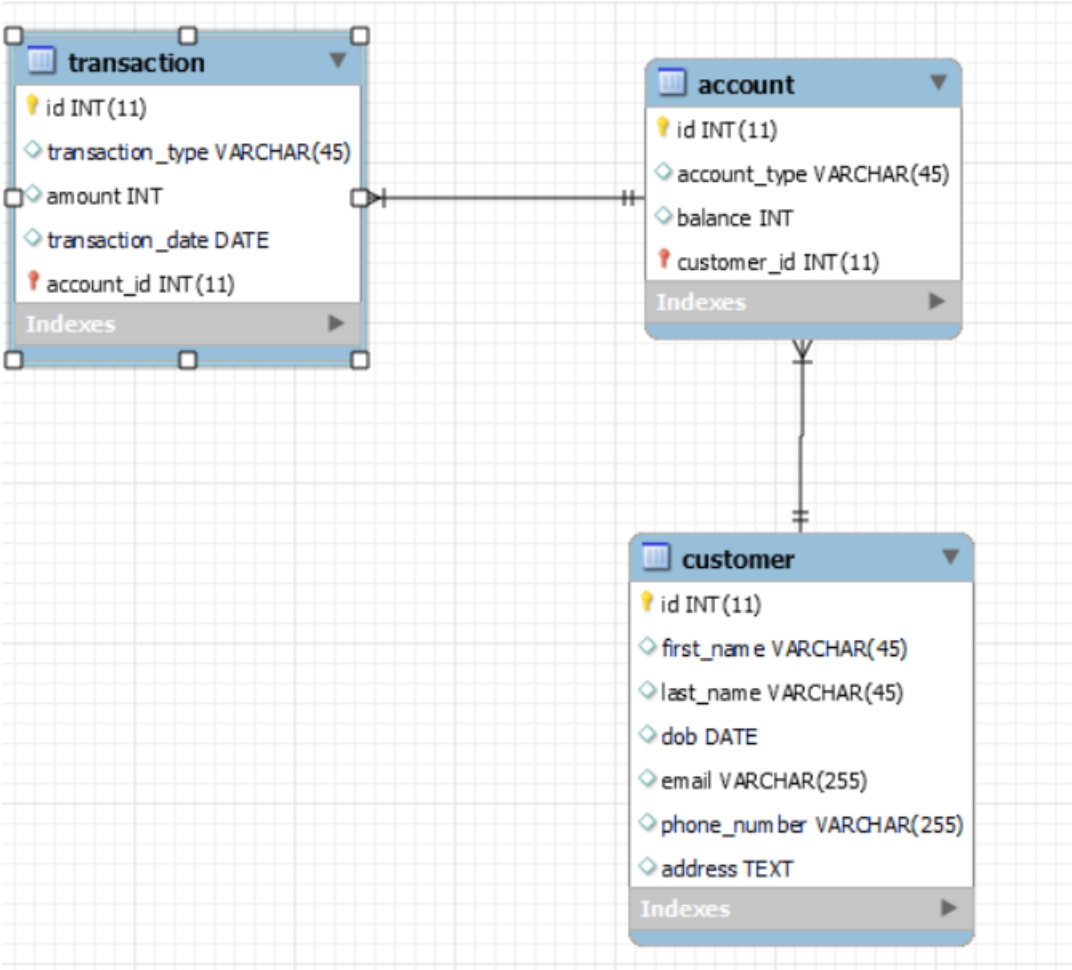
ASSIGNMENT NO : 3

Banking System

ER Diagram:



Task:1. Database Design:

-- MySQL Workbench Forward Engineering

-- ---

-- Schema hmbank

-- ---

-- ---

-- Schema hmbank

-- ---

CREATE SCHEMA IF NOT EXISTS hmbank DEFAULT CHARACTER SET utf8 ;

USE hmbank;

```
-- -----------------------------------------------------
-- Tablehmbank.customer
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS hmbank.customer (
  id INT NOT NULL AUTO_INCREMENT,
  first_name VARCHAR(255) NOT NULL,
  last_name VARCHAR(255) NOT NULL,
  dob DATE NULL,
  email VARCHAR(255) NULL,
  phone_number VARCHAR(20) NULL,
  address TEXT NULL,
  PRIMARY KEY (id),
  UNIQUE INDEX email_UNIQUE (email ASC) )
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table hmbank.account
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS hmbank.account (
  id INT NOT NULL AUTO_INCREMENT,
  account_type VARCHAR(255) NOT NULL,
  balance INT NULL,
  customer_id INT NOT NULL,
  PRIMARY KEY (id, customer_id),
  INDEX fk_account_customer1_idx (customer_id ASC) ,
  CONSTRAINT fk_account_customer1
    FOREIGN KEY (customer_id)
    REFERENCES hmbank.customer (id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
```

```
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table hmbank.transaction
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS hmbank.transaction (
  id INT NOT NULL AUTO_INCREMENT,
  transaction_type VARCHAR(255) NOT NULL,
  amount INT NULL,
  transaction_date DATE NULL,
  account_id INT NOT NULL,
  PRIMARY KEY (id, account_id),
  INDEX fk_transaction_account_idx (account_id ASC) ,
  CONSTRAINT fk_transaction_account
    FOREIGN KEY (account_id)
    REFERENCES hmbank.account (id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

use hmbank;
```

INSERTION:

-- customer table

```
insert into customer(first_name,last_name,dob,email,phone_number,address)values
('Ram','Prasad','2002-03-30','ram@gmail.com','9024554745','white town'),
('Sandiya','Vishwanath','2002-08-25','sandiya@gmail.com','9174543526','anna nagar'),
('Jayanthi','Selvam','2002-08-25','selvam@gmail.com','9082707895','nehru nagar'),
('Swetha','Seetharaman','2005-04-11','swetha@gmail.com','7098645321','sowkarpet'),
('Divya','Dharshini','2004-06-14','divya@gmail.com','9123765480','semmandalam'),
```

('Nisha','Vaithiyanathan','2000-07-14','nisha@gmail.com','9865432178','manjakupam'),

('Darshini','Balamurali','2001-07-15','darshnini@gmail.com','709834521','main street'),

('Agalya','Shanmugam','2002-12-07','agalya@gmail.com','8143256790','ranipet'),

('Harini','Murugavel','2002-12-16','harini@gmail.com','9024554745','madagadipet'),

('Selva','Ramaiah','1998-08-12','selva@gmail.com','9156473420','gandhi park');

```
mysql> select * from customer;
+----+------------+----------------+------------+--------------------+--------------+-------------+
| id | first_name | last_name      | dob        | email              | phone_number | address     |
+----+------------+----------------+------------+--------------------+--------------+-------------+
|  1 | Ram        | Prasad         | 2002-03-30 | ram@gmail.com      | 9024554745   | white town  |
|  2 | Sandiya    | Vishwanath     | 2002-08-25 | sandiya@gmail.com  | 9174543526   | anna nagar  |
|  3 | Jayanthi   | Selvam         | 2002-08-25 | selvam@gmail.com   | 9082707895   | nehru nagar |
|  4 | Swetha     | Seetharaman    | 2005-04-11 | swetha@gmail.com   | 7098645321   | sowkarpet   |
|  5 | Divya      | Dharshini      | 2004-06-14 | divya@gmail.com    | 9123765480   | semmandalam |
|  6 | Nisha      | Vaithiyanathan | 2000-07-14 | nisha@gmail.com    | 9865432178   | manjakupam  |
|  7 | Darshini   | Balamurali     | 2001-07-15 | darshnini@gmail.com | 709834521    | main street |
|  8 | Agalya     | Shanmugam      | 2002-12-07 | agalya@gmail.com   | 8143256790   | ranipet     |
|  9 | Harini     | Murugavel      | 2002-12-16 | harini@gmail.com   | 9024554745   | madagadipet |
| 10 | Selva      | Ramaiah        | 1998-08-12 | selva@gmail.com    | 9156473420   | gandhi park |
+----+------------+----------------+------------+--------------------+--------------+-------------+
```

-- account table

insert into account(account_type,balance,customer_id)values
('savings',50000,1),

('current',1200,2),

('zero_balance',100000,3),

('current',150000,1),

('savings',0,7),

('savings',300,3),

('savings',55000,4),

('current',133000,5),

('zero_balance',500000,6),

('current',220000,9),

('savings',60000,10);

```
mysql> select * from account;
+----+--------------+---------+-------------+
| id | account_type | balance | customer_id |
+----+--------------+---------+-------------+
|  1 | savings      |   50000 |           1 |
|  2 | current      |    1200 |           2 |
|  3 | zero_balance |  100000 |           3 |
|  4 | current      |  150000 |           1 |
|  5 | savings      |       0 |           7 |
|  6 | savings      |     300 |           3 |
|  7 | savings      |   55000 |           4 |
|  8 | current      |  133000 |           5 |
|  9 | zero_balance |  500000 |           6 |
| 10 | current      |  220000 |           9 |
| 11 | savings      |   60000 |          10 |
+----+--------------+---------+-------------+
```

-- transaction table

insert into transaction(transaction_type,amount,transaction_date,account_id)
values
('deposit',10000,'2024-02-01',1),

('withdrawal',5000,'2023-04-27',2),

('deposit',20000,'2024-01-02',2),

('withdrawal',8000,'2024-05-15',3),

('transfer',30000,'2024-02-01',4),

('transfer',7000,'2023-02-05',5),

('deposit',67000,'2024-12-01',6),

('withdrawal',9000,'2024-11-02',7),

('deposit',50000,'2024-02-08',8),

('transfer',10000,'2024-02-02',10);

```
mysql> select * from transaction;
+----+------------------+--------+------------------+------------+
| id | transaction_type | amount | transaction_date | account_id |
+----+------------------+--------+------------------+------------+
|  1 | deposit          |  10000 | 2024-02-01       |          1 |
|  2 | withdrawal       |   5000 | 2023-04-27       |          2 |
|  3 | deposit          |  20000 | 2024-01-02       |          2 |
|  4 | withdrawal       |   8000 | 2024-05-15       |          3 |
|  5 | transfer         |  30000 | 2024-02-01       |          4 |
|  6 | transfer         |   7000 | 2023-02-05       |          5 |
|  7 | deposit          |  67000 | 2024-12-01       |          6 |
|  8 | withdrawal       |   9000 | 2024-11-02       |          7 |
|  9 | deposit          |  50000 | 2024-02-08       |          8 |
| 10 | transfer         |  10000 | 2024-02-02       |         10 |
+----+------------------+--------+------------------+------------+
```

-- Tasks 2: Select, Where, Between, AND, LIKE:

-- 1. Insert at least 10 sample records into each of the following tables.
-- =>inserted

-- 2. Write SQL queries for the following tasks:

-- 1. Write a SQL query to retrieve the name, account type and email of all customers.

```
select concat(c.first_name,c.last_name)as name,a.account_type,c.email
from customer c,account a
where c.id=a.customer_id;
```

-- 2. Write a SQL query to list all transaction corresponding customer.

```
select concat(c.first_name,' ',c.last_name)as name,t.transaction_type,t.amount
from customer c,account a,transaction t
where c.id=a.customer_id and a.id=t.account_id;
```

/* 3. Write a SQL query to increase the balance of a specific account by a certain amount. */

```
update account
set balance = balance + 100
where id =5;
```

-- 4. Write a SQL query to Combine first and last names of customers as a full_name.

```
select concat(first_name ,' ' ,last_name)as full_name
from customer;
```

/* 5. Write a SQL query to remove accounts with a balance of zero where the account type is savings. */

```
delete
from account
where account_type='savings' and balance=0;
```

-- 6. Write a SQL query to Find customers living in a specific city.

```
select concat(first_name,' ',last_name)as names
```

```
from customer
where address='main street';
```

-- 7. Write a SQL query to Get the account balance for a specific account.

```
select id,balance
from account
where id=5;
```

-- 8. Write a SQL query to List all current accounts with a balance greater than $1,000.

```
select id,balance
from account
where balance >1000;
```

-- 9. Write a SQL query to Retrieve all transactions for a specific account.

```
select id,transaction_type,amount,transaction_date
from transaction
where account_id=2;
```

/* 10. Write a SQL query to Calculate the interest accrued on savings accounts based on a given interest rate. */

```
select id,balance*0.5 as 'savings_amount'
from account
where account_type='savings';
```

/* 11. Write a SQL query to Identify accounts where the balance is less than a specified overdraft limit. */

```
select * from account
where balance <3000;
```

-- 12. Write a SQL query to Find customers not living in a specific city.

```
select concat(first_name,' ',last_name)as names
from customer
where address !='main street';
```

-- Tasks 3: Aggregate functions, Having, Order By, GroupBy and Joins:

-- 1. Write a SQL query to Find the average account balance for all customers.

```
select customer_id,avg(balance)
from account
group by customer_id;
```

-- 2. Write a SQL query to Retrieve the top 10 highest account balances.

```
select id,balance
from account
order by balance desc
limit 0,10;
```

-- 3. Write a SQL query to Calculate Total Deposits for All Customers in specific date.

```
select c.first_name, c.last_name, SUM(t.amount) AS total_deposits
from customer c join account a on c.id = a.customer_id join transaction t on a.id =
t.account_id
where t.transaction_type = 'deposit' and t.transaction_date = '2024-02-01';
```

-- 4. Write a SQL query to Find the Oldest and Newest Customers.

```
(select first_name,dob,'oldest' as status from customer order by dob limit 0,1)
UNION
(select first_name,dob,'youngest' as status from customer order by dob DESC limit 0,1);
```

-- 5. Write a SQL query to Retrieve transaction details along with the account type.

```
select t.transaction_type,t.amount,t.transaction_date,a.account_type
from transaction t,account a
where a.id=t.account_id;
```

-- 6. Write a SQL query to Get a list of customers along with their account details.

```
select c.first_name,c.last_name,a.account_type,a.balance
from customer c, account a
where c.id=a.customer_id;
```

/* 7. Write a SQL query to Retrieve transaction details along with customer information for a specific account. */

```
select
t.transaction_type,t.amount,t.transaction_date,c.first_name,c.last_name,c.dob,a.account_type
from transaction t join account a on t.account_id join customer c on c.id=a.customer_id
WHERE a.id=1;
```

-- 8. Write a SQL query to Identify customers who have more than one account.

```
select c.first_name,count(c.id) as number_of_accounts
from customer c JOIN account a ON c.id = a.customer_id
group by a.customer_id
having number_of_accounts>1;
```

/* 9. Write a SQL query to Calculate the difference in transaction amounts between deposits and withdrawals. */

```
select
((select SUM(amount)
from transaction
where transaction_type ='deposit' ) -
(select  SUM(amount)
from transaction
where transaction_type ='withdrawal')) as diff;
```

/* 10. Write a SQL query to Calculate the average daily balance for each account over a specified period. */

```
select a.id, avg(balance) as avg_balance
from account a join transaction t on a.id = t.account_id
where transaction_date between '2024-02-01' and '2024-02-02'
group by a.id ;
```

-- 11. Calculate the total balance for each account type.

```
select account_type,sum(balance)as total_balance
from account
group by account_type;
```

/* 12. Identify accounts with the highest number of transactions order by descending order. */

```
select a.id,count(a.id) as highest_transaction
from transaction t join account a on t.account_id=a.id
group by a.id
order by highest_transaction desc;
```

/* 13. List customers with high aggregate account balances, along with their account types. */

```
select concat(c.first_name,' ',c.last_name)as name,max(a.balance) as balance,a.account_type
from customer c join account a on a.customer_id=c.id;
```

/* 14. Identify and list duplicate transactions based on transaction amount, date, and account. */

```
select amount,transaction_date,account_id , count(*)
from transaction t
group by amount,transaction_date,account_id
having count(*)>1;
```

-- Tasks 4: Subquery and its type:

-- 1. Retrieve the customer(s) with the highest account balance.

```
select c.id,c.first_name
from customer c join account a on c.id=a.customer_id
where a.balance in(
                select max(balance)
                from account);
```

/* 2. Calculate the average account balance for customers who have more than one account. */

```
select avg(balance)
            from account
            where customer_id in(select customer_id
            from account
            group by customer_id
            having count(id)>1);
```

/* 3. Retrieve accounts with transactions whose amounts exceed the average transaction amount. */

```
select *
from account a
where EXISTS (
   select amount
   from transaction t
   where t.account_id = a.id AND t.amount > (
      select avg(amount)
      from transaction
      where id = a.id));
```

-- 4. Identify customers who have no recorded transactions.

```
select c.*
from customer c
where NOT EXISTS (
   select 1
   from account a
   where a.customer_id = c.id
);
```

-- 5. Calculate the total balance of accounts with no recorded transactions.

```
select a.id,sum(a.balance)as total_balance
from account a join transaction t on a.id=t.account_id
where a.id not in(
select t.account_id from transaction t);
```

-- 6. Retrieve transactions for accounts with the lowest balance.

```
select t.*
from transaction t join  account a on t.account_id = a.id
order by a.balance
limit 0,1;
```

-- 7. Identify customers who have accounts of multiple types.

```
select c.*
from customer c join account a on c.id=a.customer_id
group by a.customer_id
having count(a.customer_id)>1;
```

-- 8. Calculate the percentage of each account type out of the total number of accounts.

```
select account_type, COUNT(*) AS account_count,COUNT(*) / (
                        select COUNT(*) from account) * 100 as percentage
from account
group by account_type;
```

-- 9. Retrieve all transactions for a customer with a given customer_id.

```
select *
from transaction
where account_id IN (
            select id from account where customer_id = 1);
```

/* 10. Calculate the total balance for each account type, including a subquery within the SELECT clause. */

```
select account_type,
    (select SUM(balance) from account
    where account_type = a.account_type) as total_balance
from account a
group by account_type;
```