

BANK

```
create database bank;
```

```
use bank;
```

```
CREATE TABLE customer_master(  
CUSTOMER_NUMBER VARCHAR(6),  
FIRSTNAME VARCHAR(30),  
middlename VARCHAR(30),  
lastname VARCHAR(30),  
CUSTOMER_CITY VARCHAR(15),  
CUSTOMER_CONTACT_NO VARCHAR(10),  
occupation VARCHAR(10),  
CUSTOMER_DATE_OF_BIRTH DATE,  
CONSTRAINT customer_custid_pk PRIMARY KEY (CUSTOMER_NUMBER));
```

```
CREATE TABLE branch_master(  
branch_id VARCHAR(6),  
branch_name VARCHAR(30),  
branch_city VARCHAR(30),  
CONSTRAINT branch_bid_pk PRIMARY KEY (branch_id));
```

```
CREATE TABLE account_master
(account_number VARCHAR(255),
customer_number VARCHAR(255),
branch_id VARCHAR(255),
opening_balance INT(20),
account_opening_date DATE,
account_type VARCHAR(10),
account_status VARCHAR(10),
PRIMARY KEY (account_number),
FOREIGN KEY (customer_number) references customer_master(customer_number),
FOREIGN KEY (branch_id) references branch_master(branch_id));
```

```
CREATE TABLE transaction_details(
transaction_number VARCHAR(6),
account_number VARCHAR(6),
date_of_transaction DATE,
medium_of_transaction VARCHAR(20),
transaction_type VARCHAR(20),
transaction_amount INT(7),
CONSTRAINT transaction_details_tnumber_pk PRIMARY KEY (transaction_number),
CONSTRAINT transaction_details_acnumber_fk FOREIGN KEY (account_number)
REFERENCES account_master (account_number));
```

```
CREATE TABLE loan_details
(customer_number varchar(255),
branch_id varchar(255),
loan_amount bigint(20),
foreign key(customer_number) references customer_master(customer_number));
```

```

insert into customer_master values('C00001', 'RAMESH', 'CHANDRA', 'SHARMA', 'DELHI',
    '9543198345', 'SERVICE', '1976-12-06');

insert into customer_master values('C00002', 'AVINASH', 'SUNDER', 'MINHA', 'DELHI',
    '9876532109', 'SERVICE', '1974-10-16');

insert into customer_master values('C00003', 'RAHUL', 'NULL', 'RASTOGI', 'DELHI',
    '9765178901', 'STUDENT', '1981-09-26');

insert into customer_master values('C00004', 'PARUL', 'NULL', 'GANDHI', 'DELHI',
    '9876532109', 'HOUSEWIFE', '1976-11-03');

insert into customer_master values('C00005', 'NAVEEN', 'CHANDRA', 'AEDEKAR',
    'MUMBAI', '8976523190', 'SERVICE', '1976-09-19');

insert into customer_master values('C00006', 'CHITRESH', 'NULL', 'BARWE', 'MUMBAI',
    '7651298321', 'STUDENT', '1992-11-06');

insert into customer_master values('C00007', 'AMIT', 'KUMAR', 'BORKAR', 'MUMBAI',
    '9875189761', 'STUDENT', '1981-09-06');

insert into customer_master values('C00008', 'NISHA', 'NULL', 'DAMLE', 'MUMBAI',
    '7954198761', 'SERVICE', '1975-12-03');

insert into customer_master values('C00009', 'ABHISHEK', 'NULL', 'DUTTA', 'KOLKATA',
    '9856198761', 'SERVICE', '1973-05-22');

insert into customer_master values('C00010', 'SHANKAR', 'NULL', 'NAIR', 'CHENNAI', '8765489076',
    'SERVICE', '1976-07-12');


insert into branch_master values('B00001', 'ASAF ALI ROAD', 'DELHI');

insert into branch_master values('B00002', 'NEW DELHI MAIN BRANCH', 'DELHI');

insert into branch_master values('B00003', 'DELHI CANTT', 'DELHI');

insert into branch_master values('B00004', 'JASOLA', 'DELHI');

insert into branch_master values('B00005', 'MAHIM', 'MUMBAI');

insert into branch_master values('B00006', 'VILE PARLE', 'MUMBAI');

insert into branch_master values('B00007', 'MANDVI', 'MUMBAI');

insert into branch_master values('B00008', 'JADAVPUR', 'KOLKATA');

insert into branch_master values('B00009', 'KODAMBAKKAM', 'CHENNAI');

```

```

insert into account_master values('A00001','C00001','B00001',1000    ,'2012-12-15', 'SAVING',
    'ACTIVE');

insert into account_master values('A00002'    ,'C00002','B00001',1000,'2012-06-12'    ,'SAVING',
    'ACTIVE');

insert into account_master values('A00003'    ,'C00003',    'B00002',    1000    ,'2012-05-17'
    ,'SAVING',    'ACTIVE');

insert into account_master values('A00004'    ,'C00002',    'B00005',    1000    ,'2013-01-27'
    ,'SAVING    ','ACTIVE');

insert into account_master values('A00005'    ,'C00006',    'B00006',    1000    ,'2012-12-17'
    ,'SAVING','ACTIVE');

insert into account_master values('A00006'    ,'C00007',    'B00007',    1000    ,'2010-08-12'
    ,'SAVING    ','SUSPENDED');

insert into account_master values('A00007'    ,'C00007',    'B00001',    1000    ,'2012-10-02'
    ,'SAVING    ','ACTIVE');

insert into account_master values('A00008'    ,'C00001','B00003',    1000    ,'2009-11-09'
    ,'SAVING    ','TERMINATED');

insert into account_master values('A00009'    ,'C00003',    'B00007',    1000    ,'2008-11-30'
    ,'SAVING',    'TERMINATED');

insert into account_master values('A00010'    ,'C00004',    'B00002',    1000    ,'2013-03-01'
    ,'SAVING',    'ACTIVE');


insert into transaction_details values('T00001', 'A00001',    '2013-01-01', 'CHEQUE',
    'DEPOSIT',    2000);

insert into transaction_details values('T00002' , 'A00001'    ,'2013-02-01'    ,'CASH'
    ,'WITHDRAWAL',    1000);

insert into transaction_details values('T00003', 'A00002'    ,'2013-01-01', 'CASH' , 'DEPOSIT',
    2000);

insert into transaction_details values('T00004', 'A00002',    '2013-02-01', 'CASH'    , 'DEPOSIT',
    3000);

insert into transaction_details values('T00005', 'A00007',    '2013-01-11', 'CASH' , 'DEPOSIT',
    7000);

insert into transaction_details values('T00006', 'A00007',    '2013-01-13', 'CASH' , 'DEPOSIT',
    9000);

insert into transaction_details values('T00007', 'A00001',    '2013-03-13', 'CASH' , 'DEPOSIT'
    ,4000);

```

```
insert into transaction_details values('T00008', 'A00001', '2013-03-14', 'CHEQUE'
, 'DEPOSIT' , 3000);

insert into transaction_details values('T00009', 'A00001', '2013-03-21', 'CASH'
, 'WITHDRAWAL' , 9000);

insert into transaction_details values('T00010', 'A00001', '2013-03-22', 'CASH'
, 'WITHDRAWAL' , 2000);

insert into transaction_details values('T00011', 'A00002', '2013-03-25', 'CASH'
, 'WITHDRAWAL' , 7000);

insert into transaction_details values('T00012', 'A00007', '2013-03-26', 'CASH'
, 'WITHDRAWAL' , 2000);


insert into Loan_details values('C00001', 'B00001', 100000);
insert into Loan_details values('C00002', 'B00002', 200000);
insert into Loan_details values('C00009', 'B00008', 400000);
insert into Loan_details values('C00010', 'B00009', 500000);
insert into Loan_details values('C00001', 'B00003', 600000);
insert into Loan_details values('C00002', 'B00001', 600000);
```

The diagram illustrates the following tables and their attributes:

- branch_master**:
 - branch_id VARCHAR(6) (Primary Key)
 - branch_name VARCHAR(30)
 - branch_city VARCHAR(30)
- loan_details**:
 - customer_number VARCHAR(6) (Foreign Key)
 - branch_id VARCHAR(6) (Foreign Key)
 - loan_amount INT(7)
- customer_master**:
 - customer_number VARCHAR(6) (Primary Key)
 - firstname VARCHAR(30)
 - lastname VARCHAR(30)
 - customer_city VARCHAR(15)
 - customer_contact_no VARCHAR(10)
 - customer_date_of_birth DATE
- account_master**:
 - account_number VARCHAR(6) (Primary Key)
 - customer_number VARCHAR(6) (Foreign Key)
 - branch_id VARCHAR(6) (Foreign Key)
 - opening_balance INT(7)
 - account_opening_date DATE
 - account_type VARCHAR(10)
 - account_status VARCHAR(10)
- transaction_details**:
 - transaction_number VARCHAR(6) (Primary Key)
 - account_number VARCHAR(6) (Foreign Key)
 - date_of_transaction DATE
 - medium_of_transaction VARCHAR(20)
 - transaction_type VARCHAR(20)
 - transaction_amount INT(7)

Relationships:

- branch_master** to **loan_details**: One-to-many relationship on branch_id.
- customer_master** to **loan_details**: One-to-many relationship on customer_number.
- customer_master** to **account_master**: One-to-many relationship on customer_number.
- branch_master** to **account_master**: One-to-many relationship on branch_id.
- account_master** to **transaction_details**: One-to-many relationship on account_number.

[illegible]

ACCOUNT MASTER

| account_number | customer_number | branch_id | opening_balance | account_opening_date | account_type | account_status |
|----------------|-----------------|-----------|-----------------|----------------------|--------------|----------------|
| A00001 | C00001 | B00001 | 1000 | 2012-12-15 | SAVING | ACTIVE |
| A00002 | C00002 | B00001 | 1000 | 2012-06-12 | SAVING | ACTIVE |
| A00003 | C00003 | B00002 | 1000 | 2012-05-17 | SAVING | ACTIVE |
| A00004 | C00002 | B00005 | 1000 | 2013-01-27 | SAVING | ACTIVE |
| A00005 | C00006 | B00006 | 1000 | 2012-12-17 | SAVING | ACTIVE |
| A00006 | C00007 | B00007 | 1000 | 2010-08-12 | SAVING | SUSPENDED |
| A00007 | C00007 | B00001 | 1000 | 2012-10-02 | SAVING | ACTIVE |
| A00008 | C00001 | B00003 | 1000 | 2009-11-09 | SAVING | TERMINATED |
| A00009 | C00003 | B00007 | 1000 | 2008-11-30 | SAVING | TERMINATED |
| A00010 | C00004 | B00002 | 1000 | 2013-03-01 | SAVING | ACTIVE |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

BRANCH MASTER

| branch_id | branch_name | branch_city |
|-----------|-----------------------|-------------|
| B00001 | ASAF ALI ROAD | DELHI |
| B00002 | NEW DELHI MAIN BRANCH | DELHI |
| B00003 | DELHI CANTT | DELHI |
| B00004 | JASOLA | DELHI |
| B00005 | MAHIM | MUMBAI |
| B00006 | VILE PARLE | MUMBAI |
| B00007 | MANDVI | MUMBAI |
| B00008 | JADAVPUR | KOLKATA |
| B00009 | KODAMBAKKAM | CHENNAI |
| NULL | NULL | NULL |

LOAN DETAILS

| customer_number | branch_id | loan_amount |
|-----------------|-----------|-------------|
| C00001 | B00001 | 100000 |
| C00002 | B00002 | 200000 |
| C00009 | B00008 | 400000 |
| C00010 | B00009 | 500000 |
| C00001 | B00003 | 600000 |
| C00002 | B00001 | 600000 |

TRANSACTION DETAILS

| transaction_number | account_number | date_of_transaction | medium_of_transaction | transaction_type | transaction_amount |
|--------------------|----------------|---------------------|-----------------------|------------------|--------------------|
| T00001 | A00001 | 2013-01-01 | CHEQUE | DEPOSIT | 2000 |
| T00002 | A00001 | 2013-02-01 | CASH | WITHDRAWAL | 1000 |
| T00003 | A00002 | 2013-01-01 | CASH | DEPOSIT | 2000 |
| T00004 | A00002 | 2013-02-01 | CASH | DEPOSIT | 3000 |
| T00005 | A00007 | 2013-01-11 | CASH | DEPOSIT | 7000 |
| T00006 | A00007 | 2013-01-13 | CASH | DEPOSIT | 9000 |
| T00007 | A00001 | 2013-03-13 | CASH | DEPOSIT | 4000 |
| T00008 | A00001 | 2013-03-14 | CHEQUE | DEPOSIT | 3000 |
| T00009 | A00001 | 2013-03-21 | CASH | WITHDRAWAL | 9000 |
| T00010 | A00001 | 2013-03-22 | CASH | WITHDRAWAL | 2000 |
| T00011 | A00002 | 2013-03-25 | CASH | WITHDRAWAL | 7000 |
| T00012 | A00007 | 2013-03-26 | CASH | WITHDRAWAL | 2000 |
| NULL | NULL | NULL | NULL | NULL | NULL |

QUERIES

1. Write a query to display account number, customer's number, customer's firstname, lastname, account opening date. Display the records sorted in ascending order based on account number.

```
SELECT a.account_number,c.customer_number,c.firstname,c.lastname,a.account_number
FROM      customer_master      c      JOIN      account_master      a      ON
c.customer_number=a.customer_number
ORDER BY a.account_number;
```


| account_number | customer_number | firstname | lastname | account_opening_date |
|----------------|-----------------|-----------|----------|----------------------|
| A00001 | C00001 | RAMESH | SHARMA | 2012-12-15 |
| A00002 | C00002 | AVINASH | MINHA | 2012-06-12 |
| A00003 | C00003 | RAHUL | RASTOGI | 2012-05-17 |
| A00004 | C00002 | AVINASH | MINHA | 2013-01-27 |
| A00005 | C00006 | CHITRESH | BARWE | 2012-12-17 |
| A00006 | C00007 | AMIT | BORKAR | 2010-08-12 |
| A00007 | C00007 | AMIT | BORKAR | 2012-10-02 |
| A00008 | C00001 | RAMESH | SHARMA | 2009-11-09 |
| A00009 | C00003 | RAHUL | RASTOGI | 2008-11-30 |
| A00010 | C00004 | PARUL | GANDHI | 2013-03-01 |

2. Write a query to display the number of customer's from Delhi. Give the count an alias name of Cust_Count.

```
SELECT count(customer_number) Cust_Count FROM customer_master WHERE customer_city='Delhi';
```

| cust_count |
|------------|
| 4 |

3. Write a query to display the customer number, customer firstname, account number for the customer's whose accounts were created after 15th of any month. Display the records sorted in ascending order based on customer number and then by account number.

```
SELECT c.customer_number,c.firstname,a.account_number FROM account_master a join
customer_master c ON c.customer_number=a.customer_number WHERE
day(a.account_opening_date)>'15' ORDER BY c.customer_number,a.account_number;
```

| customer_number | firstname | account_number |
|-----------------|-----------|----------------|
| C00002 | AVINASH | A00004 |
| C00003 | RAHUL | A00003 |
| C00003 | RAHUL | A00009 |
| C00006 | CHITRESH | A00005 |

4. Write a query to display customer number, customer's first name, account number where the account status is terminated. Display the records sorted in ascending order based on customer number and then by account number.

```
SELECT c.customer_number,c.firstname,a.account_number
FROM account_master a JOIN customer_master c
ON c.customer_number=a.customer_number
WHERE a.account_status='Terminated'
ORDER BY c.customer_number,a.account_number;
```

| customer_number | firstname | account_number |
|-----------------|-----------|----------------|
| C00001 | RAMESH | A00008 |
| C00003 | RAHUL | A00009 |

5. Write a query to display the total number of withdrawals and total number of deposits being done by customer whose customer number ends with 001. The query should display transaction type and the number of transactions. Give an alias name as Trans_Count for number of transactions. Display the records sorted in ascending order based on transaction type.

```
SELECT transaction_type,count(transaction_number) Trans_Count
FROM account_master am JOIN transaction_details td
ON am.account_number=td.account_number
WHERE customer_number like '%001'
GROUP BY transaction_type
ORDER BY transaction_type;
```

| transaction_type | Trans_count |
|------------------|-------------|
| DEPOSIT | 3 |
| WITHDRAWAL | 3 |

6. Write a query to display the number of customers who have registration but no account in the bank. Give the alias name as Count_Customer for number of customers.

```
SELECT count(customer_number) Count_Customer FROM customer_master
WHERE customer_number NOT IN (SELECT customer_number FROM account_master);
```

| Count_customer |
|----------------|
| 4 |

7. Write a query to display account number and total amount deposited by each account holder (Including the opening balance). Give the total amount deposited an alias name of Deposit_Amount. Display the records in sorted order based on account number.

```
SELECT a.account_number,a.opening_balance+sum(t.transaction_amount)
FROM account_master a JOIN transaction_details t ON a.account_number=t.account_number
WHERE t.transaction_type='Deposit' GROUP BY t.account_number;
```

| account_number | Deposit_Amount |
|----------------|----------------|
| A00001 | 10000 |
| A00002 | 6000 |
| A00007 | 17000 |

8. Write a query to display the number of accounts opened in each city .The Query should display Branch City and number of accounts as No_of_Accounts.For the branch city where we don't have any accounts opened display 0. Display the records in sorted order based on branch city.

```
SELECT branch.branch_city, count(account.account_number) No_of_Accounts
FROM branch_master LEFT JOIN account_master
ON account.branch_id=branch.branch_id
GROUP BY branch.branch_city ORDER BY branch_city;
```

| branch_city | No_of_accounts |
|-------------|----------------|
| CHENNAI | 0 |
| DELHI | 6 |
| KOLKATA | 0 |
| MUMBAI | 4 |

9. Write a query to display the firstname of the customers who have more than 1 account. Display the records in sorted order based on firstname.

```
SELECT c.firstname FROM
customer_master c JOIN account_master a ON a.customer_number=c.customer_number
GROUP BY a.customer_number HAVING count(a.account_number)>1;
```

| firstname |
|-----------|
| AMIT |
| AVINASH |
| RAHUL |
| RAMESH |

10. Write a query to display the customer number, customer firstname, customer lastname who has taken loan from more than 1 branch. Display the records sorted in order based on customer number.

```
SELECT c.customer_number,c.firstname,c.lastname FROM
customer_master c JOIN loan_details l ON c.customer_number=l.customer_number
GROUP BY l.customer_number HAVING count(l.branch_id)>1
ORDER BY c.customer_number;
```

| customer_number | firstname | lastname |
|-----------------|-----------|----------|
| C00001 | RAMESH | SHARMA |
| C00002 | AVINASH | MINHA |

11. Write a query to display the customer's number, customer's firstname, customer's city and branch city where the city of the customer and city of the branch is different. Display the records sorted in ascending order based on customer number.

```
SELECT c.customer_number,c.firstname,c.customer_city,b.branch_city FROM
Customer_master c JOIN Account_master a ON c.customer_number=a.customer_number
JOIN Branch_master b ON b.branch_id=a.branch_id
WHERE b.branch_city<>c.customer_city
ORDER BY c.customer_number;
```

| customer_number | firstname | customer_city | branch_city |
|-----------------|-----------|---------------|-------------|
| C00002 | AVINASH | DELHI | MUMBAI |
| C00003 | RAHUL | DELHI | MUMBAI |
| C00007 | AMIT | MUMBAI | DELHI |

12. Write a query to display the number of clients who have asked for loans but they don't have any account in the bank though they are registered customers. Give the count an alias name of Count.

```
SELECT count(c.customer_number)Count FROM customer_master c JOIN loan_details l
ON c.customer_number=l.customer_number
WHERE c.customer_number NOT IN (SELECT customer_number FROM account_master);
```

| Count |
|-------|
| 2 |

13. Write a query to display the account number who has done the highest transaction. For example the account A00023 has done 5 transactions i.e. suppose 3 withdrawal and 2 deposits. Whereas the account A00024 has done 3 transactions i.e. suppose 2 withdrawals and 1 deposit. So account number of A00023 should be displayed. In case of multiple records, display the records sorted in ascending order based on account number.

```
SELECT account_number FROM transaction_details
GROUP BY account_number
HAVING count(transaction_number)>=ALL
(SELECT count(transaction_number) FROM transaction_details
GROUP BY account_number) ORDER BY account_number;
```

| account_number |
|----------------|
| A00001 |

14. Write a query to show the branch name,branch city where we have the maximum customers. For example the branch B00019 has 3 customers, B00020 has 7 and B00021 has 10. So branch id B00021 is having maximum customers. If B00021 is Koramangla branch Bangalore, Koramangla branch should be displayed along with city name Bangalore. In case of multiple records, display the records sorted in ascending order based on branch name.

```
SELECT b.branch_name,b.branch_city FROM
Branch_master b JOIN account a ON a.branch_id=b.branch_id
GROUP BY a.branch_id HAVING count(a.customer_number)>=ALL
(SELECT count(customer_number) FROM
Account_master GROUP BY branch_id)
ORDER BY b.branch_name;
```

| branch_name | branch_city |
|---------------|-------------|
| ASAF ALI ROAD | DELHI |

15. Write a query to display all those account number, deposit, withdrawal where withdrawal is more than deposit amount. Hint: Deposit should include opening balance as well. For example A00011 account opened with Opening Balance 1000 and A00011 deposited 2000 rupees on 2012-12-01 and 3000 rupees on 2012-12-02. The same account i.e A00011 withdrawn 3000 rupees on 2013-01-01 and 7000 rupees on 2013-01-03. So the total deposited amount is 6000 and total withdrawal amount is 10000. So withdrawal amount is more than deposited amount for account number A00011. Display the records sorted in ascending order based on account number.

```
SELECT td.account_number,
sum(CASE WHEN transaction_type='Deposit' THEN transaction_amount END)
+(SELECT opening_balance
FROM account_master where account_number=td.account_number) Deposit,
sum(CASE WHEN transaction_type='Withdrawal' THEN transaction_amount END) Withdrawal
FROM transaction_details td
GROUP BY td.account_number
HAVING Withdrawal > Deposit
ORDER BY td.account_number;
```

(or)

```
SELECT ifnull(t1.account_number,t2.account_number) account_number,
t2.d Deposit,ifnull(t1.w,0) Withdrawal FROM
(SELECT account_number,transaction_type,sum(transaction_amount) w from transaction_details
WHERE transaction_type='Withdrawal' GROUP BY account_number) t1
RIGHT JOIN
(SELECT a.account_number,a.opening_balance+sum(t.transaction_amount) d
FROM account_master a JOIN transaction_details t ON a.account_number=t.account_number
WHERE t.transaction_type='Deposit'GROUP BY t.account_number) t2
ON t1.account_number=t2.account_number
WHERE ifnull(t1.w,0)>t2.d
ORDER BY account_number;
```

| account_number | Deposit | Withdrawal |
|----------------|---------|------------|
| A00001 | 10000 | 12000 |
| A00002 | 6000 | 7000 |

16. Write a query to show the balance amount for account number that ends with 001. Note: Balance amount includes account opening balance also. Give alias name as Balance_Amount. For example A00015 is having an opening balance of 1000. A00015 has deposited 2000 on 2012-06-12 and deposited 3000 on 2012-07-13. The same account has drawn money of 500 on 2012-08-12 , 500 on 2012-09-15, 1000 on 2012-12-17. So balance amount is 4000 i.e (1000 (opening balance)+2000+3000) – (500+500+1000).

```
SELECT ifnull((SUM(CASE WHEN transaction_type='Deposit'
THEN transaction_amount END)) -
(SUM(CASE WHEN transaction_type='Withdrawal'
THEN transaction_amount END))+(select opening_balance
from account_master where account_number like '%001'),(SUM(CASE WHEN
transaction_type='Deposit'
THEN transaction_amount END))+(select opening_balance
from account_master where account_number like '%001')) AS Balance_Amount
FROM transaction_details where account_number like '%001';
```

(or)

```
SELECT ifnull(t1.account_number,t2.account_number) account_number,
t2.d-ifnull(t1.w,0) Balance_Amount FROM
(SELECT account_number,transaction_type,sum(transaction_amount) w from transaction_details
WHERE transaction_type='Withdrawal' GROUP BY account_number) t1
RIGHT JOIN
(SELECT a.account_number,a.opening_balance+sum(t.transaction_amount) d
FROM account a JOIN transaction_details t ON a.account_number=t.account_number
WHERE t.transaction_type='Deposit'GROUP BY t.account_number) t2
ON t1.account_number=t2.account_number
WHERE ifnull(t1.account_number,t2.account_number) LIKE '%001'
ORDER BY account_number;
```

| account_number | Balance_Amount |
|----------------|----------------|
| A00001 | -2000 |

17. Display the customer number, customer's first name, account number and number of transactions being made by the customers from each account. Give the alias name for number of transactions as

Count_Trans. Display the records sorted in ascending order based on customer number and then by account number.

```
SELECT c.customer_number,c.firstname,t.account_number, count(t.account_number) Count_Trans
FROM transaction_details t JOIN account_master a ON a.account_number=t.account_number
JOIN customer c ON c.customer_number=a.customer_number
GROUP BY t.account_number ORDER BY c.customer_number, a.account_number;
```

| customer_number | firstname | account_number | Count_Trans |
|-----------------|-----------|----------------|-------------|
| C00001 | RAMESH | A00001 | 6 |
| C00002 | AVINASH | A00002 | 3 |
| C00007 | AMIT | A00007 | 3 |

18. Write a query to display the customer's firstname who have multiple accounts (atleast 2 accounts). Display the records sorted in ascending order based on customer's firstname.

```
SELECT c.firstname FROM
Customer_master c JOIN account_master a ON c.customer_number=a.customer_number
GROUP BY a.customer_number HAVING count(a.account_number)>1
ORDER BY c.firstname;
```

| firstname |
|-----------|
| AMIT |
| AVINASH |
| RAHUL |
| RAMESH |

19. Write a query to display the customer number, firstname, lastname for those client where total loan amount taken is maximum and at least taken from 2 branches. For example the customer C00012 took a loan of 100000 from bank branch with id B00009 and C00012 Took a loan of 500000 from bank branch with id B00010. So total loan amount for customer C00012 is 600000. C00013 took a loan of 100000 from bank branch B00009 and 200000 from bank branch B00011. So total loan taken is 300000. So loan taken by C00012 is more then C00013.

```
SELECT Id.customer_number, firstname, lastname
FROM customer_master cm JOIN loan_details Id
ON cm.customer_number=Id.customer_number
GROUP BY customer_number
HAVING count(branch_id)>=2 AND sum(loan_amount)>=
```


ALL(SELECT sum(loan_amount) FROM loan GROUP BY customer_number);

| customer_number | firstname | lastname |
|-----------------|-----------|----------|
| C00002 | AVINASH | MINHA |

20. Write a query to display the customer's number, customer's firstname, branch id and loan amount for people who have taken loans. Display the records sorted in ascending order based on customer number and then by branch id and then by loan amount.

```
SELECT c.customer_number,c.firstname,l.branch_id,l.loan_amount FROM  
Customer_master c JOIN loan_details l ON c.customer_number=l.customer_number  
ORDER BY c.customer_number,l.branch_id,l.loan_amount;
```

| customer_number | firstname | branch_id | loan_amount |
|-----------------|-----------|-----------|-------------|
| C00001 | RAMESH | B00001 | 100000 |
| C00001 | RAMESH | B00003 | 600000 |
| C00002 | AVINASH | B00001 | 600000 |
| C00002 | AVINASH | B00002 | 200000 |
| C00009 | ABHISHEK | B00008 | 400000 |
| C00010 | SHANKAR | B00009 | 500000 |

21. Write a query to display city name and count of branches in that city. Give the count of branches an alias name of Count_Branch. Display the records sorted in ascending order based on city name.

```
SELECT branch_city,count(branch_id) Count_Branch FROM  
Branch_master GROUP BY branch_city  
ORDER BY branch_city;
```

| branch_city | Count_Branch |
|-------------|--------------|
| CHENNAI | 1 |
| DELHI | 4 |
| KOLKATA | 1 |
| MUMBAI | 3 |

22. Write a query to display account id, customer's firstname, customer's lastname for the customer's whose account is Active. Display the records sorted in ascending order based on account id /account number.

```
SELECT a.account_number,c.firstname,c.lastname FROM  
Customer_master c JOIN account_master a ON c.customer_number=a.customer_number and  
a.account_status='Active'
```

ORDER BY a.account_number;

| account_number | firstname | lastname |
|----------------|-----------|----------|
| A00001 | RAMESH | SHARMA |
| A00002 | AVINASH | MINHA |
| A00003 | RAHUL | RASTOGI |
| A00004 | AVINASH | MINHA |
| A00005 | CHITRESH | BARWE |
| A00007 | AMIT | BORKAR |
| A00010 | PARUL | GANDHI |

23. Write a query to display customer's number, first name and middle name. For the customers who don't have middle name, display their last name as middle name. Give the alias name as Middle_Name. Display the records sorted in ascending order based on customer number.

```
SELECT customer_number,firstname,ifnull(middlename,lastname) Middle_name FROM
```

```
Customer_master ORDER BY customer_number;
```

| customer_number | firstname | Middle_name |
|-----------------|-----------|-------------|
| C00001 | RAMESH | CHANDRA |
| C00002 | AVINASH | SUNDER |
| C00003 | RAHUL | NULL |
| C00004 | PARUL | NULL |
| C00005 | NAVEEN | CHANDRA |
| C00006 | CHITRESH | NULL |
| C00007 | AMIT | KUMAR |
| C00008 | NISHA | DAMLE |
| C00009 | ABHISHEK | DUTTA |
| C00010 | SHANKAR | NAIR |

24. Write a query to display the customer number , firstname, customer's date of birth . Display the records sorted in ascending order of date of birth year and within that sort by firstname in ascending order.

```
SELECT customer_number,firstname,customer_date_of_birth FROM
```

```
Customer_master ORDER BY year(customer_date_of_birth),customer_number;
```

| customer_number | firstname | customer_date_of_birth |
|-----------------|-----------|------------------------|
| C00009 | ABHISHEK | 1973-05-22 |
| C00002 | AVINASH | 1974-10-16 |
| C00008 | NISHA | 1975-12-03 |
| C00001 | RAMESH | 1976-12-06 |
| C00004 | PARUL | 1976-11-03 |
| C00005 | NAVEEN | 1976-09-19 |
| C00010 | SHANKAR | 1976-07-12 |
| C00003 | RAHUL | 1981-09-26 |
| C00007 | AMIT | 1981-09-06 |
| C00006 | CHITRESH | 1992-11-06 |

25. Write a query to display the customers firstname, city and account number whose occupation are not into Business, Service or Student. Display the records sorted in ascending order based on customer first name and then by account number.

```
SELECT c.firstname,c.customer_city,a.account_number FROM
Customer_master c JOIN account_master a ON a.customer_number=c.customer_number
WHERE c.occupation NOT IN ('Service','Student','Business')
ORDER BY c.firstname,a.account_number;
```

| firstname | customer_city | account_number |
|-----------|---------------|----------------|
| PARUL | DELHI | A00010 |

AIRLINES

```
create database light;
use light;
```

```
CREATE TABLE credit_card_details
```

```
(
profile_id VARCHAR(10) NOT NULL,
```

```
card_number BIGINT,  
card_type VARCHAR(45),  
expiration_month INT,  
expiration_year INT  
);
```

```
CREATE TABLE air_passenger_profile  
(  
profile_id VARCHAR(10) NOT NULL ,  
password VARCHAR(45) NULL ,  
first_name VARCHAR(45) NULL ,  
last_name VARCHAR(45) NULL ,  
address VARCHAR(45) NULL ,  
mobile_number BIGINT NULL ,  
email_id VARCHAR(45) NULL  
);
```

```
CREATE TABLE air_ticket_info  
(  
ticket_id VARCHAR(45) NOT NULL ,  
profile_id VARCHAR(10) NULL ,  
flight_id VARCHAR(45) NULL ,  
flight_departure_date DATE NULL ,  
status VARCHAR(45) NULL  
);
```

```
CREATE TABLE air_flight_details  
(  
flight_id VARCHAR(45) NOT NULL ,
```

```
flight_departure_date DATE NULL ,  
price DECIMAL(10,2) NULL ,  
available_seats INT NULL  
);
```

```
CREATE TABLEair_flight  
(  
flight_id VARCHAR(45) NOT NULL ,  
airline_id VARCHAR(45) NULL ,  
airline_name VARCHAR(45) NULL ,  
from_location VARCHAR(45) NULL ,  
to_location VARCHAR(45) NULL ,  
departure_time TIME NULL ,  
arrival_time TIME NULL ,  
duration TIME NULL ,  
total_seats INT NULL  
);
```

```
INSERT INTO air_credit_card_details VALUES  
(1, 622098761234, 'debit', 5, 2013),  
(2, 652362563625, 'credit', 1, 2013),  
(1, 765432345678, 'credit', 2, 2013),  
(3, 654378561234, 'debit', 6, 2013),  
(4, 625417895623, 'debit', 2, 2013),  
(5, 865478956325, 'debit', 3, 2013),  
(6, 789563521457, 'credit', 4, 2013),  
(2, 543267895432, 'credit', 8, 2013),  
(1, 256369856321, 'debit', 1, 2013);
```

INSERT INTO air_flight VALUES

```
(3173, 'MH370',      'abc', 'hyderabad',  'chennai',      '06:30:00',    '07:15:00',
      '0:45:00',    100),
(3178, 'MH17',      'def', 'chennai',      'hyderabad',    '08:00:00',    '09:00:00',
      '1:00:00',    200),
(3172, 'AR342',      'fgh', 'kolkata',      'chennai',      '11:30:00',    '13:00:00',
      '1:30:00',    100),
(3071, 'JT564', 'jkl', 'chennai',      'delhi', '08:00:00',    '10:00:00',    '2:00:00',    100),
(3170, 'DT345',      'xyz', 'delhi', 'kolkata',      '21:00:00',    '22:30:00',    '1:30:00',
      100),
(3175, 'MJ654',      'abc', 'chennai',      'hyderabad',    '15:00:00',    '16:00:00',
      '1:00:00',    200),
(3176, 'MH370',      'def', 'kochi', 'chennai',      '18:00:00',    '19:05:00',    '1:05:00',
      100),
(3177, 'MH45',      'fgh', 'delhi', 'kochi', '19:00:00',    '21:00:00',    '2:00:00',    200),
(3174, 'MH321',      'xyz', 'kolkata',      'delhi', '0:00:00',     '2:00:00',     '2:00:00',
      100),
(3179, 'JT435', 'abc', 'chennai',      'kolkata',      '14:00:00',    '15:00:00',    '1:00:00',
      100),
(3180, 'JT456', 'ijk', 'kolkata',      'kochi', '5:00:00',     '5:45:00',     '0:45:00',    200);
```

INSERT INTO air_flight_details VALUES

```
(3170, '2013-02-14', 1000, 10),
(3171, '2013-03-15', 5000, 0),
(3172, '2013-02-05', 3000, 32),
(3173, '2013-04-07', 2000, 12),
(3174, '2013-04-05', 3800, 3),
(3175, '2013-05-25', 3500, 10),
(3176, '2013-03-14', 8000, 2),
```

```
(3177, '2013-06-15', 1500, 0),  
(3178, '2013-05-06', 3000, 5),  
(3179, '2013-04-03', 4000, 15),  
(3180, '2013-04-02', 3000, 14);
```

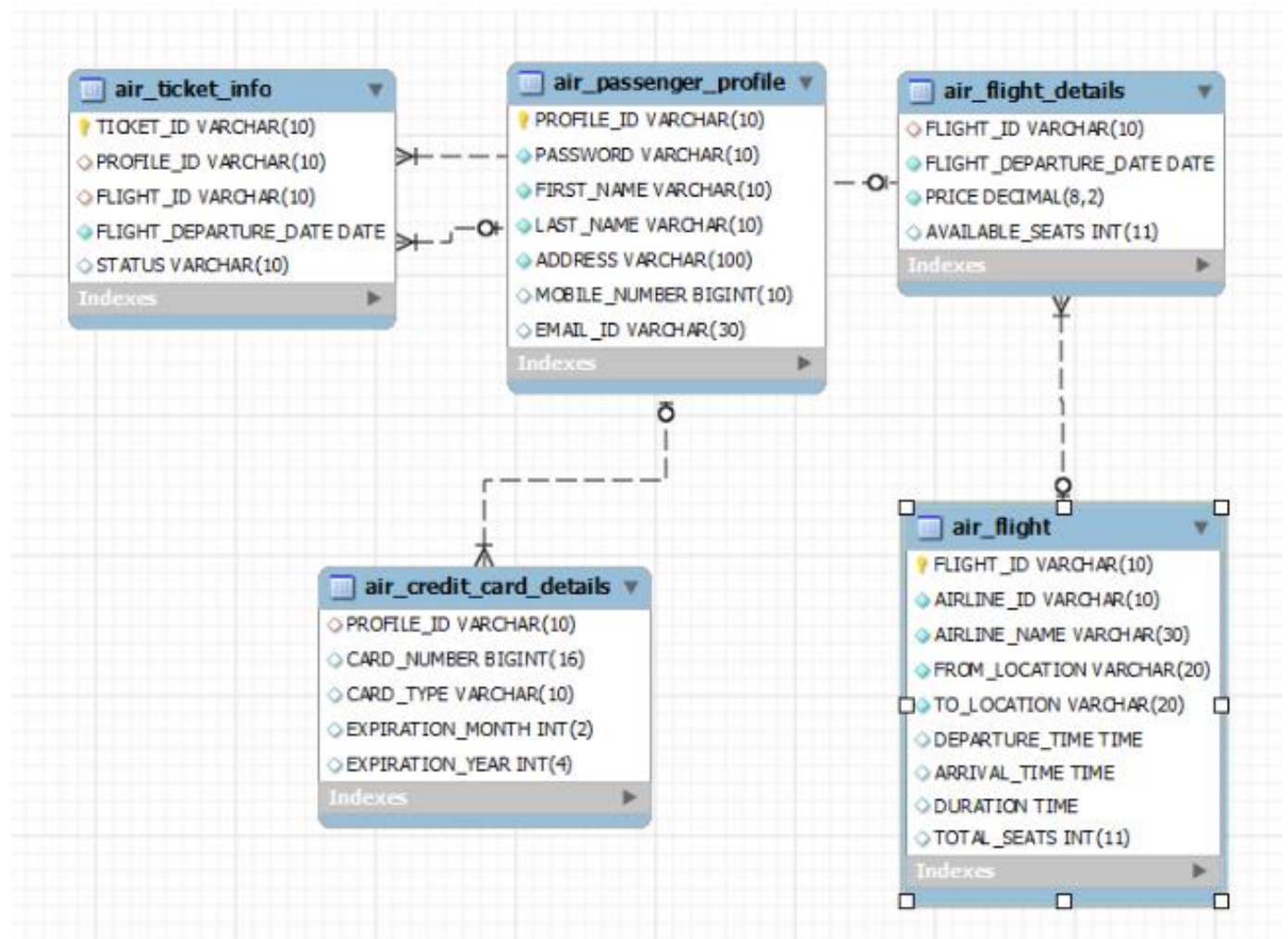
```
INSERT INTO air_ticket_info VALUES
```

```
(1, 1, 3178, '2013-05-06', 'delayed'),  
(2, 5, 3179, '2013-04-03', 'on time'),  
(2, 4, 3180, '2013-04-02', 'on time'),  
(1, 2, 3177, '2013-06-15', 'on time'),  
(1, 3, 3176, '2013-03-14', 'on time'),  
(3, 1, 3171, '2013-03-15', 'on time'),  
(4, 4, 3172, '2013-02-06', 'delayed'),  
(5, 2, 3178, '2013-06-05', 'on time'),  
(4, 3, 3171, '2013-03-15', 'on time'),  
(5, 1, 3175, '2013-05-25', 'on time'),  
(6, 3, 3177, '2013-06-15', 'on time');
```

```
INSERT INTO air_passenger_profile VALUES
```

```
(1, 'godbless', 'John', 'Stuart', 'Street 21, Near Bus Stop-Hyderabad-432126',  
9865263251, 'john@gmail.com'),  
(2, 'heyyaa', 'Robert', 'Clive', 'Sector 3, Technopolis-Kolkata-700102',  
9733015875, 'robert@yahoo.com'),  
(3, 'hello123', 'Raj', 'Sharma', 'House No. 3, Anna Nagar-Kochi-452314',  
9775470232, 'raj3452@hotmail.com'),  
(4, 'yesboss', 'Sanjay', 'Mittal', '21 Cauunaught Place-Delhi-144985',  
9856856321, 'sanjay@yahoo.com');
```

(5, 'imhere', 'Tony', 'Stark', '51A, Greems Lane-Chennai-144587',
9832015785, 'tony@gmail.com');



AIR TICKET INFO

| ticket_id | profile_id | flight_id | flight_departure_date | status |
|-----------|------------|-----------|-----------------------|---------|
| 1 | 1 | 3178 | 2013-05-06 | delayed |
| 2 | 5 | 3179 | 2013-04-03 | on time |
| 2 | 4 | 3180 | 2013-04-02 | on time |
| 1 | 2 | 3177 | 2013-06-15 | on time |
| 1 | 3 | 3176 | 2013-03-14 | on time |
| 3 | 1 | 3171 | 2013-03-15 | on time |
| 4 | 4 | 3172 | 2013-02-06 | delayed |
| 5 | 2 | 3178 | 2013-06-05 | on time |
| 4 | 3 | 3171 | 2013-03-15 | on time |
| 5 | 1 | 3175 | 2013-05-25 | on time |
| 6 | 3 | 3177 | 2013-06-15 | on time |

AIR PASSENGER DETAILS

| profile_id | password | first_name | last_name | address | mobile_number | email_id |
|------------|----------|------------|-----------|---|---------------|--------------------|
| 1 | godbless | John | Stuart | Street 21, Near Bus Stop-Hyderabad-432126 | 9865263251 | john@gmail.com |
| 2 | heyaa | Robert | Clive | Sector 3, Technopolis-Kolkata-700102 | 9733015875 | robert@yahoo.com |
| 3 | hello123 | Raj | Shama | House No. 3, Anna Nagar-Kochi-452314 | 9775470232 | raj3452@hotmail... |
| 4 | yesboss | Sanjay | Mittal | 21 Cauunaught Place-Delhi-144985 | 9856856321 | sanjay@yahoo.c... |
| 5 | imhere | Tony | Stark | 51A, Greams Lane-Chennai-144587 | 9832015785 | tony@gmail.com |

AIR FLIGHT DETAILS

| flight_id | flight_departure_date | price | available_seats |
|-----------|-----------------------|---------|-----------------|
| 3170 | 2013-02-14 | 1000.00 | 10 |
| 3171 | 2013-03-15 | 5000.00 | 0 |
| 3172 | 2013-02-05 | 3000.00 | 32 |
| 3173 | 2013-04-07 | 2000.00 | 12 |
| 3174 | 2013-04-05 | 3800.00 | 3 |
| 3175 | 2013-05-25 | 3500.00 | 10 |
| 3176 | 2013-03-14 | 8000.00 | 2 |
| 3177 | 2013-06-15 | 1500.00 | 0 |
| 3178 | 2013-05-06 | 3000.00 | 5 |
| 3179 | 2013-04-03 | 4000.00 | 15 |
| 3180 | 2013-04-02 | 3000.00 | 14 |

AIR CREDIT CARD DETAILS

| profile_id | card_number | card_type | expiration_month | expiration_year |
|------------|--------------|-----------|------------------|-----------------|
| 1 | 622098761234 | debit | 5 | 2013 |
| 2 | 652362563625 | credit | 1 | 2013 |
| 1 | 765432345678 | credit | 2 | 2013 |
| 3 | 654378561234 | debit | 6 | 2013 |
| 4 | 625417895623 | debit | 2 | 2013 |
| 5 | 865478956325 | debit | 3 | 2013 |
| 6 | 789563521457 | credit | 4 | 2013 |
| 2 | 543267895432 | credit | 8 | 2013 |
| 1 | 256369856321 | debit | 1 | 2013 |

AIR FLIGHT

| flight_id | airline_id | airline_name | from_location | to_location | departure_time | arrival_time | duration | total_seats |
|-----------|------------|--------------|---------------|-------------|----------------|--------------|----------|-------------|
| 3170 | DT345 | xyz | delhi | kolkata | 21:00:00 | 22:30:00 | 01:30:00 | 100 |
| 3171 | JT564 | jkl | chennai | delhi | 08:00:00 | 10:00:00 | 02:00:00 | 100 |
| 3172 | AR342 | fgh | kolkata | chennai | 11:30:00 | 13:00:00 | 01:30:00 | 100 |
| 3173 | MH370 | abc | hyderabad | chennai | 06:30:00 | 07:15:00 | 00:45:00 | 100 |
| 3174 | MH321 | xyz | kolkata | delhi | 00:00:00 | 02:00:00 | 02:00:00 | 100 |
| 3175 | MJ654 | abc | chennai | hyderabad | 15:00:00 | 16:00:00 | 01:00:00 | 200 |
| 3176 | MH370 | def | kochi | chennai | 18:00:00 | 19:05:00 | 01:05:00 | 100 |
| 3177 | MH45 | fgh | delhi | kochi | 19:00:00 | 21:00:00 | 02:00:00 | 200 |
| 3178 | MH17 | def | chennai | hyderabad | 08:00:00 | 09:00:00 | 01:00:00 | 200 |
| 3179 | JT435 | abc | chennai | kolkata | 14:00:00 | 15:00:00 | 01:00:00 | 100 |
| 3180 | JT456 | ijk | kolkata | kochi | 05:00:00 | 05:45:00 | 00:45:00 | 200 |

QUERIES

1. Write a query to display the average monthly ticket cost for each flight in ABC Airlines. The query should display the Flight_Id,From_location,To_Location,Month Name as "Month_Name" and average price as "Average_Price". Display the records sorted in ascending order based on flight id and then by Month Name.

```
SELECT f.flight_id,f.from_location,f.to_location,
monthname(af.flight_departure_date) Month_Name,
AVG(price) Average_Price FROM air_flight f JOIN air_flight_details af
ON f.flight_id = af.flight_id WHERE f.airline_name = 'abc'
```

GROUP BY f.flight_id,f.from_location,f.to_location,Month_Name

ORDER BY f.flight_id, Month_Name;

| flight_id | from_location | to_location | Month_Name | Average_Price |
|-----------|---------------|-------------|------------|---------------|
| 3173 | hyderabad | chennai | April | 2000.000000 |
| 3175 | chennai | hyderabad | May | 3500.000000 |
| 3179 | chennai | kolkata | April | 4000.000000 |

2. Write a query to display the number of flight services between locations in a month. The Query should display From_Location, To_Location, Month as "Month_Name" and number of flight services as "No_of_Services". Hint: The Number of Services can be calculated from the number of scheduled departure dates of a flight. The records should be displayed in ascending order based on From_Location and then by To_Location and then by month name.

```
SELECT f.from_location,f.to_location,
monthname(af.flight_departure_date) Month_Name,
count(af.flight_departure_date) No_of_Services
FROM air_flight f JOIN air_flight_details af
ON f.flight_id = af.flight_id
GROUP BY f.from_location,f.to_location,Month_Name
ORDER BY f.from_location,f.to_Location,Month_Name;
```

| from_location | to_location | Month_Name | No_of_Services |
|---------------|-------------|------------|----------------|
| chennai | delhi | March | 1 |
| chennai | hyderabad | May | 2 |
| chennai | kolkata | April | 1 |
| delhi | kochi | June | 1 |
| delhi | kolkata | February | 1 |
| hyderabad | chennai | April | 1 |
| kochi | chennai | March | 1 |
| kolkata | chennai | February | 1 |
| kolkata | delhi | April | 1 |
| kolkata | kochi | April | 1 |

3. Write a query to display the customer(s) who has/have booked least number of tickets in ABC Airlines. The Query should display profile_id, customer's first_name, Address and Number of tickets

booked as “No_of_Tickets” Display the records sorted in ascending order based on customer's first name.

```
SELECT ap.profile_id,ap.first_name,ap.address,count(ati.ticket_id) No_of_Tickets FROM
air_passenger_profile ap JOIN air_ticket_info ati ON ap.profile_id=ati.profile_id
JOIN air_flight af ON af.flight_id=ati.flight_id and af.airline_name='abc'
GROUP BY ap.profile_id,ap.first_name,ap.address HAVING count(ati.ticket_id)<=ALL
(SELECT count(ticket_id)
FROM air_ticket_info GROUP BY profile_id)
ORDER BY ap.first_name;
```

| profile_id | first_name | address | No_of_Tickets |
|------------|------------|---|---------------|
| 1 | John | Street 21, Near Bus Stop-Hyderabad-432126 | 1 |
| 5 | Tony | 51A, Greams Lane-Chennai-144587 | 1 |

4. Write a query to display the number of tickets booked from Chennai to Hyderabad. The Query should display passenger profile_id,first_name,last_name, Flight_Id , Departure_Date and number of tickets booked as “No_of_Tickets”.Display the records sorted in ascending order based on profile id and then by flight id and then by departure date.

```
SELECT ap.profile_id,ap.first_name,ap.last_name,af.flight_id,ati.flight_departure_date,
count(ati.profile_id) No_of_Tickets FROM
air_ticket_info ati JOIN air_passenger_profile ap ON ap.profile_id=ati.profile_id
JOIN air_flight af ON af.flight_id=ati.flight_id
WHERE af.from_location='Chennai' and af.to_location='Hyderabad'
GROUP BY ati.flight_id,ati.profile_id
ORDER BY ap.profile_id,af.flight_id,ati.flight_departure_date;
```

| profile_id | first_name | last_name | flight_id | flight_departure_date | No_of_Tickets |
|------------|------------|-----------|-----------|-----------------------|---------------|
| 1 | John | Stuart | 3175 | 2013-05-25 | 1 |
| 1 | John | Stuart | 3178 | 2013-05-06 | 1 |
| 2 | Robert | Clive | 3178 | 2013-06-05 | 1 |

5. Write a query to display flight id,from location, to location and ticket price of flights whose departure is in the month of april.Display the records sorted in ascending order based on flight id and then by from location.

```
SELECT af.flight_id,af.from_location,af.to_location,afd.price FROM
```

```
air_flight af JOIN air_flight_details afd ON af.flight_id=afd.flight_id
and month(afd.flight_departure_date)='04'

ORDER BY af.flight_id,af.from_location;
```

| flight_id | from_location | to_location | price |
|-----------|---------------|-------------|---------|
| 3173 | hyderabad | chennai | 2000.00 |
| 3174 | kolkata | delhi | 3800.00 |
| 3179 | chennai | kolkata | 4000.00 |
| 3180 | kolkata | kochi | 3000.00 |

6. Write a query to display the average cost of the tickets in each flight on all scheduled dates. The query should display flight_id, from_location, to_location and Average price as "Price". Display the records sorted in ascending order based on flight id and then by from_location and then by to_location.

```
SELECT af.flight_id,af.from_location,af.to_location,avg(afd.price) Average_Price FROM
air_flight af JOIN air_flight_details afd ON af.flight_id=afd.flight_id
GROUP BY af.flight_id
ORDER BY af.flight_id,af.from_location,af.to_location;
```

| flight_id | from_location | to_location | Average_Price |
|-----------|---------------|-------------|---------------|
| 3170 | delhi | kolkata | 1000.000000 |
| 3171 | chennai | delhi | 5000.000000 |
| 3172 | kolkata | chennai | 3000.000000 |
| 3173 | hyderabad | chennai | 2000.000000 |
| 3174 | kolkata | delhi | 3800.000000 |
| 3175 | chennai | hyderabad | 3500.000000 |
| 3176 | kochi | chennai | 8000.000000 |
| 3177 | delhi | kochi | 1500.000000 |
| 3178 | chennai | hyderabad | 3000.000000 |
| 3179 | chennai | kolkata | 4000.000000 |
| 3180 | kolkata | kochi | 3000.000000 |

7. Write a query to display the customers who have booked tickets from Chennai to Hyderabad. The query should display profile_id, customer_name (combine first_name & last_name with comma in

b/w), address of the customer. Give an alias to the name as customer_name. Hint: Query should fetch unique customers irrespective of multiple tickets booked. Display the records sorted in ascending order based on profile id.

```
SELECT ap.profile_id,concat(ap.first_name,',',ap.last_name) customer_name,ap.address FROM
air_passenger_profile ap JOIN air_ticket_info ati ON ap.profile_id=ati.profile_id
JOIN air_flight af ON af.flight_id=ati.flight_id
WHERE af.from_location='Chennai' and af.to_location='Hyderabad'
GROUP BY ati.profile_id
ORDER BY ap.profile_id;
```

| profile_id | Customer_name | address |
|------------|---------------|---|
| 1 | John,Stuart | Street 21, Near Bus Stop-Hyderabad-432126 |
| 2 | Robert,Clive | Sector 3, Technopolis-Kolkata-700102 |

8. Write a query to display profile id of the passenger(s) who has/have booked maximum number of tickets. In case of multiple records, display the records sorted in ascending order based on profile id.

```
SELECT profile_id FROM air_ticket_info
group by profile_id
having count(ticket_id)>=all(select count(ticket_id)
from air_ticket_info
group by profile_id) order by profile_id;
```

| profile_id |
|------------|
| 1 |
| 3 |

9. Write a query to display the total number of tickets as “No_of_Tickets” booked in each flight in ABC Airlines. The Query should display the flight_id, from_location, to_location and the number of tickets. Display only the flights in which atleast 1 ticket is booked. Display the records sorted in ascending order based on flight id.

```
SELECT f.flight_id,f.from_location,f.to_location,COUNT(t.ticket_id) AS No_of_Tickets
FROM air_ticket_info t JOIN air_flight f
ON f.flight_id = t.flight_id where AIRLINE_NAME = 'abc' GROUP by
f.flight_id,f.from_location,f.to_location
```

having count(t.ticket_id)>=1

ORDER by f.flight_id;

| flight_id | from_location | to_location | No_of_Tickets |
|-----------|---------------|-------------|---------------|
| 3175 | chennai | hyderabad | 1 |
| 3179 | chennai | kolkata | 1 |

10. Write a query to display the no of services offered by each flight and the total price of the services. The Query should display flight_id, number of services as “No_of_Services” and the cost as “Total_Price” in the same order. Order the result by Total Price in descending order and then by flight_id in descending order.Hint:The number of services can be calculated from the number of scheduled departure dates of the flight

SELECT flight_id,count(flight_departure_date) No_of_services,sum(price) Total_Price FROM

air_flight_details GROUP BY flight_id

ORDER BY Total_price DESC,flight_id DESC;

| flight_id | No_of_services | Total_Price |
|-----------|----------------|-------------|
| 3176 | 1 | 8000.00 |
| 3171 | 1 | 5000.00 |
| 3179 | 1 | 4000.00 |
| 3174 | 1 | 3800.00 |
| 3175 | 1 | 3500.00 |
| 3180 | 1 | 3000.00 |
| 3178 | 1 | 3000.00 |
| 3172 | 1 | 3000.00 |
| 3173 | 1 | 2000.00 |
| 3177 | 1 | 1500.00 |
| 3170 | 1 | 1000.00 |

11. Write a query to display the number of passengers who have travelled in each flight in each scheduled date. The Query should display flight_id, flight_departure_date and the number of passengers as “No_of_Passengers” in the same order.Display the records sorted in ascending order based on flight id and then by flight departure date.

SELECT flight_id,flight_departure_date,count(ticket_id) No_of_passengers FROM

air_ticket_info GROUP BY flight_id,flight_departure_date

ORDER BY flight_id,flight_departure_date;

| flight_id | flight_departure_date | No_of_passengers |
|-----------|-----------------------|------------------|
| 3171 | 2013-03-15 | 2 |
| 3172 | 2013-02-06 | 1 |
| 3175 | 2013-05-25 | 1 |
| 3176 | 2013-03-14 | 1 |
| 3177 | 2013-06-15 | 2 |
| 3178 | 2013-05-06 | 1 |
| 3178 | 2013-06-05 | 1 |
| 3179 | 2013-04-03 | 1 |
| 3180 | 2013-04-02 | 1 |

12. Write a query to display profile id of passenger(s) who booked minimum number of tickets. In case of multiple records, display the records sorted in ascending order based on profile id.

```
SELECT profile_id FROM air_ticket_info
GROUP BY profile_id HAVING count(ticket_id)<=ALL
(SELECT count(ticket_id) FROM air_ticket_info GROUP BY profile_id)
ORDER BY profile_id;
```

| profile_id |
|------------|
| 5 |

13. Write a query to display unique passenger profile id, first name, mobile number and email address of passengers who booked ticket to travel from HYDERABAD to CHENNAI. Display the records sorted in ascending order based on profile id.

```
SELECT DISTINCT ap.profile_id,ap.first_name,ap.mobile_number,ap.email_id FROM
air_passenger_profile ap JOIN air_ticket_info ati ON ap.profile_id=ati.profile_id
JOIN air_flight af ON ati.flight_id=af.flight_id
WHERE af.from_location='Hyderabad' and af.to_location='Chennai'
ORDER BY profile_id;
```

| profile_id | first_name | mobile_number | email_id |
|------------|------------|---------------|----------|
|------------|------------|---------------|----------|

14. Write a query to intimate the passengers who are boarding Chennai to Hyderabad Flight on 6th May 2013 stating the delay of 1hr in the departure time. The Query should display the passenger's profile_id, first_name,last_name, flight_id, flight_departure_date, actual departure time , actual arrival time , delayed departure time as "Delayed_Departure_Time", delayed arrival time as "Delayed_Arrival_Time" Hint: Distinct Profile ID should be displayed irrespective of multiple tickets booked by the same profile.Display the records sorted in ascending order based on passenger's profile id.

```
SELECT DISTINCT ap.profile_id,ap.first_name,ap.last_name,ati.flight_id,ati.flight_departure_date,
af.departure_time,af.arrival_time,
addtime(af.departure_time,'01:00:00') Delayed_Departure_Time,
addtime(af.arrival_time,'01:00:00') Delayed_Arrival_Time FROM
air_passenger_profile ap JOIN air_ticket_info ati ON ap.profile_id=ati.profile_id
JOIN air_flight af ON af.flight_id=ati.flight_id
WHERE af.from_location='Chennai' and af.to_location='Hyderabad'
and ati.flight_departure_date='2013-05-06'
ORDER BY profile_id;
```

| profile_id | first_name | last_name | flight_id | flight_departure_date | departure_time | arrival_time | Delayed_Departure_Time | Delayed_Arrival_Time |
|------------|------------|-----------|-----------|-----------------------|----------------|--------------|------------------------|----------------------|
| 1 | John | Stuart | 3178 | 2013-05-06 | 08:00:00 | 09:00:00 | 09:00:00 | 10:00:00 |

15. Write a query to display the number of tickets as “No_of_Tickets” booked by Kochi Customers. The Query should display the Profile_Id, First_Name, Base_Location and number of tickets booked.Hint: Use String functions to get the base location of customer from their Address and give alias name as “Base_Location”Display the records sorted in ascending order based on customer first name.

```
SELECT ap.profile_id,ap.first_name,
substring_index(substring_index(ap.address,'-',2),'-',-1) Base_Location,
count(ati.ticket_id) No_of_Tickets FROM
air_passenger_profile ap JOIN air_ticket_info ati ON ati.profile_id=ap.profile_id
WHERE ap.address LIKE '%Kochi%'
ORDER BY ap.first_name;
```

| profile_id | first_name | Base_Location | No_of_Tickets |
|------------|------------|---------------|---------------|
| 3 | Raj | Kochi | 3 |

16. Write a query to display the flight_id, from_location, to_location, number of Services as “No_of_Services” offered in the month of May.

```
SELECT af.flight_id,af.from_location,af.to_location,count(afd.flight_departure_date) No_of_services
FROM
```

```
air_flight af JOIN air_flight_details afd ON af.flight_id=afd.flight_id
```

```
WHERE month(flight_departure_date)='05'
```

```
GROUP BY af.flight_id,af.from_location,af.to_location
```

```
ORDER BY af.flight_id;
```

| flight_id | from_location | to_location | No_of_services |
|-----------|---------------|-------------|----------------|
| 3175 | chennai | hyderabad | 1 |
| 3178 | chennai | hyderabad | 1 |

17. Write a query to display profile id,last name,mobile number and email id of passengers whose base location is chennai.Display the records sorted in ascending order based on profile id.

```
SELECT profile_id, last_name, mobile_number, email_id
```

```
FROM air_passenger_profile
```

```
WHERE address LIKE '%Chennai%'
```

```
ORDER BY profile_id;
```

| profile_id | last_name | mobile_number | email_id |
|------------|-----------|---------------|----------------|
| 5 | Stark | 9832015785 | tony@gmail.com |

18. Write a query to display number of flights between 6.00 AM and 6.00 PM from chennai. Hint Use FLIGHT_COUNT as alias name.

```
SELECT count(flight_id) FLIGHT_COUNT FROM air_flight
```

```
WHERE from_location='CHENNAI'
```

```
AND departure_time BETWEEN '06:00:00' AND '18:00:00';
```

| FLIGHT_COUNT |
|--------------|
|--------------|

19. Write a query to display unique profile id,first name , email id and contact number of passenger(s) who travelled on flight with id 3178. Display the records sorted in ascending order based on first name.

```
SELECT DISTINCT ap.profile_id,ap.first_name,ap.email_id,ap.mobile_number FROM
air_passenger_profile ap JOIN air_ticket_info ati ON ap.profile_id=ati.profile_id
WHERE ati.flight_id='3178'
ORDER BY ap.first_name;
```

| profile_id | first_name | email_id | mobile_number |
|------------|------------|------------------|---------------|
| 1 | John | john@gmail.com | 9865263251 |
| 2 | Robert | robert@yahoo.com | 9733015875 |

20. Write a query to display flight id,departure date,flight type of all flights. Flight type can be identified based on the following rules : if ticket price is less than 3000 then 'AIR PASSENGER',ticket price between 3000 and less than 4000 'AIR BUS' and ticket price between 4000 and greater than 4000 then 'EXECUTIVE PASSENGER'. Hint use FLIGHT_TYPE as alias name.Display the records sorted in ascendeing order based on flight_id and then by departure date.

```
SELECT flight_id,flight_departure_date,
case when price<3000 then 'AIR PASSENGER'
      when price>=3000 and price<4000 then 'AIR BUS'
      when price>=4000 then 'EXECUTIVE PASSENGER'
end FLIGHT_TYPE FROM air_flight_details
ORDER BY flight_id,flight_departure_date;
```

| flight_id | flight_departure_date | FLIGHT_TYPE |
|-----------|-----------------------|---------------------|
| 3170 | 2013-02-14 | AIR PASSENGER |
| 3171 | 2013-03-15 | EXECUTIVE PASSENGER |
| 3172 | 2013-02-05 | AIR BUS |
| 3173 | 2013-04-07 | AIR PASSENGER |
| 3174 | 2013-04-05 | AIR BUS |
| 3175 | 2013-05-25 | AIR BUS |
| 3176 | 2013-03-14 | EXECUTIVE PASSENGER |
| 3177 | 2013-06-15 | AIR PASSENGER |
| 3178 | 2013-05-06 | AIR BUS |
| 3179 | 2013-04-03 | EXECUTIVE PASSENGER |
| 3180 | 2013-04-02 | AIR BUS |

21. Write a query to display the credit card type and no of credit cards used on the same type. Display the records sorted in ascending order based on credit card type.Hint: Use CARD_COUNT AS Alias name for no of cards.

```
SELECT card_type, count(card_type) Card_Count FROM air_credit_card_details
GROUP BY card_type ORDER BY card_type;
```

| card_type | Card_Count |
|-----------|------------|
| credit | 4 |
| debit | 5 |

22. Write a Query to display serial no, first name, mobile number, email id of all the passengers who holds email address from gmail.com. The Serial No will be the last three digits of profile ID. Hint: Use SERIAL_NO as Alias name for serial number. Display the records sorted in ascending order based on name.

```
SELECT substring(profile_id,-3) SERIAL_NO,first_name,mobile_number,email_id FROM
air_passenger_profile
WHERE email_id LIKE '%@gmail.com'
ORDER BY first_name;
```

| SERIAL_NO | first_name | mobile_number | email_id |
|-----------|------------|---------------|----------------|
| | John | 9865263251 | john@gmail.com |
| | Tony | 9832015785 | tony@gmail.com |

23. Write a query to display the flight(s) which has least number of services in the month of May. The Query should fetch flight_id, from_location, to_location, least number of Services as “No_of_Services” Hint: Number of services offered can be calculated from the number of scheduled departure dates of a flight if there are multiple flights, display them sorted in ascending order based on flight id.

```
SELECT afd.flight_id,af.from_location,af.to_location,count(afd.flight_id) No_of_Services
FROM air_flight_details afd JOIN air_flight af ON af.flight_id=afd.flight_id
WHERE monthname(afd.flight_departure_date)='May'
GROUP BY afd.flight_departure_date HAVING count(afd.flight_id) <=
ALL(SELECT count(flight_id) FROM air_flight_details
WHERE monthname(flight_departure_date)='May'
GROUP BY flight_departure_date)
```

ORDER BY flight_id;

| flight_id | from_location | to_location | No_of_Services |
|-----------|---------------|-------------|----------------|
| 3175 | chennai | hyderabad | 1 |
| 3178 | chennai | hyderabad | 1 |

24. Write a query to display the flights available in Morning, AfterNoon, Evening& Night. The Query should display the Flight_Id, From_Location, To_Location , Departure_Time, time of service as "Time_of_Service". Time of Service should be calculated as: From 05:00:01 Hrs to 12:00:00 Hrs - Morning, 12:00:01 to 18:00:00 Hrs -AfterNoon, 18:00:01 to 24:00:00 - Evening and 00:00:01 to 05:00:00 - NightDisplay the records sorted in ascending order based on flight id.

```
SELECT flight_id,from_location,to_location,Departure_Time,
CASE
WHEN departure_time BETWEEN ('05:00:01') AND ('12:00:00')
THEN 'Morning'
WHEN departure_time BETWEEN ('12:00:01') AND ('18:00:00')
THEN 'AfterNoon'
WHEN departure_time BETWEEN ('18:00:01') AND ('24:00:00')
THEN 'Evening'
WHEN departure_time='00:00:00'
THEN 'Evening'
WHEN departure_time BETWEEN ('00:00:01') AND ('05:00:00')
THEN 'Night'
END Time_of_Service
FROM air_flight
order by flight_id;
```

| flight_id | from_location | to_location | Departure_Time | Time_of_Service |
|-----------|---------------|-------------|----------------|-----------------|
| 3170 | delhi | kolkata | 21:00:00 | Evening |
| 3171 | chennai | delhi | 08:00:00 | Moming |
| 3172 | kolkata | chennai | 11:30:00 | Moming |
| 3173 | hyderabad | chennai | 06:30:00 | Moming |
| 3174 | kolkata | delhi | 00:00:00 | Evening |
| 3175 | chennai | hyderabad | 15:00:00 | AfterNoon |
| 3176 | kochi | chennai | 18:00:00 | AfterNoon |
| 3177 | delhi | kochi | 19:00:00 | Evening |
| 3178 | chennai | hyderabad | 08:00:00 | Moming |
| 3179 | chennai | kolkata | 14:00:00 | AfterNoon |
| 3180 | kolkata | kochi | 05:00:00 | Night |

25. Write a query to display the number of flights flying from each location. The Query should display the from location and the number of flights to other locations as “No_of_Flights”. Hint: Get the distinct from location and to location. Display the records sorted in ascending order based on from location.

```
SELECT from_location, count(flight_id) No_of_Flights FROM
air_flight GROUP BY from_location
ORDER BY from_location;
```

| from_location | No_of_Flights |
|---------------|---------------|
| chennai | 4 |
| delhi | 2 |
| hyderabad | 1 |
| kochi | 1 |
| kolkata | 3 |

26. Write a query to display the number of passengers traveled in each flight in each scheduled date. The Query should display flight_id, from_location, To_location, flight_departure_date and the number of passengers as “No_of_Passengers”. Hint: The Number of passengers inclusive of all the tickets booked with single profile id. Display the records sorted in ascending order based on flight id and then by flight departure date.

```
SELECT af.flight_id, af.from_location, af.to_location, ati.flight_departure_date,
count(ati.ticket_id) No_of_Passengers FROM
air_flight af JOIN air_ticket_info ati ON af.flight_id=ati.flight_id
GROUP BY af.flight_id, af.from_location, af.to_location, ati.flight_departure_date
```

ORDER BY af.flight_id,ati.flight_departure_date;

| flight_id | from_location | to_location | flight_departure_date | No_of_Passengers |
|-----------|---------------|-------------|-----------------------|------------------|
| 3171 | chennai | delhi | 2013-03-15 | 2 |
| 3172 | kolkata | chennai | 2013-02-06 | 1 |
| 3175 | chennai | hyderabad | 2013-05-25 | 1 |
| 3176 | kochi | chennai | 2013-03-14 | 1 |
| 3177 | delhi | kochi | 2013-06-15 | 2 |
| 3178 | chennai | hyderabad | 2013-05-06 | 1 |
| 3178 | chennai | hyderabad | 2013-06-05 | 1 |
| 3179 | chennai | kolkata | 2013-04-03 | 1 |
| 3180 | kolkata | kochi | 2013-04-02 | 1 |

27. Write a query to display the flight details in which more than 10% of seats have been booked. The query should display Flight_Id, From_Location, To_Location,Total_Seats, seats booked as “No_of_Seats_Booked” .Display the records sorted in ascending order based on flight id and then by No_of_Seats_Booked.

```
SELECT af.flight_id,af.from_location,af.to_location,af.total_seats,
(af.total_seats-afd.available_seats) No_of_Seats_Booked FROM
air_flight_details afd JOIN air_flight af ON afd.flight_id=af.flight_id
WHERE (af.total_seats-afd.available_seats)>(af.total_seats*0.1)
ORDER BY flight_id,No_of_Seats_Booked;
```

| flight_id | from_location | to_location | total_seats | No_of_Seats_Booked |
|-----------|---------------|-------------|-------------|--------------------|
| 3170 | delhi | kolkata | 100 | 90 |
| 3171 | chennai | delhi | 100 | 100 |
| 3172 | kolkata | chennai | 100 | 68 |
| 3173 | hyderabad | chennai | 100 | 88 |
| 3174 | kolkata | delhi | 100 | 97 |
| 3175 | chennai | hyderabad | 200 | 190 |
| 3176 | kochi | chennai | 100 | 98 |
| 3177 | delhi | kochi | 200 | 200 |
| 3178 | chennai | hyderabad | 200 | 195 |
| 3179 | chennai | kolkata | 100 | 85 |
| 3180 | kolkata | kochi | 200 | 186 |

28. Write a query to display the Flight_Id, Flight_Departure_Date, From_Location,To_Location and Duration of all flights which has duration of travel less than 1 Hour, 10 Minutes.

```
SELECT af.flight_Id,afd.flight_Departure_Date,af.From_Location,af.To_Location,af.duration
```

```
FROM air_flight af JOIN air_flight_details afd ON af.flight_id=afd.flight_id
WHERE af.duration<'01:10:00';
```

| flight_id | flight_Departure_Date | From_Location | To_Location | duration |
|-----------|-----------------------|---------------|-------------|----------|
| 3173 | 2013-04-07 | hyderabad | chennai | 00:45:00 |
| 3175 | 2013-05-25 | chennai | hyderabad | 01:00:00 |
| 3176 | 2013-03-14 | kochi | chennai | 01:05:00 |
| 3178 | 2013-05-06 | chennai | hyderabad | 01:00:00 |
| 3179 | 2013-04-03 | chennai | kolkata | 01:00:00 |
| 3180 | 2013-04-02 | kolkata | kochi | 00:45:00 |

29. Write a query to display the flight_id, from_location,to_location,number of services as “No_of_Services” , average ticket price as “Average_Price” whose average ticket price is greater than the total average ticket cost of all flights. Order the result by lowest average price.

```
SELECT afd.flight_id,af.from_location,af.to_location,
count(afd.flight_departure_date) No_of_Service, avg(price) Average_Price
FROM air_flight af JOIN air_flight_details afd ON af.flight_id=afd.flight_id
GROUP BY af.flight_id,af.from_location,af.to_location
HAVING avg(price)>(SELECT avg(price) FROM air_flight_details)
ORDER BY average_price;
```

| flight_id | from_location | to_location | No_of_Service | Average_Price |
|-----------|---------------|-------------|---------------|---------------|
| 3175 | chennai | hyderabad | 1 | 3500.000000 |
| 3174 | kolkata | delhi | 1 | 3800.000000 |
| 3179 | chennai | kolkata | 1 | 4000.000000 |
| 3171 | chennai | delhi | 1 | 5000.000000 |
| 3176 | kochi | chennai | 1 | 8000.000000 |

MOVIE

CREATE DATABASE video;USE video;

Create table CUSTOMER_MASTER

(CUSTOMER_ID Varchar(10),CUSTOMER_NAME Varchar(30) NOT NULL,CONTACT_NO BIGINT(10),CONTACT_ADD Varchar(20),DATE_OF_REGISTRATION Date NOT NULL,AGE Varchar(15)NOT NULL,Constraint MT_cts1 PRIMARY KEY(CUSTOMER_ID));

Create table LIBRARY_CARD_MASTER

(CARD_ID Varchar(10),DESCRIPTION Varchar(30) NOT NULL,AMOUNT BIGINT(50),NUMBER_OF_YEARS bigint(10) NOT NULL,Constraint MT_cts2 PRIMARY KEY(CARD_ID));

Create table MOVIES_MASTER

(MOVIE_ID Varchar(10), MOVIE_NAME Varchar(50) NOT NULL,RELEASE_DATE Varchar(30) NOT NULL,LANGUAGE Varchar(30),RATING int(2),DURATION VARCHAR(10) NOT NULL, MOVIE_TYPE Varchar(3),MOVIE_CATEGORY VARCHAR(20) NOT NULL,DIRECTOR VARCHAR(20) NOT NULL,

LEAD_ROLE_1 Varchar(3) NOT NULL,LEAD_ROLE_2 VARCHAR(4) NOT NULL,RENT_COST BIGINT(10),Constraint MT_cts4 PRIMARY KEY(MOVIE_ID));

Create table CUSTOMER_CARD_DETAILS

(CUSTOMER_ID Varchar(10),CARD_ID VARCHAR(10),ISSUE_DATE DATE NOT NULL,Constraint MT_cts3 PRIMARY KEY(CUSTOMER_ID),Constraint MT_CTS41 FOREIGN KEY(CUSTOMER_ID) References CUSTOMER_MASTER(CUSTOMER_ID),Constraint MT_CTS42 FOREIGN KEY(CARD_ID) References LIBRARY_CARD_MASTER(CARD_ID));

Create table CUSTOMER_ISSUE_DETAILS

(ISSUE_ID Varchar(10) NOT NULL,CUSTOMER_ID Varchar(10) NOT NULL,MOVIE_ID VARCHAR(10), ISSUE_DATE Date NOT NULL,RETURN_DATE Date NOT NULL,

ACTUAL_DATE_RETURN Date NOT NULL,Constraint MT_cts5 PRIMARY KEY(ISSUE_ID),Constraint MT_Mem FOREIGN KEY(CUSTOMER_ID) References CUSTOMER_MASTER(CUSTOMER_ID), Constraint MT_Mem1 FOREIGN KEY(MOVIE_ID) References MOVIES_MASTER(MOVIE_ID));

Insert into CUSTOMER_MASTER Values('CUS001', 'AMIT', 9876543210,'ADD1', '2012-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS002', 'ABDHUL', 8765432109,'ADD2', '2012-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS003', 'GAYAN', 7654321098,'ADD3', '2012-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS004', 'RADHA', 6543210987,'ADD4', '2012-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS005', 'GURU', NULL,'ADD5', '2012-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS006', 'MOHAN', 4321098765,'ADD6', '2012-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS007', 'NAME7', 3210987654,'ADD7', '2012-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS008', 'NAME8', 2109876543,'ADD8', '2013-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS009', 'NAME9', NULL,'ADD9', '2013-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS010', 'NAM10', 9934567890,'ADD10', '2013-02-12', '21');

Insert into CUSTOMER_MASTER Values('CUS011', 'NAM11', 9875678910,'ADD11', '2013-02-12', '21');

Insert into LIBRARY_CARD_MASTER Values('CR001', 'Silver', 200, 5);

Insert into LIBRARY_CARD_MASTER Values('CR002', 'Gold', 400, 9);

Insert into LIBRARY_CARD_MASTER Values('CR003', 'Platinum', 600, 8);

Insert into LIBRARY_CARD_MASTER Values('CR004', 'VISA', 800, 7);

Insert into LIBRARY_CARD_MASTER Values('CR005', 'CREDIT', 1200, 6);

Insert into MOVIES_MASTER Values('MV001', 'DIEHARD', '2012-05-13','ENGLISH', 4 , '2HRS', 'U/A','ACTION','DIR1','L1','L2',100);

Insert into MOVIES_MASTER Values('MV002', 'THE MATRIX', '2012-05-13','ENGLISH', 4 , '2HRS', 'A','ACTION','DIR2','L1','L2',100);

Insert into MOVIES_MASTER Values('MV003', 'INCEPTION', '2012-05-13','ENGLISH', 4 , '2HRS', 'U/A','ACTION','DIR3','L15','L2',100);

Insert into MOVIES_MASTER Values('MV004', 'DARK KNIGHT', '2012-05-13','ENGLISH', 4 , '2HRS', 'A','ACTION','DIR4','L15','L2',100);

Insert into MOVIES_MASTER Values('MV005', 'OFFICE S', '2012-05-13','ENGLISH', 4 , '2HRS', 'U/A','COMEDY','DIR5','L12','L24',100);

Insert into MOVIES_MASTER Values('MV006', 'SHAWN OF DEAD', '2012-05-13','ENGLISH', 4 , '2HRS', 'U/A','COMEDY','DIR6','L1','L25',100);

Insert into MOVIES_MASTER Values('MV007', 'YOUNG FRANKEN', '2012-05-13','ENGLISH', 4 , '2HRS', 'U/A','COMEDY','DIR7','L1','L2',100);

Insert into MOVIES_MASTER Values('MV008', 'CAS', '2012-05-13','ENGLISH', 4 , '2HRS', 'A','ROMANCE','DIR8','L1','L2',100);

Insert into MOVIES_MASTER Values('MV009', 'GWW', '2012-05-13','ENGLISH', 4 , '2HRS', 'A','ROMANCE','DIR9','L1','L2',100);

Insert into MOVIES_MASTER Values('MV010', 'TITANIC', '2012-05-13','ENGLISH', 4 , '2HRS', 'A','ROMANCE','DIR10','L1','L2',100);

Insert into MOVIES_MASTER Values('MV011', 'THE NOTE BOOK', '2012-05-13','ENGLISH', 4 , '2HRS', 'A','ROMANCE','DIR11','L1','L2',100);

Insert into CUSTOMER_CARD_DETAILS Values('CUS001', 'CR001', '2012-05-13');

Insert into CUSTOMER_CARD_DETAILS Values('CUS002', 'CR002', '2012-05-13');

Insert into CUSTOMER_CARD_DETAILS Values('CUS003', 'CR002', '2013-05-13');

Insert into CUSTOMER_CARD_DETAILS Values('CUS004', 'CR003', '2013-05-13');

Insert into CUSTOMER_CARD_DETAILS Values('CUS005', 'CR003', '2012-05-13');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS001', 'CUS001', 'MV001', '2012-05-13', '2012-05-13','2012-05-13');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS002', 'CUS001', 'MV001', '2012-05-01', '2012-05-16','2012-05-16');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS003', 'CUS002', 'MV004', '2012-05-02', '2012-05-06', '2012-05-16');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS004', 'CUS002', 'MV004', '2012-04-03', '2012-04-16', '2012-04-20');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS005', 'CUS002', 'MV009', '2012-04-04', '2012-04-16', '2012-04-20');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS006', 'CUS003', 'MV002', '2012-03-30', '2012-04-15', '2012-04-20');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS007', 'CUS003', 'MV003', '2012-04-20', '2012-05-05', '2012-05-05');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS008', 'CUS003', 'MV005', '2012-04-21', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS009', 'CUS003', 'MV001', '2012-04-22', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS010', 'CUS003', 'MV009', '2012-04-22', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS011', 'CUS003', 'MV010', '2012-04-23', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS012', 'CUS003', 'MV010', '2012-04-24', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS013', 'CUS003', 'MV008', '2012-04-25', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS014', 'CUS004', 'MV007', '2012-04-26', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS015', 'CUS004', 'MV006', '2012-04-27', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS016', 'CUS004', 'MV006', '2012-04-28', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS017', 'CUS004', 'MV001', '2012-04-29', '2012-05-07', '2012-05-25');

Insert into CUSTOMER_ISSUE_DETAILS Values ('IS018', 'CUS010', 'MV008', '2012-04-24', '2012-05-07', '2012-05-25');



| LEAD_ROLE_2 | RENT_COST |
|-------------|-----------|
| L2 | 100 |
| L2 | 100 |
| L2 | 100 |
| L2 | 100 |
| L24 | 100 |
| L25 | 100 |
| L2 | 100 |
| L2 | 100 |
| L2 | 100 |
| L2 | 100 |
| L2 | 100 |
| NULL | NULL |

CUSTOMER MASTER

| CUSTOMER_ID | CUSTOMER_NAME | CONTACT_NO | CONTACT_ADD | DATE_OF_REGISTRATION | AGE |
|-------------|---------------|------------|-------------|----------------------|------|
| CUS001 | AMIT | 9876543210 | ADD1 | 2012-02-12 | 21 |
| CUS002 | ABDHUL | 8765432109 | ADD2 | 2012-02-12 | 21 |
| CUS003 | GAYAN | 7654321098 | ADD3 | 2012-02-12 | 21 |
| CUS004 | RADHA | 6543210987 | ADD4 | 2012-02-12 | 21 |
| CUS005 | GURU | NULL | ADD5 | 2012-02-12 | 21 |
| CUS006 | MOHAN | 4321098765 | ADD6 | 2012-02-12 | 21 |
| CUS007 | NAME7 | 3210987654 | ADD7 | 2012-02-12 | 21 |
| CUS008 | NAME8 | 2109876543 | ADD8 | 2013-02-12 | 21 |
| CUS009 | NAME9 | NULL | ADD9 | 2013-02-12 | 21 |
| CUS010 | NAM10 | 9934567890 | ADD10 | 2013-02-12 | 21 |
| CUS011 | NAM11 | 9875678910 | ADD11 | 2013-02-12 | 21 |
| NULL | NULL | NULL | NULL | NULL | NULL |

LIBRARY CARD MASTER

| CARD_ID | DESCRIPTION | AMOUNT | NUMBER_OF_YEARS |
|---------|-------------|--------|-----------------|
| CR001 | Silver | 200 | 5 |
| CR002 | Gold | 400 | 9 |
| CR003 | Platinum | 600 | 8 |
| CR004 | VISA | 800 | 7 |
| CR005 | CREDIT | 1200 | 6 |
| NULL | NULL | NULL | NULL |

CUSTOMER CARD DETAILS

| CUSTOMER_ID | CARD_ID | ISSUE_DATE |
|-------------|---------|------------|
| CUS001 | CR001 | 2012-05-13 |
| CUS002 | CR002 | 2012-05-13 |
| CUS003 | CR002 | 2013-05-13 |
| CUS004 | CR003 | 2013-05-13 |
| CUS005 | CR003 | 2012-05-13 |
| NULL | NULL | NULL |

CUSTOMER ISSUE DETAILS

| ISSUE_ID | CUSTOMER_ID | MOVIE_ID | ISSUE_DATE | RETURN_DATE | ACTUAL_DATE_RETURN |
|----------|-------------|----------|------------|-------------|--------------------|
| IS001 | CUS001 | MV001 | 2012-05-13 | 2012-05-13 | 2012-05-13 |
| IS002 | CUS001 | MV001 | 2012-05-01 | 2012-05-16 | 2012-05-16 |
| IS003 | CUS002 | MV004 | 2012-05-02 | 2012-05-06 | 2012-05-16 |
| IS004 | CUS002 | MV004 | 2012-04-03 | 2012-04-16 | 2012-04-20 |
| IS005 | CUS002 | MV009 | 2012-04-04 | 2012-04-16 | 2012-04-20 |
| IS006 | CUS003 | MV002 | 2012-03-30 | 2012-04-15 | 2012-04-20 |
| IS007 | CUS003 | MV003 | 2012-04-20 | 2012-05-05 | 2012-05-05 |
| IS008 | CUS003 | MV005 | 2012-04-21 | 2012-05-07 | 2012-05-25 |
| IS009 | CUS003 | MV001 | 2012-04-22 | 2012-05-07 | 2012-05-25 |
| IS010 | CUS003 | MV009 | 2012-04-22 | 2012-05-07 | 2012-05-25 |
| IS011 | CUS003 | MV010 | 2012-04-23 | 2012-05-07 | 2012-05-25 |
| IS012 | CUS003 | MV010 | 2012-04-24 | 2012-05-07 | 2012-05-25 |
| IS013 | CUS003 | MV008 | 2012-04-25 | 2012-05-07 | 2012-05-25 |
| IS014 | CUS004 | MV007 | 2012-04-26 | 2012-05-07 | 2012-05-25 |
| IS015 | CUS004 | MV006 | 2012-04-27 | 2012-05-07 | 2012-05-25 |
| IS016 | CUS004 | MV006 | 2012-04-28 | 2012-05-07 | 2012-05-25 |
| IS017 | CUS004 | MV001 | 2012-04-29 | 2012-05-07 | 2012-05-25 |
| IS018 | CUS010 | MV008 | 2012-04-24 | 2012-05-07 | 2012-05-25 |
| IS019 | CUS011 | MV009 | 2012-04-27 | 2012-05-07 | 2012-05-25 |
| NULL | NULL | NULL | NULL | NULL | NULL |

1. Write a query to display movie names and number of times that movie is issued to customers. In case movies are never issued to customers display number of times as 0. Display the details in sorted order based on number of times (in descending order) and then by movie name (in ascending order). The Alias name for the number of movies issued is ISSUE_COUNT.

```
SELECT m.MOVIE_NAME, count(ISSUE_ID) ISSUE_COUNT FROM
movies_master m LEFT JOIN customer_issue_details c ON m.MOVIE_ID=c.MOVIE_ID
GROUP BY m.movie_name
```

ORDER BY ISSUE_COUNT DESC,MOVIE_NAME;

| MOVIE_NAME | ISSUE_COUNT |
|---------------|-------------|
| DIEHARD | 4 |
| GWW | 3 |
| CAS | 2 |
| DARK KNIGHT | 2 |
| SHAWN OF DEAD | 2 |
| TITANIC | 2 |
| INCEPTION | 1 |
| OFFICE S | 1 |
| THE MATRIX | 1 |
| YOUNG FRANKEN | 1 |
| THE NOTE BOOK | 0 |

2. Write a query to display id,name,age,contact no of customers whose age is greater than 25 and who have registered in the year 2012. Display contact no in the below format +91-XXX-XXX-XXXX example +91-987-678-3434 and use the alias name as "CONTACT_ISD". If the contact no is null then display as 'N/A' Sort all the records in ascending order based on age and then by name.

```
SELECT CUSTOMER_ID,CUSTOMER_NAME,AGE,ifnull(
concat('+91-',substring(contact_no,1,3)),'-',
substring(contact_no,4,3)),'-',substring(contact_no,7)), 'N/A') CONTACT_ISD
FROM customer_master WHERE age>25 and year(date_of_registration)='2012'
ORDER BY age,CUSTOMER_NAME;
```

| CUSTOMER_ID | CUSTOMER_NAME | AGE | CONTACT_ISD |
|-------------|---------------|-----|-------------|
|-------------|---------------|-----|-------------|

3. Write a query to display the movie category and number of movies in that category. Display records based on number of movies from higher to lower order and then by movie category in ascending order. Hint: Use NO_OF_MOVIES as alias name for number of movies.

```
SELECT MOVIE_CATEGORY,count(MOVIE_ID) NO_OF_MOVIES FROM
movies_master GROUP BY MOVIE_CATEGORY
ORDER BY NO_OF_MOVIES DESC,MOVIE_CATEGORY;
```


| MOVIE_CATEGORY | NO_OF_MOVIES |
|----------------|--------------|
| ACTION | 4 |
| ROMANCE | 4 |
| COMEDY | 3 |

4. Write a query to display the number of customers having card with description "Gold card".

**
Hint: Use CUSTOMER_COUNT as alias name for number of customers**

```
SELECT count(c.customer_id) CUSTOMER_COUNT FROM
library_card_master l JOIN customer_card_details c ON l.CARD_ID=c.CARD_ID
WHERE description='Gold';
```

| CUSTOMER_COUNT |
|----------------|
| 2 |

5. Write a query to display the customer id, customer name, year of registration, library card id, card issue date of all the customers who hold library card. Display the records sorted by customer name in descending order. Use REGISTERED_YEAR as alias name for year of registration.

```
SELECT c.customer_id,c.customer_name,
year(c.DATE_OF_REGISTRATION) REGISTERED_YEAR,cd.card_id,cd.issue_date FROM
customer_master c JOIN customer_card_details cd ON c.customer_id=cd.customer_id
ORDER BY CUSTOMER_NAME DESC;
```

| customer_id | customer_name | REGISTERED_YEAR | card_id | issue_date |
|-------------|---------------|-----------------|---------|------------|
| CUS004 | RADHA | 2012 | CR003 | 2013-05-13 |
| CUS005 | GURU | 2012 | CR003 | 2012-05-13 |
| CUS003 | GAYAN | 2012 | CR002 | 2013-05-13 |
| CUS001 | AMIT | 2012 | CR001 | 2012-05-13 |
| CUS002 | ABDHUL | 2012 | CR002 | 2012-05-13 |

6. Write a query to display issue id, customer id, customer name for the customers who have paid fine and whose name starts with 'R'. Fine is calculated based on return date and actual date of return. If the date of actual return is after date of return then fine need to be paid by the customer order by customer name.

```
SELECT ci.issue_id,ci.CUSTOMER_ID,c.CUSTOMER_NAME FROM
customer_master c JOIN customer_issue_details ci ON c.customer_id=ci.customer_id
WHERE customer_name LIKE 'R%' and ci.actual_date_return>ci.return_date
```

ORDER BY customer_name;

| issue_id | CUSTOMER_ID | CUSTOMER_NAME |
|----------|-------------|---------------|
| IS014 | CUS004 | RADHA |
| IS015 | CUS004 | RADHA |
| IS016 | CUS004 | RADHA |
| IS017 | CUS004 | RADHA |

7. Write a query to display customer id, customer name, card id, card description and card amount in dollars of customers who have taken movie on the same day the library card is registered. For Example Assume John registered a library card on 12th Jan 2013 and he took a movie on 12th Jan 2013 then display his details. AMOUNT_DOLLAR = amount/52.42 and round it to zero decimal places and display as \$Amount. Example Assume 500 is the amount then dollar value will be \$10. Hint: Use AMOUNT_DOLLAR as alias name for amount in dollar. Display the records in ascending order based on customer name.

```
SELECT c.CUSTOMER_ID,c.CUSTOMER_NAME,l.card_id,l.DESRIPTION,
concat('$',round(amount/52.42)) AMOUNT_DOLLAR FROM
customer_master c JOIN customer_issue_details ci ON c.customer_id=ci.customer_id
JOIN customer_card_details cc ON cc.customer_id=c.customer_id
JOIN library_card_master l ON cc.card_id=l.card_id
WHERE c.DATE_OF_REGISTRATION=ci.issue_date
ORDER BY customer_name;
```

| CUSTOMER_ID | CUSTOMER_NAME | card_id | DESCRIPTION | AMOUNT_DOLLAR |
|-------------|---------------|---------|-------------|---------------|
|-------------|---------------|---------|-------------|---------------|

8. Write a query to display the customer id, customer name, contact number and address of customers who have taken movies from library without library card and whose address ends with 'Nagar'. Display customer name in upper case. Hint: Use CUSTOMER_NAME as alias name for customer name. Display the details sorted in ascending order based on customer name.

```
SELECT CUSTOMER_ID,upper(CUSTOMER_NAME) CUSTOMER_NAME,contact_no,contact_add
FROM
customer_master WHERE contact_add LIKE '%Nagar' and
customer_id NOT IN (SELECT customer_id FROM customer_card_details)
and customer_id IN (SELECT customer_id FROM customer_issue_details)
```

ORDER BY CUSTOMER_NAME;

| CUSTOMER_ID | CUSTOMER_NAME | contact_no | contact_add |
|-------------|---------------|------------|-------------|
|-------------|---------------|------------|-------------|

9. Write a query to display the movie id, movie name, release year, director name of movies acted by the lead actor 1 who acted maximum number of movies. Display the records sorted in ascending order based on movie name.

```
SELECT movie_id, movie_name, release_date, director FROM movies_master
WHERE lead_role_1 IN (SELECT lead_role_1 FROM
(SELECT lead_role_1, count(movie_id) ct FROM movies_master
GROUP BY lead_role_1) t WHERE t.ct >= ALL (SELECT count(movie_id)
FROM movies_master GROUP BY lead_role_1)) ORDER BY movie_name;
```

| movie_id | movie_name | release_date | director |
|----------|----------------|--------------|----------|
| MV008 | CAS | 2012-05-13 | DIR8 |
| MV001 | DIEHARD | 2012-05-13 | DIR1 |
| MV009 | GWW | 2012-05-13 | DIR9 |
| MV006 | SHAWN OF DEAD | 2012-05-13 | DIR6 |
| MV002 | THE MATRIX | 2012-05-13 | DIR2 |
| MV011 | THE NOTE BOOK | 2012-05-13 | DIR11 |
| MV010 | TITANIC | 2012-05-13 | DIR10 |
| MV007 | YOUNG FRANK... | 2012-05-13 | DIR7 |

**10. Write a query to display the customer name and number of movies issued to that customer sorted by customer name in ascending order. If a customer has not been issued with any movie then display 0.
Hint: Use MOVIE_COUNT as alias name for number of movies issued.**

```
SELECT c.customer_name, count(ci.movie_id) MOVIE_COUNT FROM
customer_master c LEFT JOIN customer_issue_details ci ON c.customer_id=ci.customer_id
GROUP BY c.customer_id ORDER BY c.customer_name;
```

| customer_name | MOVIE_COUNT |
|---------------|-------------|
| ABDHUL | 3 |
| AMIT | 2 |
| GAYAN | 8 |
| GURU | 0 |
| MOHAN | 0 |
| NAM10 | 1 |
| NAM11 | 1 |
| NAME7 | 0 |
| NAME8 | 0 |
| NAME9 | 0 |
| RADHA | 4 |

11. Write a query to display serial number, issue id, customer id, customer name, movie id and movie name of all the videos that are issued and display in ascending order based on serial number. Serial number can be generated from the issue id , that is last two characters of issue id is the serial number. For Example Assume the issue id is I00005 then the serial number is 05 Hint: Alias name for serial number is 'SERIAL_NO'

```
SELECT substring(ci.issue_id,-2) SERIAL_NO,ci.issue_id,c.customer_id,c.customer_name,
m.movie_id,m.movie_name FROM customer_master c JOIN customer_issue_details ci
ON c.customer_id=ci.customer_id JOIN movies_master m ON m.movie_id=ci.movie_id
ORDER BY SERIAL_NO;
```

| SERIAL_NO | issue_id | customer_id | customer_name | movie_id | movie_name |
|-----------|----------|-------------|---------------|----------|---------------|
| 01 | IS001 | CUS001 | AMIT | MV001 | DIEHARD |
| 02 | IS002 | CUS001 | AMIT | MV001 | DIEHARD |
| 03 | IS003 | CUS002 | ABDHUL | MV004 | DARK KNIGHT |
| 04 | IS004 | CUS002 | ABDHUL | MV004 | DARK KNIGHT |
| 05 | IS005 | CUS002 | ABDHUL | MV009 | GWW |
| 06 | IS006 | CUS003 | GAYAN | MV002 | THE MATRIX |
| 07 | IS007 | CUS003 | GAYAN | MV003 | INCEPTION |
| 08 | IS008 | CUS003 | GAYAN | MV005 | OFFICE S |
| 09 | IS009 | CUS003 | GAYAN | MV001 | DIEHARD |
| 10 | IS010 | CUS003 | GAYAN | MV009 | GWW |
| 11 | IS011 | CUS003 | GAYAN | MV010 | TITANIC |
| 12 | IS012 | CUS003 | GAYAN | MV010 | TITANIC |
| 13 | IS013 | CUS003 | GAYAN | MV008 | CAS |
| 14 | IS014 | CUS004 | RADHA | MV007 | YOUNG FRAN... |
| 15 | IS015 | CUS004 | RADHA | MV006 | SHAWN OF D... |
| 16 | IS016 | CUS004 | RADHA | MV006 | SHAWN OF D... |
| 17 | IS017 | CUS004 | RADHA | MV001 | DIEHARD |
| 18 | IS018 | CUS010 | NAM10 | MV008 | CAS |
| 19 | IS019 | CUS011 | NAM11 | MV009 | GWW |

12. Write a query to display the issue id, issue date, customer id, customer name and contact number for videos that are issued in the year 2013. Display the records in descending order based on issue date of the video.

```
SELECT ci.issue_id, ci.issue_date, c.customer_id, c.customer_name, c.contact_no FROM
customer_master c JOIN customer_issue_details ci ON c.customer_id=ci.customer_id
and year(ci.issue_date)='2013' ORDER BY ci.issue_date DESC;
```

| issue_id | issue_date | customer_id | customer_name | contact_no |
|----------|------------|-------------|---------------|------------|
|----------|------------|-------------|---------------|------------|

**13. Write a query to display movie id, movie name and actor names of movies which are not issued to any customers.
 Actors Name to be displayed in the below format. LEAD_ACTOR_ONE space ambersant space LEAD_ACTOR_TWO. Example: Assume lead**

actor one's name is "Jack Tomson" and Lead actor two's name is "Maria" then Actors name will be "Jack Tomsom & Maria"Hint:Use ACTORS as alias name for actors name.
 Display the records in ascending order based on movie name.

```
SELECT movie_id,movie_name,concat(lead_role_1,' & ',lead_role_2) ACTOR FROM
movies_master
```

```
WHERE movie_id NOT IN (SELECT movie_id FROM customer_issue_details) ORDER BY
movie_name;
```

| movie_id | movie_name | ACTOR |
|----------|---------------|---------|
| MV011 | THE NOTE BOOK | L1 & L2 |

14.Write a query to display the director's name, movie name and lead_actor_name1 of all the movies directed by the director who directed more than one movie. Display the directors name in capital letters. Use DIRECTOR_NAME as alias name for director name column Display the records sorted in ascending order based on director_name and then by movie_name in descending order.

```
SELECT upper(director) DIRECTOR_NAME,movie_name,lead_role_1 FROM movies_master
GROUP BY director HAVING count(movie_id)>1 ORDER BY director,movie_name DESC;
```

| DIRECTOR_NAME | movie_name | lead_role_1 |
|---------------|------------|-------------|
|---------------|------------|-------------|

15.Write a query to display number of customers who have registered in the library in the year 2012 and who have given/provided contact number.
 Hint:Use NO_OF_CUSTOMERS as alias name for number of customers.

```
SELECT count(customer_id) NO_OF_CUSTOMER FROM customer_master
WHERE contact_no is not null and year(date_of_registration)='2012';
```

| NO_OF_CUSTOMER |
|----------------|
| 6 |

16.Write a query to display the customer's name, contact number,library card id and library card description of all the customers irrespective of customers holding a library card. If customer contact number is not available then display his address. Display the records sorted in ascending order based on customer name. Hint: Use CONTACT_DETAILS as alias name for customer contact.

```

SELECT c.customer_name,ifnull(c.contact_no,c.contact_add)
CONTACT_DETAILS,l.card_id,l.description FROM
customer_master c LEFT JOIN customer_card_details cc ON c.customer_id=cc.customer_id
LEFT JOIN library_card_master l ON l.card_id=cc.card_id
ORDER BY customer_name;

```

| customer_name | CONTACT_DETAILS | card_id | description |
|---------------|-----------------|---------|-------------|
| ABDHUL | 8765432109 | CR002 | Gold |
| AMIT | 9876543210 | CR001 | Silver |
| GAYAN | 7654321098 | CR002 | Gold |
| GURU | ADD5 | CR003 | Platinum |
| MOHAN | 4321098765 | NULL | NULL |
| NAM10 | 9934567890 | NULL | NULL |
| NAM11 | 9875678910 | NULL | NULL |
| NAME7 | 3210987654 | NULL | NULL |
| NAME8 | 2109876543 | NULL | NULL |
| NAME9 | ADD9 | NULL | NULL |
| RADHA | 6543210987 | CR003 | Platinum |

17. Write a query to display the customer id, customer name and number of times the same movie is issued to the same customers who have taken same movie more than once. Display the records sorted by customer name in decending order For Example: Assume customer John has taken Titanic three times and customer Ram has taken Die hard only once then display the details of john. Hint: Use NO_OF_TIMES as alias name for number of times

```

SELECT ci.customer_id,c.customer_name,count(ci.movie_id) NO_OF_TIMES FROM
customer_issue_details ci JOIN customer_master c ON c.customer_id=ci.customer_id
GROUP BY ci.customer_id,ci.movie_id HAVING count(movie_id)>1
ORDER BY customer_name DESC;

```

| customer_id | customer_name | NO_OF_TIMES |
|-------------|---------------|-------------|
| CUS004 | RADHA | 2 |
| CUS003 | GAYAN | 2 |
| CUS001 | AMIT | 2 |
| CUS002 | ABDHUL | 2 |

18. Write a query to display customer id, customer name, contact number, movie category and number of movies issued to each customer based on movie category who has been issued with more than one movie in that category. Example: Display contact number as "+91-876-

456-2345" format. Hint:Use NO_OF_MOVIES as alias name for number of movies column. Hint:Use CONTACT_ISD as alias name for contact number. Display the records sorted in ascending order based on customer name and then by movie category.

```
SELECT c.customer_id,c.customer_name,concat('+91-',substring(c.contact_no,1,3),'-',
substring(c.contact_no,4,3),'-',substring(c.contact_no,7)) CONTACT_ISD
,m.movie_category,count(cc.movie_id) NO_OF_MOVIES FROM customer_master c JOIN
customer_issue_details cc
ON c.customer_id=cc.customer_id JOIN movies_master m ON m.movie_id=cc.movie_id
GROUP BY c.customer_id,m.movie_category HAVING count(cc.movie_id)>1
ORDER BY customer_name,movie_category;
```

| customer_id | customer_name | CONTACT_ISD | movie_category | NO_OF_MOVIES |
|-------------|---------------|------------------|----------------|--------------|
| CUS002 | ABDHUL | +91-876-543-2109 | ACTION | 2 |
| CUS001 | AMIT | +91-987-654-3210 | ACTION | 2 |
| CUS003 | GAYAN | +91-765-432-1098 | ACTION | 3 |
| CUS003 | GAYAN | +91-765-432-1098 | ROMANCE | 4 |
| CUS004 | RADHA | +91-654-321-0987 | COMEDY | 3 |

19. Write a query to display customer id and customer name of customers who has been issued with maximum number of movies and customer who has been issued with minimum no of movies. For example Assume customer John has been issued 5 movies, Ram has been issued 10 movies and Kumar has been issued 2 movies. The name and id of Ram should be displayed for issuing maximum movies and Kumar should be displayed for issuing minimum movies. Consider only the customers who have been issued with atleast 1 movie Customer(s) who has/have been issued the maximum number of movies must be displayed first followed by the customer(s) who has/have been issued with the minimum number of movies. In case of multiple customers who have been displayed with the maximum or minimum number of movies, display the records sorted in ascending order based on customer name.

```
SELECT cid.customer_id , customer_name FROM customer_master cm JOIN
customer_issue_details cid ON cm.customer_id=cid.customer_id

GROUP BY customer_id , customer_name

HAVING count(movie_id)>=ALL(SELECT count(movie_id)

FROM customer_issue_details

GROUP BY customer_id)

UNION
```



```

SELECT cid.customer_id , customer_name FROM
customer_master cm JOIN customer_issue_details cid
ON cm.customer_id=cid.customer_id
GROUP BY customer_id , customer_name
HAVING count(movie_id)<=ALL(SELECT count(movie_id)
FROM customer_issue_details
GROUP BY customer_id) ORDER BY customer_name;

```

| customer_id | customer_name |
|-------------|---------------|
| CUS003 | GAYAN |
| CUS010 | NAM10 |
| CUS011 | NAM11 |

20. Write a query to display the customer id , customer name and number of times movies have been issued from Comedy category. Display only for customers who has taken more than once. Hint: Use NO_OF_TIMES as alias name Display the records in ascending order based on customer name.

```

SELECT c.customer_id,c.customer_name,count(m.movie_id) NO_OF_TIMES FROM
customer_master c JOIN customer_issue_details cc ON c.customer_id=cc.customer_id
JOIN movies_master m ON m.movie_id=cc.movie_id
WHERE m.movie_category='Comedy'
GROUP BY c.customer_id HAVING count(m.movie_id)>1
ORDER BY customer_name;

```

| customer_id | customer_name | NO_OF_TIMES |
|-------------|---------------|-------------|
| CUS004 | RADHA | 3 |

21. Write a query to display customer id and total rent paid by the customers who are issued with the videos. Need not display the customers who has not taken / issued with any videos. Hint: Alias Name for total rent paid is TOTAL_COST. Display the records sorted in ascending order based on customer id

```

SELECT cid.customer_id, sum(m.rent_cost) TOTAL_COST FROM customer_issue_details cid
JOIN movies_master mm ON cid.movie_id=mm.movie_id GROUP BY cid.customer_id order by
customer_id;

```

| customer_id | TOTAL_COST |
|-------------|------------|
| CUS001 | 200 |
| CUS002 | 300 |
| CUS003 | 800 |
| CUS004 | 400 |
| CUS010 | 100 |
| CUS011 | 100 |

LOAN

```
create database loan;
```

```
use loan;
```

```
CREATE TABLE loan_card_master
```

```
(  
    loan_id      varchar(6)    PRIMARY KEY,  
    loan_type    varchar(15),  
    duration_in_years  int(2)  
);
```

```
CREATE TABLE employee_master
```

```
(  
    employee_id    varchar(6)    PRIMARY KEY,  
    employee_name  varchar(20),  
    designation     varchar(25),  
    department     varchar(25),  
    gender         char(1),  
    date_of_birth  date,  
    date_of_joining  date  
);
```

```
CREATE TABLE item_master
```

```
(  
    item_id      varchar(6)    PRIMARY KEY,  
    item_description  varchar(25),
```

```
        issue_status      char(1),
        item_make         varchar(25),
        item_category     varchar(20),
        item_valuation    int(6)
    );
```

```
CREATE TABLE employee_card_details
```

```
(
    employee_id          varchar(6)    REFERENCES employee_master,
    loan_id              varchar(6)    REFERENCES loan_card_master,
    card_issue_date      date
);
```

```
CREATE TABLE employee_issue_details
```

```
(
    issue_id             varchar(6)    PRIMARY KEY,
    employee_id          varchar(6)    REFERENCES employee_master,
    item_id              varchar(6)    REFERENCES item_master,
    issue_date           date,
    return_date          date
);
```

```
insert into loan_card_master values('L00001','Furniture',5);
```

```
insert into loan_card_master values('L00002','Stationary',0);
```

```
insert into loan_card_master values('L00003','Crockery',1);
```

```
insert into employee_issue_details values('ISS001','E00001','I00001','2012-02-03','2014-02-03');
insert into employee_issue_details values('ISS002','E00001','I00004','2012-02-03','2020-02-03');
insert into employee_issue_details values('ISS003','E00002','I00005','2013-01-03','2015-01-03');
insert into employee_issue_details values('ISS004','E00003','I00007','2010-07-04','2012-07-04');
insert into employee_issue_details values('ISS005','E00003','I00008','2010-07-04','2012-08-05');
insert into employee_issue_details values('ISS006','E00003','I00010','2012-03-14','2012-06-15');
insert into employee_issue_details values('ISS007','E00004','I00012','2013-04-14','2016-04-14');
insert into employee_issue_details values('ISS008','E00006','I00018','2012-08-18','2019-04-17');
insert into employee_issue_details values('ISS009','E00004','I00018','2013-04-18','2013-05-18');
```

```
insert into employee_master values('E00001','Ram','Manager','Finance','M','1973-12-01','2000-01-01');
```

```
insert into employee_master values('E00002','Abhay','Assistant Manager','Finance','M','1976-01-01','2006-12-01');
```

```
insert into employee_master values('E00003','Anita','Senior Executive','Marketing','F','1977-05-12','2007-03-21');
```

```
insert into employee_master values('E00004','Zuben','Manager','Marketing','M','1974-10-12','2003-07-23');
```

```
insert into employee_master values('E00005','Radhica','Manager','HR','F','1976-07-22','2004-01-23');
```

```
insert into employee_master values('E00006','John','Executive','HR','M','1983-11-08','2010-05-17');
```

```
insert into employee_card_details values('E00001','L00001','2000-01-01');
```

```
insert into employee_card_details values('E00001','L00002','2000-01-01');
```

```
insert into employee_card_details values('E00001','L00003','2002-12-14');
```

```
insert into employee_card_details values('E00002','L00001','2007-02-01');
```

insert into employee_card_details values('E00002','L00002','2007-03-11');

insert into employee_card_details values('E00003','L00001','2007-04-15');

insert into employee_card_details values('E00003','L00002','2007-04-15');

insert into employee_card_details values('E00003','L00003','2007-04-15');

INSERT INTO item_master VALUES ('I00001','Tea Table','Y','Wooden','Furniture',5000);

INSERT INTO item_master VALUES ('I00002','Dinning Table','N','Wooden','Furniture',15000);

INSERT INTO item_master VALUES ('I00003','Tea Table','N','Steel','Furniture',6000);

INSERT INTO item_master VALUES ('I00004','Side Table','Y','Wooden','Furniture',2000);

INSERT INTO item_master VALUES ('I00005','Side Table','Y','Steel','Furniture',1500);

INSERT INTO item_master VALUES ('I00006','Tea Table','N','Steel','Furniture',7000);

INSERT INTO item_master VALUES ('I00007','Dinning Chair','Y','Wooden','Furniture',1500);

INSERT INTO item_master VALUES ('I00008','Tea Table','Y','Wooden','Furniture',4000);

INSERT INTO item_master VALUES ('I00009','Sofa','N','Wooden','Furniture',18000);

INSERT INTO item_master VALUES ('I00010','Cupboard','Y','Steel','Furniture',10000);

INSERT INTO item_master VALUES ('I00011','Cupboard','N','Steel','Furniture',14000);

INSERT INTO item_master VALUES ('I00012','Double Bed','Y','Wooden','Furniture',21000);

INSERT INTO item_master VALUES ('I00013','Double Bed','Y','Wooden','Furniture',20000);

INSERT INTO item_master VALUES ('I00014','Single Bed','Y','Steel','Furniture',10000);

INSERT INTO item_master VALUES ('I00015','Single Bed','N','Steel','Furniture',10000);

INSERT INTO item_master VALUES ('I00016','Tea Set','Y','Glass','Crockery',3000);

INSERT INTO item_master VALUES ('I00017','Tea Set','Y','Bonechina','Crockery',4000);

INSERT INTO item_master VALUES ('I00018','Dinning Set','Y','Glass','Crockery',4500);

INSERT INTO item_master VALUES ('I00019','Dinning Set','N','Bonechina','Crockery',5000);

INSERT INTO item_master VALUES ('I00020','Pencil','Y','Wooden','Stationary',5);

INSERT INTO item_master VALUES ('I00021','Pen','Y','Plastic','Stationary',100);

INSERT INTO item_master VALUES ('I00022','Pen','N','Plastic','Stationary',200);

LOAN CARD MASTER

| loan_id | loan_type | duration_in_years |
|---------|------------|-------------------|
| L00001 | Furniture | 5 |
| L00002 | Stationary | 0 |
| L00003 | Crockery | 1 |
| NULL | NULL | NULL |

EMPLOYEE CARD DETAILS

| employee_id | loan_id | card_issue_date |
|-------------|---------|-----------------|
| E00001 | L00001 | 2000-01-01 |
| E00001 | L00002 | 2000-01-01 |
| E00001 | L00003 | 2002-12-14 |
| E00002 | L00001 | 2007-02-01 |
| E00002 | L00002 | 2007-03-11 |
| E00003 | L00001 | 2007-04-15 |
| E00003 | L00002 | 2007-04-15 |
| E00003 | L00003 | 2007-04-15 |

EMPLOYEE ISSUE DETAILS

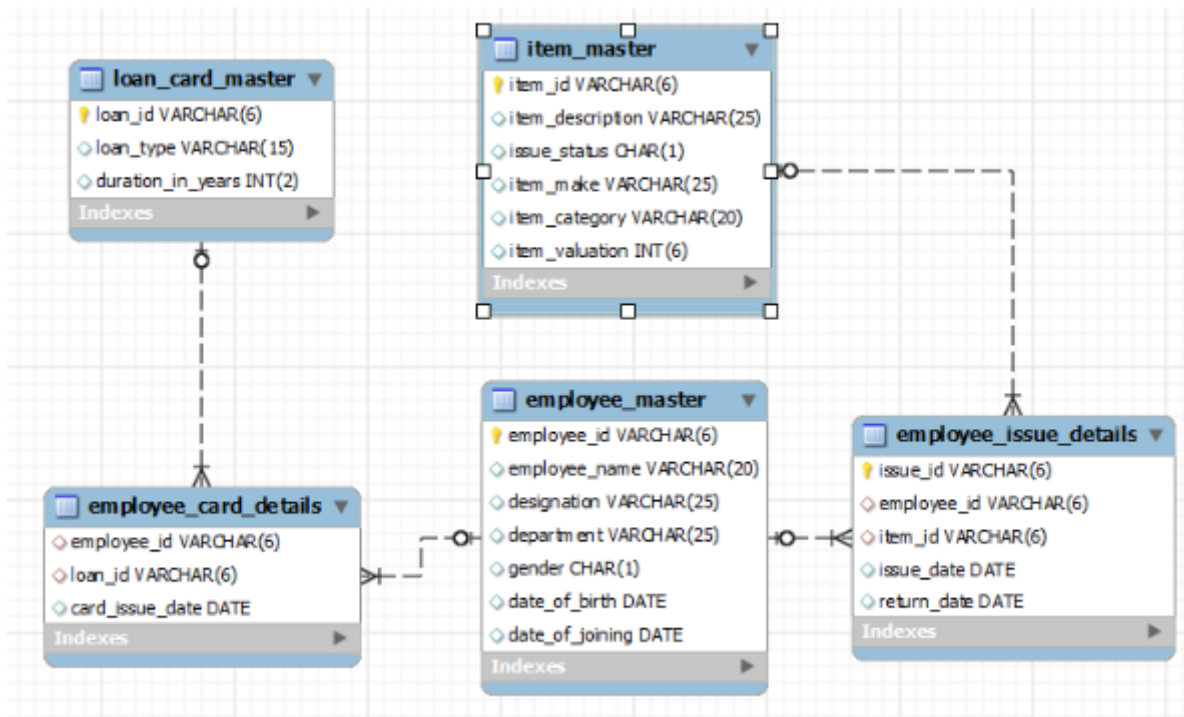
| issue_id | employee_id | item_id | issue_date | return_date |
|----------|-------------|---------|------------|-------------|
| ISS001 | E00001 | I00001 | 2012-02-03 | 2014-02-03 |
| ISS002 | E00001 | I00004 | 2012-02-03 | 2020-02-03 |
| ISS003 | E00002 | I00005 | 2013-01-03 | 2015-01-03 |
| ISS004 | E00003 | I00007 | 2010-07-04 | 2012-07-04 |
| ISS005 | E00003 | I00008 | 2010-07-04 | 2012-08-05 |
| ISS006 | E00003 | I00010 | 2012-03-14 | 2012-06-15 |
| ISS007 | E00004 | I00012 | 2013-04-14 | 2016-04-14 |
| ISS008 | E00006 | I00018 | 2012-08-18 | 2019-04-17 |
| ISS009 | E00004 | I00018 | 2013-04-18 | 2013-05-18 |
| NULL | NULL | NULL | NULL | NULL |

EMPLOYEE MASTER

| employee_id | employee_name | designation | department | gender | date_of_birth | date_of_joining |
|-------------|---------------|-------------------|------------|--------|---------------|-----------------|
| E00001 | Ram | Manager | Finance | M | 1973-12-01 | 2000-01-01 |
| E00002 | Abhay | Assistant Manager | Finance | M | 1976-01-01 | 2006-12-01 |
| E00003 | Anita | Senior Executive | Marketing | F | 1977-05-12 | 2007-03-21 |
| E00004 | Zuben | Manager | Marketing | M | 1974-10-12 | 2003-07-23 |
| E00005 | Radhica | Manager | HR | F | 1976-07-22 | 2004-01-23 |
| E00006 | John | Executive | HR | M | 1983-11-08 | 2010-05-17 |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |

ITEM MASTER

| item_id | item_description | issue_status | item_make | item_category | item_valuation |
|---------|------------------|--------------|-----------|---------------|----------------|
| I00001 | Tea Table | Y | Wooden | Furniture | 5000 |
| I00002 | Dinning Table | N | Wooden | Furniture | 15000 |
| I00003 | Tea Table | N | Steel | Furniture | 6000 |
| I00004 | Side Table | Y | Wooden | Furniture | 2000 |
| I00005 | Side Table | Y | Steel | Furniture | 1500 |
| I00006 | Tea Table | N | Steel | Furniture | 7000 |
| I00007 | Dinning Chair | Y | Wooden | Furniture | 1500 |
| I00008 | Tea Table | Y | Wooden | Furniture | 4000 |
| I00009 | Sofa | N | Wooden | Furniture | 18000 |
| I00010 | Cupboard | Y | Steel | Furniture | 10000 |
| I00011 | Cupboard | N | Steel | Furniture | 14000 |
| I00012 | Double Bed | Y | Wooden | Furniture | 21000 |
| I00013 | Double Bed | Y | Wooden | Furniture | 20000 |
| I00014 | Single Bed | Y | Steel | Furniture | 10000 |
| I00015 | Single Bed | N | Steel | Furniture | 10000 |
| I00016 | Tea Set | Y | Glass | Crockery | 3000 |
| I00017 | Tea Set | Y | Bonechina | Crockery | 4000 |
| I00018 | Dinning Set | Y | Glass | Crockery | 4500 |
| I00019 | Dinning Set | N | Bonechina | Crockery | 5000 |
| I00020 | Pencil | Y | Wooden | Stationary | 5 |
| I00021 | Pen | Y | Plastic | Stationary | 100 |
| I00022 | Pen | N | Plastic | Stationary | 200 |
| NULL | NULL | NULL | NULL | NULL | NULL |



1. Write a query to display category and number of items in that category. Give the count an alias name of Count_category. Display the details on the sorted order of count in descending order.

```
SELECT item_category,count(item_id) Count_category FROM
item_master GROUP BY item_category ORDER BY Count_category DESC;
```

| item_category | Count_category |
|---------------|----------------|
| Furniture | 15 |
| Crockery | 4 |
| Stationary | 3 |

2. Write a query to display the number of employees in HR department. Give the alias name as No_of_Employees.

```
SELECT count(employee_id) No_of_Employees FROM
employee_master WHERE department='HR';
```

| No_of_Employees |
|-----------------|
| 2 |

3. Write a query to display employee id, employee name, designation and department for employees who have never been issued an item as a loan from the company. Display the records sorted in ascending order based on employee id.

```
SELECT employee_id,employee_name,designation,department FROM employee_master
```

WHERE employee_id NOT IN (SELECT employee_id FROM employee_issue_details)

ORDER BY employee_id;

| employee_id | employee_name | designation | department |
|-------------|---------------|-------------|------------|
| E00005 | Radhica | Manager | HR |
| NULL | NULL | NULL | NULL |

4. Write a query to display the employee id, employee name who was issued an item of highest valuation. In case of multiple records, display the records sorted in ascending order based on employee id.[Hint Suppose an item called dinning table is of 22000 and that is the highest price of the item that has been issued. So display the employee id and employee name who issued dinning table whose price is 22000.]

SELECT employee_id,employee_name FROM employee_master

WHERE employee_id IN(SELECT employee_id FROM employee_issue_details

WHERE item_id IN (SELECT item_id FROM item_master

WHERE item_valuation=(SELECT max(item_valuation) FROM

item_master i JOIN employee_issue_details e ON i.item_id=e.item_id)));

| employee_id | employee_name |
|-------------|---------------|
| E00004 | Zuben |
| NULL | NULL |

5. Write a query to display issue_id, employee_id, employee_name. Display the records sorted in ascending order based on issue id.

SELECT eid.issue_id, eid.employee_id, em.employee_name

FROM employee_master em JOIN employee_issue_details eid

ON em.employee_id=eid.employee_id ORDER BY eid.issue_id;

| issue_id | employee_id | employee_name |
|----------|-------------|---------------|
| ISS001 | E00001 | Ram |
| ISS002 | E00001 | Ram |
| ISS003 | E00002 | Abhay |
| ISS004 | E00003 | Anita |
| ISS005 | E00003 | Anita |
| ISS006 | E00003 | Anita |
| ISS007 | E00004 | Zuben |
| ISS008 | E00006 | John |
| ISS009 | E00004 | Zuben |

6. Write a query to display employee id, employee name who don't have loan cards. Display the records sorted in ascending order based on employee id.

```
SELECT employee_id, employee_name FROM employee_master
```

```
WHERE employee_id NOT IN (SELECT employee_id FROM employee_card_details);
```

| employee_id | employee_name |
|-------------|---------------|
| E00004 | Zuben |
| E00005 | Radhica |
| E00006 | John |
| NULL | NULL |

7. Write a query to count the number of cards issued to an employee "Ram". Give the count an alias name as No_of_Cards.

```
SELECT count(loan_id) No_of_Cards FROM
```

```
employee_card_details WHERE employee_id IN
```

```
(SELECT employee_id FROM employee_master WHERE employee_name='Ram');
```

(or)

```
SELECT count(loan_id) No_of_Cards FROM
```

```
employee_card_details c JOIN employee_master e
```

```
ON c.employee_id = e.employee_id
```

```
WHERE e.employee_name = 'Ram';
```

| No_of_Cards |
|-------------|
| 3 |

8. Write a query to display the count of customers who have gone for loan type stationary. Give the count an alias name as Count_stationary.

```
SELECT count(e.employee_id) Count_Stationary
```

```
FROM employee_card_details e JOIN loan_card_master l
```

```
ON e.loan_id=l.loan_id WHERE l.loan_type='Stationary';
```

| Count_Stationary |
|------------------|
| 3 |

9. Write a query to display the employee id, employee name and number of items issued to them. Give the number of items an alias name as Count. Display the details in descending order of count and then

```
SELECT e.employee_id, employee_name, count(e.item_id) Count FROM
```

employee_issue_details e JOIN employee_master em ON e.employee_id=em.employee_id
GROUP BY e.employee_id ORDER BY count DESC,e.employee_id;

| employee_id | employee_name | Count |
|-------------|---------------|-------|
| E00003 | Anita | 3 |
| E00001 | Ram | 2 |
| E00004 | Zuben | 2 |
| E00002 | Abhay | 1 |
| E00006 | John | 1 |

10. Write a query to display the employee id, employee name who was issued an item of minimum valuation. In case of multiple records, display them sorted in ascending order based on employee id. [Hint Suppose an item called pen is of rupees 20 and that is the lowest price. So display the employee id and employee name who issued pen where the valuation is 20.]

```
SELECT employee_id, employee_name FROM employee_master
WHERE employee_id IN (SELECT employee_id FROM employee_issue_details
WHERE item_id IN (SELECT item_id FROM item_master
WHERE item_valuation = (SELECT min(item_valuation) FROM
item_master i JOIN employee_issue_details e ON i.item_id = e.item_id)))
ORDER BY employee_id;
```

| employee_id | employee_name |
|-------------|---------------|
| E00002 | Abhay |
| E00003 | Anita |
| NULL | NULL |

11. Write a query to display the employee id, employee name and total valuation of the product issued to each employee. Give the alias name as TOTAL_VALUATION. Display the records sorted in ascending order based on employee id. Consider only employees who have been issued atleast 1 item.

```
SELECT e.employee_id, em.employee_name, sum(i.item_valuation) TOTAL_VALUATION FROM
item_master i JOIN employee_issue_details e ON e.item_id = i.item_id
JOIN employee_master em ON em.employee_id = e.employee_id
GROUP BY e.employee_id ORDER BY employee_id;
```

| employee_id | employee_name | TOTAL_VALUATION |
|-------------|---------------|-----------------|
| E00001 | Ram | 7000 |
| E00002 | Abhay | 1500 |
| E00003 | Anita | 15500 |
| E00004 | Zuben | 25500 |
| E00006 | John | 4500 |

12. Write a query to display distinct employee id, employee name who kept the item issued for more than a year. Hint: Use Date time function to calculate the difference between item issue and return date. Display the records only if it is more than 365 Days.Display the records sorted in ascending order based on employee id.

```
SELECT DISTINCT e.employee_id,e.employee_name FROM
employee_master e JOIN employee_issue_details ei ON e.employee_id=ei.employee_id
WHERE datediff(ei.return_date,ei.issue_date)>365
ORDER BY employee_id;
```

| employee_id | employee_name |
|-------------|---------------|
| E00001 | Ram |
| E00002 | Abhay |
| E00003 | Anita |
| E00004 | Zuben |
| E00006 | John |

13. Write a query to display employee id, employee name and count of items of those who asked for more than 1 furniture. Give the alias name for count of items as COUNT_ITEMS.Display the records sorted in ascending order on employee id.

```
SELECT e.employee_id,e.employee_name,count(ei.item_id) COUNT_ITEMS FROM
employee_master e JOIN employee_issue_details ei ON e.employee_id=ei.employee_id
JOIN item_master i ON ei.item_id=i.item_id
WHERE i.item_category='Furniture'
GROUP BY ei.employee_id HAVING count(ei.item_id)>1;
```

| employee_id | employee_name | COUNT_ITEMS |
|-------------|---------------|-------------|
| E00001 | Ram | 2 |
| E00003 | Anita | 3 |

14. Write a query to display the number of men & women Employees. The query should display the gender and number of Employees as No_of_Employees. Display the records sorted in ascending order based on gender.

```
SELECT gender,count(employee_id) FROM employee_master
```

```
GROUP BY gender ORDER BY gender;
```

| gender | count(employee_id) |
|--------|--------------------|
| F | 2 |
| M | 4 |

15. Write a query to display employee id, employee name who joined the company after 2005. Display the records sorted in ascending order based on employee id.

```
SELECT employee_id,employee_name FROM employee_master
```

```
WHERE year(date_of_joining)>'2005'
```

```
ORDER BY employee_id;
```

| employee_id | employee_name |
|-------------|---------------|
| E00002 | Abhay |
| E00003 | Anita |
| E00006 | John |
| NULL | NULL |

16. Write a query to get the number of items of the furniture category issued and not issued. The query should display issue status and the number of furniture as No_of_Furnitures. Display the records sorted in ascending order based on issue_status.

```
SELECT issue_status,count(item_id) No_of_Furniture FROM
```

```
item_master WHERE item_category='Furniture'
```

```
GROUP BY issue_status ORDER BY issue_status;
```

| issue_status | No_of_Furniture |
|--------------|-----------------|
| N | 6 |
| Y | 9 |

17. Write a query to find the number of items in each category, make and description. The Query should display Item Category, Make, description and the number of items as No_of_Items. Display the records in ascending order based on Item Category, then by item make and then by item description.

```
SELECT item_category,item_make,item_description,count(item_id) No_of_items FROM
```

```
item_master GROUP BY item_category,item_make,item_description
```

```
ORDER BY item_category,item_make,item_description;
```

| item_category | item_make | item_description | No_of_items |
|---------------|-----------|------------------|-------------|
| Crockery | Bonechina | Dinning Set | 1 |
| Crockery | Bonechina | Tea Set | 1 |
| Crockery | Glass | Dinning Set | 1 |
| Crockery | Glass | Tea Set | 1 |
| Furniture | Steel | Cupboard | 2 |
| Furniture | Steel | Side Table | 1 |
| Furniture | Steel | Single Bed | 2 |
| Furniture | Steel | Tea Table | 2 |
| Furniture | Wooden | Dinning Chair | 1 |
| Furniture | Wooden | Dinning Table | 1 |
| Furniture | Wooden | Double Bed | 2 |
| Furniture | Wooden | Side Table | 1 |
| Furniture | Wooden | Sofa | 1 |
| Furniture | Wooden | Tea Table | 2 |
| Stationary | Plastic | Pen | 2 |
| Stationary | Wooden | Pencil | 1 |

18. Write a query to display employee id, employee name, item id and item description of employees who were issued item(s) in the month of January 2013. Display the records sorted in order based on employee id and then by item id in ascending order.

```
SELECT e.employee_id,employee_name,i.item_id,i.item_description FROM
employee_master e JOIN employee_issue_details ei ON e.employee_id=ei.employee_id
JOIN item_master i ON i.item_id=ei.item_id
WHERE month(ei.issue_date)='01' and year(ei.issue_date)='2013'
ORDER BY employee_id,item_id;
```

| employee_id | employee_name | item_id | item_description |
|-------------|---------------|---------|------------------|
| E00002 | Abhay | I00005 | Side Table |

19. Write a query to display the employee id, employee name and count of item category of the employees who have been issued items in at least 2 different categories. Give the alias name for category count as COUNT_CATEGORY. Display the records sorted in ascending order based on employee id.

```
SELECT ei.employee_id,e.employee_name,count(DISTINCT i.item_category) COUNT_CATEGORY FROM
employee_master e JOIN employee_issue_details ei ON e.employee_id=ei.employee_id
JOIN item_master i ON i.item_id=ei.item_id
```


GROUP BY ei.employee_id

HAVING COUNT_CATEGORY>=2

ORDER BY employee_id;

| employee_id | employee_name | COUNT_CATEGORY |
|-------------|---------------|----------------|
| E00004 | Zuben | 2 |

20. Write a query to display the item id , item description which was never issued to any employee. Display the records sorted in ascending order based on item id.

SELECT item_id, item_description FROM item_master

WHERE item_id NOT IN (SELECT item_id from employee_issue_details)

ORDER BY item_id;

| item_id | item_description |
|---------|------------------|
| I00002 | Dinning Table |
| I00003 | Tea Table |
| I00006 | Tea Table |
| I00009 | Sofa |
| I00011 | Cupboard |
| I00013 | Double Bed |
| I00014 | Single Bed |
| I00015 | Single Bed |
| I00016 | Tea Set |
| I00017 | Tea Set |
| I00019 | Dinning Set |
| I00020 | Pencil |
| I00021 | Pen |
| I00022 | Pen |
| NULL | NULL |

21. Write a query to display the employee id, employee name and total valuation for the employees who has issued minimum total valuation of the product. Give the alias name for total valuation as TOTAL_VALUATION.[Hint: Suppose an employee E00019 issued item of price 5000, 10000, 12000 and E00020 issue item of price 2000, 7000 and 1000. So the valuation of items taken by E00019 is 27000 and for E00020 it is 10000. So the employee id, employee name of E00020 should be displayed.]

SELECT e.employee_id, em.employee_name, sum(i.item_valuation) TOTAL_VALUATION FROM

item_master i JOIN employee_issue_details e ON e.item_id=i.item_id

JOIN employee_master em ON em.employee_id=e.employee_id

```

GROUP BY e.employee_id HAVING sum(i.item_valuation)<=ALL(
SELECT sum(i.item_valuation) TOTAL_VALUATION FROM
item_master i JOIN employee_issue_details e ON e.item_id=i.item_id
JOIN employee_master em ON em.employee_id=e.employee_id
GROUP BY e.employee_id);

```

| employee_id | employee_name | TOTAL_VALUATION |
|-------------|---------------|-----------------|
| E00002 | Abhay | 1500 |

22. Write a query to display the employee id, employee name, card issue date and card valid date. Order by employee name and then by card valid date. Give the alias name to display the card valid date as CARD_VALID_DATE. [Hint: Validity in years for the loan card is given in loan_card_master table. Validity date is calculated by adding number of years in the loan card issue date. If the duration of year is zero then display AS 'No Validity Date'.]

```

SELECT e.employee_id,e.employee_name,card_issue_date,
case
when l.duration_in_years>0 then date_add(ec.card_issue_date,interval l.duration_in_years year)
when l.duration_in_years=0 then 'No Validity Date' end CARD_VALID_DATE
FROM
employee_master e JOIN employee_card_details ec ON e.employee_id=ec.employee_id
JOIN loan_card_master l ON l.loan_id=ec.loan_id
ORDER BY employee_name,CARD_VALID_DATE;

```

| employee_id | employee_name | card_issue_date | CARD_VALID_DATE |
|-------------|---------------|-----------------|------------------|
| E00002 | Abhay | 2007-02-01 | 2012-02-01 |
| E00002 | Abhay | 2007-03-11 | No Validity Date |
| E00003 | Anita | 2007-04-15 | 2008-04-15 |
| E00003 | Anita | 2007-04-15 | 2012-04-15 |
| E00003 | Anita | 2007-04-15 | No Validity Date |
| E00001 | Ram | 2002-12-14 | 2003-12-14 |
| E00001 | Ram | 2000-01-01 | 2005-01-01 |
| E00001 | Ram | 2000-01-01 | No Validity Date |

23. Write a query to display the employee id, employee name who have not issued with any item in the year 2013. Hint: Exclude those employees who was never issued with any of the items in all the years. Display the records sorted in ascending order based on employee id.

```

SELECT DISTINCT e.employee_id,e.employee_name FROM

```

```

employee_master e JOIN employee_issue_details ei ON e.employee_id=ei.employee_id
WHERE e.employee_id NOT IN (SELECT employee_id FROM employee_issue_details
WHERE year(issue_date)='2013')
ORDER BY employee_id;

```

| employee_id | employee_name |
|-------------|---------------|
| E00001 | Ram |
| E00003 | Anita |
| E00006 | John |

24. Write a query to display issue id, employee id, employee name, item id, item description and issue date. Display the data in descending order of date and then by issue id in ascending order.

```

SELECT issue_id, eid.employee_id, employee_name, im.item_id, item_description, issue_date
FROM employee_issue_details eid JOIN employee_master em ON eid.employee_id=em.employee_id
JOIN item_master im ON eid.item_id=im.item_id
ORDER BY issue_date DESC, issue_id;

```

| issue_id | employee_id | employee_name | item_id | item_description | issue_date |
|----------|-------------|---------------|---------|------------------|------------|
| ISS009 | E00004 | Zuben | I00018 | Dinning Set | 2013-04-18 |
| ISS007 | E00004 | Zuben | I00012 | Double Bed | 2013-04-14 |
| ISS003 | E00002 | Abhay | I00005 | Side Table | 2013-01-03 |
| ISS008 | E00006 | John | I00018 | Dinning Set | 2012-08-18 |
| ISS006 | E00003 | Anita | I00010 | Cupboard | 2012-03-14 |
| ISS001 | E00001 | Ram | I00001 | Tea Table | 2012-02-03 |
| ISS002 | E00001 | Ram | I00004 | Side Table | 2012-02-03 |
| ISS004 | E00003 | Anita | I00007 | Dinning Chair | 2010-07-04 |
| ISS005 | E00003 | Anita | I00008 | Tea Table | 2010-07-04 |

25. Write a query to display the employee id, employee name and total valuation for employee who has issued maximum total valuation of the product. Give the alias name for total valuation as TOTAL_VALUATION.[Hint: Suppose an employee E00019 issued item of price 5000, 10000, 12000 and E00020 issue item of price 2000, 7000, and 1000. So the valuation of items taken by E00019 is 27000 and for E00020 it is 10000. So the employee id, employee name and total valuation of E00019 should display.]

```

SELECT e.employee_id, em.employee_name, sum(i.item_valuation) TOTAL_VALUATION FROM
item_master i JOIN employee_issue_details e ON e.item_id=i.item_id
JOIN employee_master em ON em.employee_id=e.employee_id
GROUP BY e.employee_id HAVING sum(i.item_valuation)>=ALL(

```

```
SELECT sum(i.item_valuation) TOTAL_VALUATION FROM  
item_master i JOIN employee_issue_details e ON e.item_id=i.item_id  
JOIN employee_master em ON em.employee_id=e.employee_id  
GROUP BY e.employee_id);
```

| employee_id | employee_name | TOTAL_VALUATION |
|-------------|---------------|-----------------|
| E00004 | Zuben | 25500 |