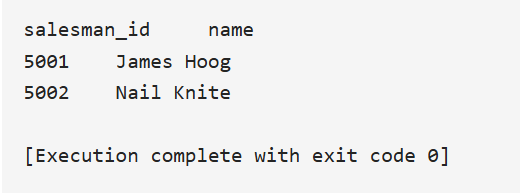
**WHERE** 1 <

(**SELECT** **COUNT**(\*)

**FROM** customer **AS** **c**

**WHERE** **c**.salesman\_id = **a**.salesman\_id);

**Output:**

****

1. **Write a query to find all the salesmen who worked for only one customer**

**Code:**

**SELECT** \*

**FROM** salesman

**WHERE** salesman\_id **IN** (

**SELECT** **DISTINCT** **a**.salesman\_id

**FROM** customer **a**

**WHERE** **NOT** **EXISTS** (

**SELECT** 1

**FROM** customer b

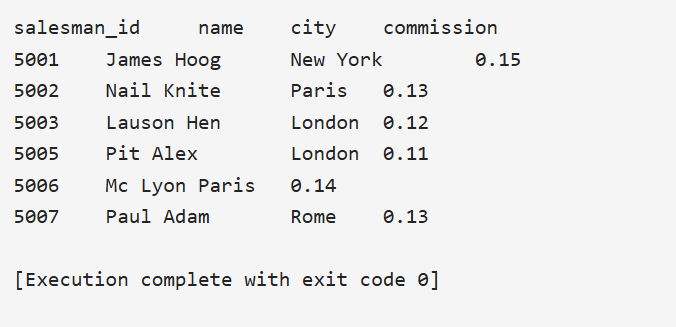
**WHERE** **a**.salesman\_id = b.salesman\_id

**AND** **a**.customer\_name <> b.customer\_name

**AND** b.city **IN** ('Rome', 'San Jose')

)

**Output:**

****

1. **Write a query to find all the salesmen who worked for only one customer.**

**Code:**

**SELECT** **c**.salesman\_id, **s**.name, **s**.city, **s**.commission

**FROM** salesman **s**

**JOIN** customer **c** **ON** **s**.salesman\_id = **c**.salesman\_id

**GROUP** **BY** **c**.salesman\_id, **s**.name, **s**.city, **s**.commission

**HAVING** **COUNT**(**c**.salesman\_id) = 1;

**SELECT** \*

**FROM** salesman

**WHERE** city **IN** ('Rome', 'San Jose')

**AND** salesman\_id **NOT** **IN** (

**SELECT** **a**.salesman\_id

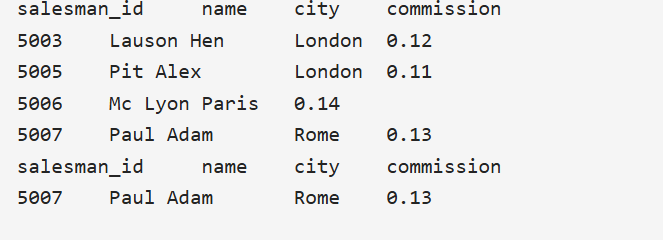
**FROM** customer **a**

**JOIN** customer b **ON** **a**.salesman\_id = b.salesman\_id

**WHERE** **a**.customer\_name <> b.customer\_name

);

**Output:**

****

1. **Display all the orders that had amounts that were greater than at least one of the orders from September 10th 2012.**

**Code:**

**SELECT** \*

**FROM** orders

**WHERE** purch\_amt > **ANY** (

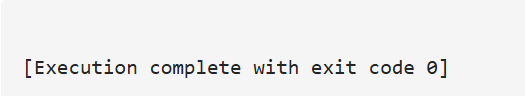
**SELECT** purch\_amt

**FROM** orders

**WHERE** order\_date = '2012-09-10'

);

**Output:**

****

1. **Display only those customers whose grade are, in fact, higher than every customer in New York**

**Code:**

**SELECT** \*

**FROM** customer

**WHERE** grade > **ALL** (

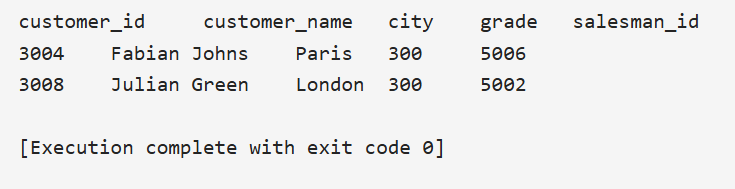
**SELECT** grade

**FROM** customer

**WHERE** city = 'New York'

);

**Output:**

****

**CONCLUSION:**

**Hence, various practices of implementation of DDL, DML, etc. were implemented using MYSQL.**