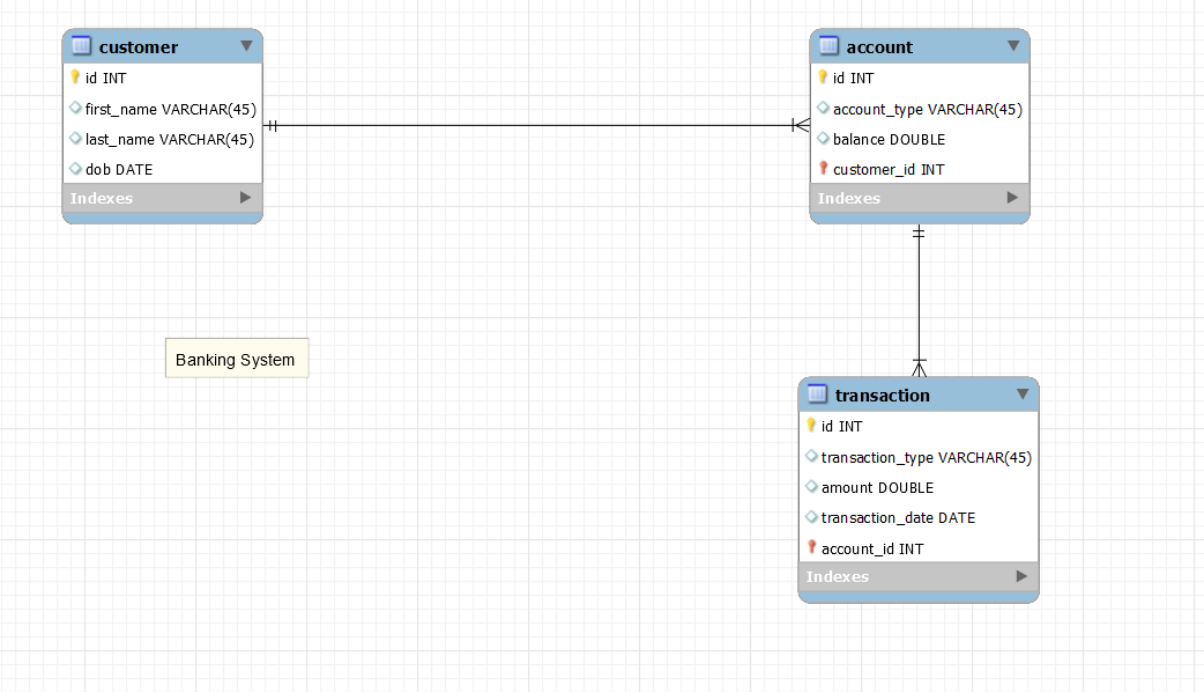
Banking System



-- TASK 1

create database HMBank;

use HMBank;

CREATE TABLE Customers

(CUSTOMER_ID int PRIMARY KEY,

FIRST_NAME VARCHAR(10),

LAST_NAME VARCHAR(9),

DATE_OF_BIRTH date,

EMAIL VARCHAR(25),

PHONE_NUMBER bigint,

ADDRESS VARCHAR(50));

```
CREATE TABLE ACCOUNTS
(ACCOUNT_ID int PRIMARY KEY,
CUSTOMER_ID int,
FOREIGN KEY (CUSTOMER_ID) references CUSTOMERS(CUSTOMER_ID),
ACCOUNT_TYPE VARCHAR(15),
BALANCE BIGINT);

CREATE TABLE TRANSACTIONS
(TRANSACTION_ID int PRIMARY KEY,
ACCOUNT_ID int,
FOREIGN KEY (ACCOUNT_ID) references ACCOUNTS(ACCOUNT_ID),
transaction_type varchar(25),
amount BIGINT,
TRANSACTION_DATE date);

drop table transactions;

-- TASK - 2

-- 1. Insert at least 10 sample records into each of the following tables.• Customers• Accounts• Transactions

INSERT INTO CUSTOMERS
(CUSTOMER_id,first_name,last_name,date_of_birth,email,phone_number,ADDRESS)
VALUES
(1, 'John', 'Doe', '1990-01-15', 'john@gmail.com', 1234567890,'Chetty Street'),
(2, 'Jane', 'Smith', '1992-05-20', 'jane@gmail.com', 2345678901,'Bharathi Street'),
(3, 'Anu', 'Kaviya', '1997-07-12', 'dhivya@gmail.com', 3456789012,'KV nagar'),
(4, 'Dhivya', 'Dharshini', '1991-11-2', 'darshh@gmail.com', 4567890123,'RK Nagar'),
```

```
(5, 'Divya', 'Praba', '1995-02-11', 'praba@gmail.com', 5678901234,'Anand Nagar'),
(6, 'Arun', 'Kumar', '1999-04-19', 'arun@gmail.com', 6789012345,'Kathirkamam'),
(7, 'Raj', 'Ram', '1993-03-28', 'raja@example.com', 7890123456,'Barathi Street'),
(8, 'Vimal', 'Raj', '1994-06-16', 'kanna@gmail.com', 8901234567,'KK Nagar'),
(9, 'Devi', 'Bala', '1998-10-25', 'devi@example.com', 9012345678,'Prince town'),
(10, 'Vedha', 'Ratchana', '2001-11-19', 'vratchana@gmail.com', 1123456789,'White town');

update customers set address='Bharathi Street' where customer_id=7;

insert into customers values(11,'Harini','Murugan','2002-12-
21','harini@gmail.com',1234123421,'white town');

insert into accounts (account_id,customer_id,account_type,balance) values
(101,1,'savings',140000),
(102,2,'current',50000),
(103,3,'zero_balance',0),
(104,4,'savings',550000),
(105,5,'current',35000),
(106,6,'zero_balance',0),
(107,7,'savings',200000),
(108,8,'current',20000),
(109,9,'zero_balance',0),
(110,10,'savings',320000);

update accounts set account_type='deposits' where customer_id=3;
update accounts set balance = 30000 where customer_id=3;
update accounts set account_type='deposits' where customer_id=6;
update accounts set balance = 45000 where customer_id=6;
insert into accounts values(112,11,'current',32000);
insert into accounts values(111,11,'savings',320000);
```

```
insert into transactions
(transaction_id,account_id,transaction_type,amount,transaction_date) values
(001,101,'deposits',50000,'2020-11-02'),
(002,102,'deposits',60000,'2021-10-09'),
(003,103,'withdrawal',55000,'2019-06-17'),
(004,104,'withdrawal',40000,'2020-08-22'),
(005,105,'transfer',35000,'2021-11-28'),
(006,106,'transfer',45000,'2018-03-26'),
(007,107,'deposits',11000,'2020-07-04'),
(008,108,'deposits',20000,'2020-11-02'),
(009,109,'withdrawal',25000,'2023-09-19'),
(010,110,'transfer',30000,'2021-05-12');

insert into transactions values(011,111,'deposits',20000,'2020-11-13');

update transactions set transaction_date='2019-06-17' where account_id=109;

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

 select first_name, last_name, account_type, email from customers,accounts where
customers.customer_id = accounts.customer_id ;

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

select first_name, last_name from customers,transactions,accounts  where
transactions.account_id = accounts.account_id and customers.customer_id =
accounts.customer_id ;

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

```
update accounts set balance = balance + 20000 where account_id=101;

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

select concat(first_name ,' ',last_name) from customers;

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

delete from Accounts where account_type = 'savings' AND balance = 0;

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

select first_name, last_name from customers where address = 'Bharathi street';

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

select balance from accounts where account_type = 'current' and balance < 35000;

-- 8. Write a SQL query to List all current accounts with a balance greater than $1,000.
($1000 approximately 80000)

select * from accounts where balance>80000;

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

select * from transactions where account_id=102;

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

```
select (balance* 0.5) "Savimgs Amount" from accounts where account_type='savings';


-- 11. Write a SQL query to Identify accounts where the balance is less than a specified
overdraft limit.
-- lets take the overdraft limit as 300000

select * from accounts where balance <300000;


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


select first_name, last_name from customers where address != 'Bharathi street';


-----------------------------------------------------------------------------------------------------------
----------
-- TASK - 3

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


select avg(balance) from accounts ;


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


select account_id , balance from accounts order by balance desc limit 10;


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


select sum(amount) as Deposits from transactions where transaction_date='2020-11-02'
and transaction_type='deposits';
```

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

```
(select first_name,last_name,date_of_birth,'oldest' as status from customers order by
date_of_birth limit 0,1)

UNION

(select first_name,last_name,date_of_birth,'youngest' as status from customers order by
date_of_birth DESC limit 0,1);
```

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

```
Select  accounts.account_type, transactions.* from transactions, accounts where
accounts.account_id = transactions.account_id;
```

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

```
select customers.first_name, customers.last_name,accounts.* from customers,accounts
where customers.customer_id = accounts.customer_id;
```

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

```
select customers.customer_id, customers.first_name, transactions.* from customers ,
transactions , accounts where (customers.customer_id = accounts.customer_id) and
(accounts.account_id = transactions.account_id);
```

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

```
select customers.* from customers,accounts group by customer_id  having
count(account_id)>1;
```

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

```
select (select sum(amount) from transactions where transaction_type = 'deposits')-
(select sum(amount) from transactions where transaction_type = 'withdrawal') as
difference;


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


select accounts.account_id, avg(balance) from accounts join transactions on
accounts.account_id = transactions.account_id where transaction_date between '2021-10-
09' and '2022-09-19' group by account_id ;


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


select account_type , sum(balance) from accounts group by account_type;


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


select account_id , count(*) as transaction_count from transactions group by account_id
order by transaction_count desc ;


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


select customers.customer_id,  accounts.account_type ,sum(accounts.balance) as
aggregate_balance from customers join accounts on customers.customer_id =
accounts.customer_id group by customers.customer_id,accounts.account_type order  by
aggregate_balance desc;


-- 14. Identify and list duplicate transactions based on transaction amount, date, and
account.
```

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

 ----------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------
-----------------------------

 -- TASK - 4


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


 select customers.customer_id , concat(customers.first_name,customers.last_name) as
name , accounts.account_id, accounts.balance from accounts join customers on
customers.customer_id = accounts.customer_id order by accounts.balance desc limit 1;


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

   Select a.customer_id, c.first_name, Avg(balance) from accounts a JOIN customers c ON
c.customer_id = a.customer_id group by a.customer_id having count(a.account_id)>1;


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


  select account_id from transactions where amount > (select avg(amount) from
transactions);


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


  delete from transactions where account_id = 111; -- deleted to get tge customers who
have no records


  select distinct customers.customer_id , customers.first_name from accounts join
customers on customers.customer_id = accounts.account_id where accounts.account_id
not in (select account_id from transactions);
```

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

```
select  customer_id, (select first_name from customers  where customers.customer_id =
accounts.customer_id) first_name, account_id, balance from accounts where account_id
not in(select account_id from transactions) ;
```

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

```
select  transactions.* from accounts join transactions on accounts.account_id =
transactions.account_id where accounts.balance = (select min(balance) from accounts );
```

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

```
 select distinct accounts.customer_id from accounts group by accounts.customer_id having
count(distinct account_type)>1;
```

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

-- let the customer_id be 6

```
select transactions.* from transactions  where  account_id in (Select account_id from
accounts where customer_id = 6);
```

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

```
select account_type ,(select SUM(balance) from Accounts as B where B.account_type =
A.account_type) as total_balance
from Accounts as A group by account_type;
```