Q.1 From the following tables write a SQL query to find the salesperson and customer who reside in the same city. Return Salesman, cust_name and city.

Ans:

```
SELECT salesman.name AS "Salesman",
customer.cust_name, customer.city
FROM salesman,customer
WHERE salesman.city=customer.city
```

Q.2 From the following tables write a SQL query to find those orders where the order amount exists between 500 and 2000. Return ord_no, purch_amt, cust_name, city.

Ans:

```
SELECT  a.ord_no,a.purch_amt,
b.cust_name,b.city
FROM orders a,customer b
WHERE a.customer_id=b.customer_id
AND a.purch_amt BETWEEN 500 AND 2000;
```

Q.3 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.

Ans:

```
SELECT a.cust_name AS "Customer Name",
a.city, b.name AS "Salesman", b.commission
FROM customer a
INNER JOIN salesman b
ON a.salesman_id=b.salesman_id;
```

Q.4 From the following tables write a SQL query to find salespeople who received commissions of more than 12 percent from the company. Return Customer Name, customer city, Salesman, commission.

Ans:

```
SELECT a.cust_name AS "Customer Name",
a.city, b.name AS "Salesman", b.commission
FROM customer a
INNER JOIN salesman b
ON a.salesman_id=b.salesman_id
WHERE b.commission>.12;
```

Q.5 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.

Ans:

```
SELECT a.cust_name AS "Customer Name",
a.city, b.name AS "Salesman", b.city,b.commission
FROM customer a
INNER JOIN salesman b
ON a.salesman_id=b.salesman_id
WHERE b.commission>.12
AND a.city<>b.city;
```

Q.6 From the following tables write a SQL query to find the details of an order. Return ord_no, ord_date, purch_amt, Customer Name, grade, Salesman, commission.

Ans:

```
SELECT a.ord_no,a.ord_date,a.purch_amt,
b.cust_name AS "Customer Name", b.grade,
c.name AS "Salesman", c.commission
FROM orders a
INNER JOIN customer b
ON a.customer_id=b.customer_id
INNER JOIN salesman c
ON a.salesman_id=c.salesman_id;
```

Q.7 Write a SQL statement to join the tables salesman, customer and orders so that the same column of each table appears once and only the relational rows are returned.

Ans:

```
SELECT *
FROM orders
NATURAL JOIN customer
NATURAL JOIN salesman;
```

Q.8 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.

Ans:

```
SELECT a.cust_name,a.city,a.grade,
b.name AS "Salesman",b.city
```

```
FROM customer a
LEFT JOIN salesman b
ON a.salesman_id=b.salesman_id
order by a.customer_id;
```

Q.9 From the following tables write a SQL query to find those customers with a grade less than 300. Return cust_name, customer city, grade, Salesman, salesmancity. The result should be ordered by ascending customer_id.

Ans:

```
SELECT a.cust_name,a.city,a.grade,
b.name AS "Salesman", b.city
FROM customer a
LEFT OUTER JOIN salesman b
ON a.salesman_id=b.salesman_id
WHERE a.grade<300
ORDER BY a.customer_id;
```

Q.10 Write a SQL statement to make a report with customer name, city, order number, order date, and order amount in ascending order according to the order date to determine whether any of the existing customers have placed an order or not.

Ans:

```
SELECT a.cust_name,a.city, b.ord_no,
b.ord_date,b.purch_amt AS "Order Amount"
FROM customer a
LEFT OUTER JOIN orders b
ON a.customer_id=b.customer_id
order by b.ord_date;
```

Q.11 SQL statement to generate a report with customer name, city, order number, order date, order amount, salesperson name, and commission to determine if any of the existing customers have not placed orders or if they have placed orders through their salesman or by themselves.

Ans:

```
SELECT a.cust_name,a.city, b.ord_no,
b.ord_date,b.purch_amt AS "Order Amount",
c.name,c.commission
FROM customer a
LEFT OUTER JOIN orders b
ON a.customer_id=b.customer_id
LEFT OUTER JOIN salesman c
ON c.salesman_id=b.salesman_id;
```

Q.12 Write a SQL statement to generate a list in ascending order of salespersons who work either for one or more customers or have not yet joined any of the customers.

Ans:

```
SELECT a.cust_name,a.city,a.grade,
b.name AS "Salesman", b.city
FROM customer a
RIGHT OUTER JOIN salesman b
ON b.salesman_id=a.salesman_id
ORDER BY b.salesman_id;
```

Q.13 From the following tables write a SQL query to list all salespersons along with customer name, city, grade, order number, date, and amount.

Ans:

```
SELECT a.cust_name,a.city,a.grade,
b.name AS "Salesman",
c.ord_no, c.ord_date, c.purch_amt
FROM customer a
RIGHT OUTER JOIN salesman b
ON b.salesman_id=a.salesman_id
RIGHT OUTER JOIN orders c
ON c.customer_id=a.customer_id;
```

14. Write a SQL statement to make a list for the salesmen who either work for one or more customers or yet to join any of the customer. The customer may have placed, either one or more orders on or above order amount 2000 and must have a grade, or he may not have placed any order to the associated supplier.

Ans:

```
SELECT a.cust_name,a.city,a.grade,
b.name AS "Salesman",
c.ord_no, c.ord_date, c.purch_amt
FROM customer a
RIGHT OUTER JOIN salesman b
ON b.salesman_id=a.salesman_id
LEFT OUTER JOIN orders c
ON c.customer_id=a.customer_id
WHERE c.purch_amt>=2000
AND a.grade IS NOT NULL;
```

Q.15 For those customers from the existing list who put one or more orders, or which orders have been placed by the customer who is not on the list, create a report containing the customer name, city, order number, order date, and purchase amount

Ans:

```
SELECT a.cust_name,a.city, b.ord_no,
b.ord_date,b.purch_amt AS "Order Amount"
FROM customer a
LEFT OUTER JOIN orders b
ON a.customer_id=b.customer_id;
```

**********MongoDB**********

Write a MongoDB query to display all the documents in the collection restaurants.

ANS : db.restaurants.find();

Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine for all the documents in the collection restaurant.

Ans : db.restaurants.find({},{"restaurant_id":1,"name":1,"borough":1,"cuisine" :1});

Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine, but exclude the field _id for all the documents in the collection restaurant.

Ans:
db.restaurants.find({},{"restaurant_id":1,"name":1,"borough":1,"cuisine" :1,"_id":0});

Write a MongoDB query to display the fields restaurant_id, name, borough and zip code, but exclude the field _id for all the documents in the collection restaurant.

Ans : db.restaurants.find({},{"restaurant_id" :
1,"name":1,"borough":1,"address.zipcode" :1,"_id":0});

Write a MongoDB query to display all the restaurant which is in the borough Bronx.

Ans : db.restaurants.find({"borough": "Bronx"});

Write a MongoDB query to display the first 5 restaurant which is in the borough Bronx.

Ans : db.restaurants.find({"borough": "Bronx"}).limit(5);

Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx.

Ans: db.restaurants.find({"borough": "Bronx"}).skip(5).limit(5);

Write a MongoDB query to find the restaurants who achieved a score more than 90.

Ans:

db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 90}}}});

Write a MongoDB query to find the restaurants that achieved a score, more than 80 but less than 100.

Ans: db.restaurants.find({grades : { $elemMatch:{"score":{$gt : 80 , $lt :100}}}});

Write a MongoDB query to find the restaurants which locate in latitude value less than -95.754168.

Ans : db.restaurants.find({"address.coord" : {$lt : -95.754168}});

Write a MongoDB query to find the restaurants that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than -65.754168.

Ans : db.restaurants.find(

```
{$and:
   [
     {"cuisine" : {$ne :"American "}},
     {"grades.score" : {$gt : 70}},
     {"address.coord" : {$lt : -65.754168}}
   ]
}
   );
```

Write a MongoDB query to find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish.

Ans : db.restaurants.find(

{

"borough": "Bronx" ,

$or : [

{ "cuisine" : "American " },

{ "cuisine" : "Chinese" }

]

}

);