CS 315 Modern Database Management

Fall 2016

Northeastern Illinois University

SQL Exercise Sheet

Create the following tables in your local MySQL database and write the queries in your SQL editor.

Aggregate Function Exercises

Table name: customers

customer_id	cust_name	city	grade	salesman_id
3002	Nick Rimando	New York	100	5001
3005	Graham Zusi	California	200	5002
3001	Brad Guzan	London		5005
3004	Fabian Johns	Paris	300	5006
3007	Brad Davis	New York	200	5001
3009	Geoff Camero	Berlin	100	5003

1. Create SQL code that shows how many customer have listed their names.

```
SELECT COUNT(*)

FROM customer;
```

2. Create SQL code that selects the highest grade for each of the cities of the customers.

```
SELECT city,MAX(grade)

FROM customer

GROUP BY city;
```

3. Create SQL code that find the number of customers who gets at least a gradation for his/her performance.

```
SELECT COUNT (ALL grade)

FROM customer;
```

Table name: orders

ord_no	purch_amt	ord_date	customer_id	salesman_id
70001	150.5	2012-10-05	3005	5002
70009	270.65	2012-09-10	3001	5005
70002	65.26	2012-10-05	3002	5001
70004	110.5	2012-08-17	3009	5003
70007	948.5	2012-09-10	3005	5002
70005	2400.6	2012-07-27	3007	5001

4. Create SQL code that finds the total purchase amount of all orders.

```
SELECT SUM (purch_amt)

FROM orders;
```

5. Create SQL code that finds the average purchase amount of all orders.

```
SELECT AVG (purch_amt)

FROM orders;
```

6. Create SQL code that to finds the number of salesmen currently listing for all of their customers.

```
SELECT COUNT (DISTINCT salesman_id)

FROM orders;
```

7. Create SQL code that gets the maximum purchase amount of all the orders.

```
SELECT MAX (purch_amt)

FROM orders;
```

8. Create SQL code that gets the minimum purchase amount of all the orders.

```
SELECT MIN(purch_amt)

FROM orders;
```

9. Create SQL code that finds the highest purchase amount ordered by the each customer with their ID and highest purchase amount.

```
SELECT customer_id,MAX(purch_amt)

FROM orders

GROUP BY customer_id;
```

10. Create SQL code that finds the highest purchase amount ordered by the each customer on a particular date with their ID, order date and highest purchase amount.

```
SELECT customer_id,ord_date,MAX(purch_amt)

FROM orders

GROUP BY customer_id,ord_date;
```

11. Create SQL code that finds the highest purchase amount on a date '2012-08-17' for each salesman with their ID.

```
SELECT salesman_id,MAX(purch_amt)
```

```
FROM orders

WHERE ord_date = '2012-08-17'

GROUP BY salesman_id;
```

12. Create SQL code that finds the highest purchase amount with their ID and order date, for only those customers who have highest purchase amount in a day is more than 2000.

```
SELECT customer_id,ord_date,MAX(purch_amt)

FROM orders

GROUP BY customer_id,ord_date

HAVING MAX(purch_amt)>2000.00;
```

13. Create SQL code that finds the highest purchase amount with their ID and order date, for those customers who have a higher purchase amount in a day is within the range 2000 and 6000.

```
SELECT customer_id,ord_date,MAX(purch_amt)

FROM orders

GROUP BY customer_id,ord_date

HAVING MAX(purch_amt) BETWEEN 2000 AND 6000;
```

14. Create SQL code that finds the highest purchase amount with their ID, for only those customers whose ID is within the range 3002 and 3007.

```
SELECT customer_id,MAX(purch_amt)

FROM orders

WHERE customer_id BETWEEN 3002 and 3007

GROUP BY customer_id;
```

Table name: salesman

salesman_id	name	city	commission
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		0.12
5007	Paul Adam	Rome	0.13

16. Create SQL code that counts the number of different non NULL city values for salesmen.

```
SELECT COUNT(*)
```

```
FROM salesman

WHERE city IS NOT NULL;
```

17. Create SQL code that counts the number of salesmen with their order date and ID registering orders for each day.

```
SELECT ord_date,salesman_id,COUNT(*)

FROM orders

GROUP BY ord_date,salesman_id;
```

Joins Exercises

salesman_id	name	city	commission
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		0.12
5007	Paul Adam	Rome	0.13

customer_id	cust_name	city	grade	salesman_id
3002	Nick Rimando	New York	100	5001
3005	Graham Zusi	California	200	5002
3001	Brad Guzan	London		5005
3004	Fabian Johns	Paris	300	5006
3007	Brad Davis	New York	200	5001
3009	Geoff Camero	Berlin	100	5003

ord_no	purch_amt	ord_date	customer_id	salesman_id
70001	150.5	2012-10-05	3005	5002
70009	270.65	2012-09-10	3001	5005
70002	65.26	2012-10-05	3002	5001
70004	110.5	2012-08-17	3009	5003
70007	948.5	2012-09-10	3005	5002
70005	2400.6	2012-07-27	3007	5001

1. Create SQL code that makes a list with order no, purchase amount, customer name and their cities for those orders which order amount between 500 and 2000.

```
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;
```

2. Create SQL code that shows which salesman are working for which customer.

```
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;
```

3. Create SQL code that finds the list of customers who appointed a salesman for their jobs who gets a commission from the company is more than 12%.

```
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;
```

4. Create SQL code that finds the list of customers who appointed a salesman for their jobs who does not live in same city where there customer lives, and gets a commission is above 12% .

```
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;
```

5. Create SQL code that makes a list in ascending order for the customer who works either through a salesman or by own.

```
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;
```