Q.1) Display the total amount of payment made in every month. Your results should display month in word (Eg. JANUARY, FEBRUARY ...) and the total amount paid in that month.
ANS

```
SELECT SUM(AMOUNT),CASE WHEN DATE LIKE '%-01-%'
THEN 'JANUARY' WHEN DATE LIKE '%-02-%' THEN
'FEBRUARY' WHEN DATE LIKE '%-03-%' THEN 'MARCH' WHEN
DATE LIKE '%-04-%' THEN 'APRIL' WHEN DATE LIKE '%-05-%'
THEN 'MAY' WHEN DATE LIKE '%-06-%' THEN 'JUNE' WHEN
DATE LIKE '%-07-%' THEN 'JULY' WHEN DATE LIKE'%-08-%'
THEN 'AUGUST' WHEN DATE LIKE '%-10-%' THEN 'OCTOBER'
WHEN DATE LIKE '%-11-%' THEN 'NOVEMBER' WHEN DATE
LIKE '%-12-%' THEN 'DECEMBER' ELSE 'INVALID' END AS
MONTH from PAYMENT GROUP BY MONTH;
```

```
+-------------+---------+
| SUM(AMOUNT) | MONTH   |
+-------------+---------+
|        8000 | APRIL   |
|     2080428 | JANUARY |
|       10000 | MARCH   |
|     1053714 | OCTOBER |
+-------------+---------+
4 rows in set (0.05 sec)
```

Q.2) Print as the following condition satisfies for the time that passed between two payments for the same payment number.

- If the time between payments is within 1 year: Flag should be assigned 'FIRST YEAR'.
- If the time between payments is between 1st and 2nd year, then Flag should be assigned 'SECONDYEAR'.
- If time between payments is between 2nd and 3rd year, then flag should be assigned “THIRD YEAR”.

- Otherwise assign flag = “Defaulter”.

You must display the Payment Number, first date of payment, second date of payment, date difference and Flag.
You may use the following hint.

Hint: DATEDIFF(date1, date2)–gives difference between the dates in days.

ANS

```
SELECT
S1.P_NO,S1.DATE,S2.DATE,DATEDIFF(S2.DATE,S1.DATE) AS
DDIFF, CASE WHEN DATEDIFF(S2.DATE,S1.DATE)/365<1 THEN
'FIRST YEAR' WHEN DATEDIFF(S2.DATE,S1.DATE)/365<2 AND
DATEDIFF(S2.DATE,S1.DATE)/365>1 THEN 'SECOND YEAR'
WHEN DATEDIFF(S2.DATE,S1.DATE)/365<3 AND
DATEDIFF(S2.DATE,S1.DATE)/365>2 THEN 'THIRD YEAR' ELSE
'DEFAULTER' END AS FLAG FROM PAYMENT AS S1,PAYMENT
AS S2 WHERE S1.P_NO=S2.P_NO AND S2.DATE>S1.DATE
ORDER BY FLAG;
```

```
+------+------------+------------+-------+-------------+
| P_NO | DATE       | DATE       | DDIFF | FLAG        |
+------+------------+------------+-------+-------------+
| p2   | 2000-01-09 | 2011-10-09 |  4291 | DEFAULTER   |
| p2   | 2000-01-09 | 2012-03-11 |  4445 | DEFAULTER   |
| p1   | 2011-01-09 | 2011-03-11 |    61 | FIRST YEAR  |
| p1   | 2011-10-08 | 2012-04-11 |   186 | FIRST YEAR  |
| p1   | 2012-04-11 | 2012-10-25 |   197 | FIRST YEAR  |
| p1   | 2011-01-09 | 2011-10-11 |   275 | FIRST YEAR  |
| p1   | 2011-10-11 | 2012-04-11 |   183 | FIRST YEAR  |
| p1   | 2011-01-09 | 2011-10-08 |   272 | FIRST YEAR  |
| p1   | 2011-10-08 | 2011-10-11 |     3 | FIRST YEAR  |
| p2   | 2011-10-09 | 2012-03-11 |   154 | FIRST YEAR  |
| p1   | 2011-03-11 | 2011-10-08 |   211 | FIRST YEAR  |
| p1   | 2011-03-11 | 2011-10-11 |   214 | FIRST YEAR  |
```

p1	2011-03-11	2012-10-25	594	SECOND YEAR
p1	2011-01-09	2012-10-25	655	SECOND YEAR
p1	2011-10-08	2012-10-25	383	SECOND YEAR
p1	2011-03-11	2012-04-11	397	SECOND YEAR
p1	2011-10-11	2012-10-25	380	SECOND YEAR
p1	2011-01-09	2012-04-11	458	SECOND YEAR
+------+------------+------------+-------+-------------+
18 rows in set (0.05 sec)

Q.3) Create a view which contains Branch Name, the Total amount in that branch and its rank based on the total amount. (For eg. If the total amount in the branch is >= 10000 then its Rank is 1. If it is >=8000 and <10000, then rank is 2, If it is >=7000 and <8000, then rank is 3 Otherwise rank is 4.

ANS:

CREATE VIEW ZONE_RANK_INFO AS select BR_NAME,sum(BALANCE) as sum,CASE WHEN sum(BALANCE)>=10000 THEN 1 WHEN sum(BALANCE)<10000 AND sum(BALANCE)>=8000 THEN 2 WHEN SUM(BALANCE)>=7000 AND SUM(BALANCE)<8000 THEN 3 ELSE 4 END AS RANK_ZONE from ACCOUNT_COPY group by BR_NAME order by sum(BALANCE) DESC;

Note: Tas please check by running the following queries one by one.

1. UPDATE ACCOUNT_COPY SET BALANCE = 12000 WHERE

BR_NAME = 'zone6';
Query OK, 1 row affected (0.10 sec)
Rows matched: 1 Changed: 1 Warnings: 0

2. SELECT * FROM ZONE_RANK_INFO;

BR_NAME	sum	RANK_ZONE
zone1	13000	1
zone6	12000	1
zone2	10000	1
zone4	10000	1
zone7	9000	2
zone8	8000	2
zone9	7000	3
zone3	6000	4

8 rows in set (0.00 sec)

3. UPDATE ACCOUNT_COPY SET BALANCE = 6000 WHERE BR_NAME = 'zone6';
Query OK, 1 row affected (0.07 sec)
Rows matched: 1 Changed: 1 Warnings: 0

4. SELECT * FROM ZONE_RANK_INFO;

BR_NAME	sum	RANK_ZONE
zone1	13000	1
zone2	10000	1
zone4	10000	1
zone7	9000	2
zone8	8000	2
zone9	7000	3
zone3	6000	4

| zone6 | 6000 | 4 |
+---------+-------+-----------+
8 rows in set (0.00 sec)

Q.4) Run the following queries one by one.

```
DROP TABLE IF EXISTS ORDERS;

 CREATE TABLE ORDERS (ORD_ID INT, AMOUNT FLOAT,
ORD_DATE DATE, CUSTOMERID INT, SALESMANID INT,
PRIMARY KEY(ORD_ID));

INSERT INTO ORDERS VALUES (1, 270.65,'2012-09-10', 3001,
5005),(2, 150.5,'2012-10-05',3005,5002) ,(3,65.26,'2012-10-05',
3002,5001) ,(4,110.5,'2012-08-17',3009,5003) ,(5,948.5,'2012-09-10',
3005,5002) ,(6,2400.6,'2012-07-27',3007,5001) ,(7,5760,'2012-09-10',
3002,5001) ,(8,1983.43,'2012-10-10',3004,5006) ,
(9,2480.4,'2012-10-10',3009,5003),(10,250.45,'      2012-06-27',
3008,5002),(11,75.29,'2012-08-17', 3003, 5007),
(12,3045.6,'2012-04-25',3002,5001);

DROP TABLE IF EXISTS SALESMAN;

CREATE TABLE SALESMAN(SALESMANID INT, NAME
VARCHAR(10), PRIMARY KEY(SALESMANID));

 INSERT INTO SALESMAN VALUES (5001, 'JAMES'),(5002,
'NAIL'),(5005, 'PIT'),(5006, 'LYON'),(5003, 'LAUSON'),(5007,
'PAUL');
```

QUESTION – part a) You have to create a view called GOODSALESMAN that finds the salesman who has the customer with the highest order of a day. Display salesmanid, customerid and amount. Part b) You have to create a view GREATSALESMAN that finds the salesman who has the customer with the highest order of a day at least 3 times overall. Display salesmanid.

ANSWER

```
CREATE VIEW GOODSALESMAN AS SELECT
SALESMANID,CUSTOMERID,AMOUNT FROM ORDERS A,
(SELECT MAX(AMOUNT) AS AMT, ORD_DATE AS ORD_DATE
FROM ORDERS C GROUP BY ORD_DATE) AS T WHERE
T.ORD_DATE = A.ORD_DATE AND T.AMT = A.AMOUNT;

SELECT * FROM GOODSALESMAN;
+------------+------------+--------+
| SALESMANID | CUSTOMERID | AMOUNT |
+------------+------------+--------+
|       5002 |       3005 |  150.5 |
|       5003 |       3009 |  110.5 |
|       5001 |       3007 | 2400.6 |
|       5001 |       3002 |   5760 |
|       5003 |       3009 | 2480.4 |
|       5002 |       3008 | 250.45 |
|       5001 |       3002 | 3045.6 |
+------------+------------+--------+
7 rows in set (0.00 sec)

CREATE VIEW GREATSALESMAN AS SELECT DISTINCT
SALESMANID,COUNT(*) AS A FROM GOODSALESMAN
GROUP BY SALESMANID HAVING A >= 3;
Query OK, 0 rows affected (0.03 sec)

mysql> SELECT * FROM GREATSALESMAN;
+------------+---+
| SALESMANID | A |
+------------+---+
|       5001 | 3 |
+------------+---+
```