show databases;

create database company;

use company;

CREATE TABLE salesman (

salesman\_id INT PRIMARY KEY,

name VARCHAR(50),

city VARCHAR(50),

commission DECIMAL(4,2)

);

INSERT INTO salesman VALUES (5001, 'James Hoog', 'New York', 0.15);

INSERT INTO salesman VALUES (5002, 'Nail Knite', 'Paris', 0.13);

INSERT INTO salesman VALUES (5005, 'Pit Alex', 'London', 0.11);

INSERT INTO salesman VALUES (5006, 'Mc Lyon', 'Paris', 0.14);

INSERT INTO salesman VALUES (5003, 'Lauson Hense', ' ', 0.12);

UPDATE salesman

SET city = NULL

WHERE salesman\_id = 5003;

INSERT INTO salesman VALUES (5007, 'Paul Adam', 'Rome', 0.13);

SELECT \* FROM salesman;

CREATE TABLE customer (

customer\_id INT PRIMARY KEY,

cust\_name VARCHAR(50),

city VARCHAR(50),

grade INT,

salesman\_id INT,

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

);

-- Insert values into customer

INSERT INTO customer VALUES (3002, 'Nick Rimando', 'New York', 100, 5001);

INSERT INTO customer VALUES (3005, 'Graham Zusi', 'California', 200, 5002);

INSERT INTO customer VALUES (3004, 'Fabian Johnson', 'Paris', 300, 5006);

INSERT INTO customer VALUES (3007, 'Brad Davis', 'New York', 200, 5001);

INSERT INTO customer VALUES (3009, 'Geoff Cameron', 'Berlin', 100, 5003);

INSERT INTO customer VALUES (3008, 'Julian Green', 'London', 300, 5002);

INSERT INTO customer VALUES (3001, 'Brad Guzan', 'London', NULL, 5005);

INSERT INTO customer VALUES (3003, 'Jozy Altidore', 'Moscow', 200, 5007);

SELECT \* FROM customer;

CREATE TABLE orders (

ord\_no INT PRIMARY KEY,

purch\_amt DECIMAL(10,2),

ord\_date DATE,

customer\_id INT,

salesman\_id INT,

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

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

);

INSERT INTO orders VALUES (70009, 270.65, '2012-09-10', 3001, 5005);

INSERT INTO orders VALUES (70002, 65.26, '2012-10-05', 3002, 5001);

INSERT INTO orders VALUES (70004, 110.50, '2012-08-17', 3009, 5003);

INSERT INTO orders VALUES (70005, 2400.60, '2012-07-27', 3007, 5001);

INSERT INTO orders VALUES (70008, 5760.00, '2012-09-10', 3002, 5001);

INSERT INTO orders VALUES (70010, 1983.43, '2012-10-10', 3004, 5006);

INSERT INTO orders VALUES (70003, 2480.40, '2012-10-10', 3009, 5003);

INSERT INTO orders VALUES (70011, 75.29, '2012-08-17', 3003, 5007);

INSERT INTO orders VALUES (70013, 3045.60, '2012-04-25', 3002, 5001);

INSERT INTO orders VALUES (70001, 150.50, '2012-10-05', 3005, 5002);

INSERT INTO orders VALUES (70007, 948.50, '2012-09-10', 3005, 5002);

INSERT INTO orders VALUES (70012, 250.45, '2012-06-27', 3008, 5002);

/\* 1) From the following tables write a SQL query to find the salesperson(s) and the customer(s) he represents. Return Customer Name, city, Salesman, commission.

\*/

SELECT

c.cust\_name AS Customer\_Name,

c.city AS Customer\_City,

s.name AS Salesman\_Name,

s.commission

FROM

customer c

JOIN

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

/\* 2. From the following tables write a SQL query to display the customer name, customer city, grade, salesman, salesman city. The results should be sorted by ascending customer\_id. \*/

SELECT

c.cust\_name,

c.city AS customer\_city,

c.grade,

s.name AS salesman\_name,

s.city AS salesman\_city

FROM

customer c

JOIN

salesman s ON c.salesman\_id = s.salesman\_id

ORDER BY

c.customer\_id ASC;

/\* 3. From the following tables write a SQL query to locate those salespeople who do not live in the same city where their customers live and have received a commission of more than 12% from the company. Return Customer Name, customer city, Salesman, salesman city, commission. \*/

SELECT

c.cust\_name,

c.city AS customer\_city,

s.name AS salesman\_name,

s.city AS salesman\_city,

s.commission

FROM

customer c

JOIN

salesman s ON c.salesman\_id = s.salesman\_id

WHERE

c.city != s.city AND s.commission > 0.12;

/\* 4. From the following tables write a SQL query to count the number of customers with grades above the average in New York City. Return grade and count. \*/

SELECT

c.grade,

COUNT(c.customer\_id) AS customer\_count

FROM customer AS c

WHERE

c.city = 'New York' AND c.grade > (

SELECT

AVG(c2.grade)

FROM customer AS c2

WHERE

c2.city = 'New York'

)

GROUP BY

c.grade;

/\* 5. Write a query to find the sums of the amounts from the orders table, grouped by date, and eliminate all dates where the sum was not at least 1000.00 above the maximum order amount for that date. \*/

SELECT

ord\_date,

SUM(purch\_amt) AS total\_amount

FROM

orders

GROUP BY

ord\_date

HAVING

total\_amount > (

SELECT MAX(purch\_amt)

FROM orders o2

WHERE o2.ord\_date = orders.ord\_date

) + 1000;

/\* 6. From the following tables write a SQL query to find the salespeople who deal with customers with more than one order. Return salesman\_id, name, city and commission. \*/

SELECT DISTINCT

s.salesman\_id,

s.name,

s.city,

s.commission

FROM

orders o

JOIN

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

JOIN

salesman s ON c.salesman\_id = s.salesman\_id

WHERE

o.customer\_id IN (

SELECT customer\_id

FROM orders

GROUP BY customer\_id

HAVING COUNT(\*) > 1

);

/\* Q7 \*/

SELECT

cust\_city,

SUM(opening\_amt) AS total\_opening\_amt,

AVG(receive\_amt) AS avg\_receive\_amt,

MAX(payment\_amt) AS max\_payment\_amt

FROM

customer

WHERE

grade = 2

GROUP BY

cust\_city

HAVING

AVG(receive\_amt) > 500;

/\* Q8 8. From the table “orders”, arrange the entries in ascending order of their ord\_no. Then return all the entries which have purch\_amt > Rs. 500. \*/

SELECT

cust\_city,

SUM(opening\_amt) AS total\_opening\_amt,

AVG(receive\_amt) AS avg\_receive\_amt,

MAX(payment\_amt) AS max\_payment\_amt

FROM

customer

WHERE

grade = 2

GROUP BY

cust\_city

HAVING

AVG(receive\_amt) > 500;