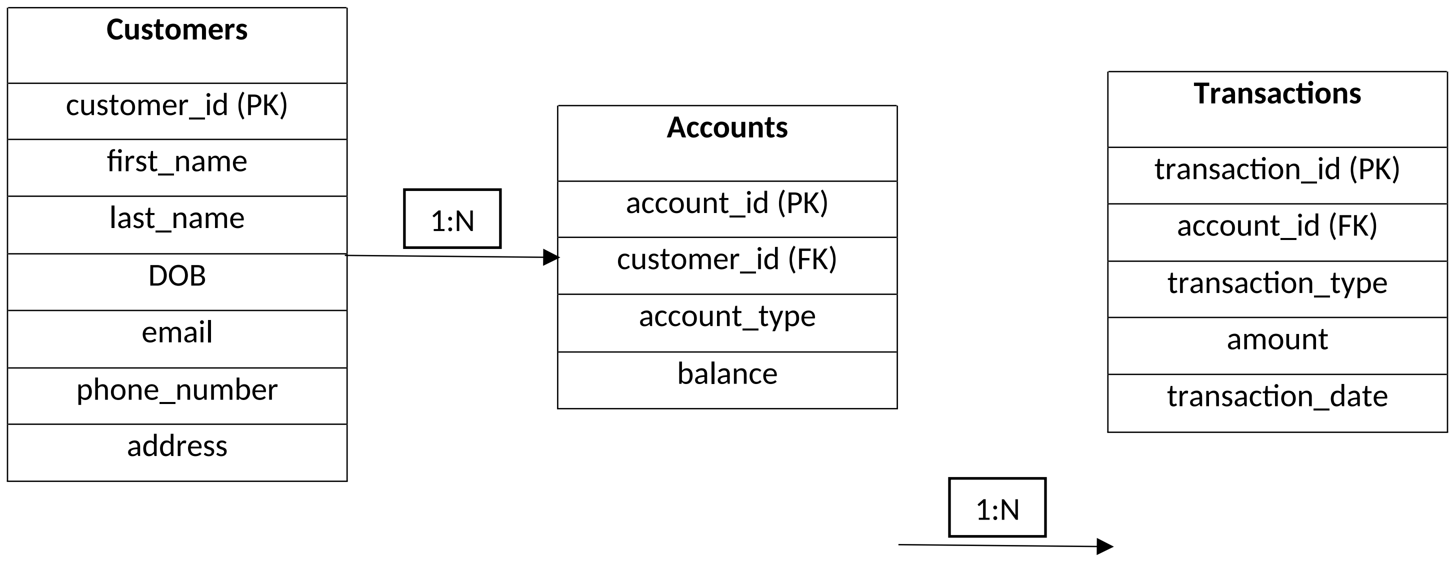
# ASSIGNMENT 1 – BANKING SYSTEM (SQL)

**By KARTHIKA KALIMUTHU, Excel Engineering College**

**-----------------------------------------------------Task 1: Database Design---------------------------------------------------------**

## ERD Diagram



create database HMBank

use HMBank

create table Customers (customer_id int primary key, first_name varchar (30), last_name varchar (30), DOB date, email varchar (100), phone_number int, address varchar (100))

create table Accounts (account_id int primary key, customer_id int, account_type varchar (30), balance int, foreign key(customer_id) references Customers(customer_id))

create table Transactions (transaction_id int primary key, account_id int, transaction_type varchar (20), amount int, transaction_date date, foreign key (account_id) references Accounts(account_id))

**--------------------------------------------Task 2: Select, Where, Between, AND, LIKE-----------------------------------------**

## 1. Insert at least 10 sample records into each of the tables

### Customers table

insert into Customers values (1, 'Dharun', 'Rohinth', '1990-01-15', 'dharun.rohinth@example.com', 987654321, 'chennai')

insert into Customers values (2, 'Maha', 'Lakshmi', '1985-03-22', 'maha.lakshmi@example.com', 912345678, 'coimbatore') insert into Customers values (3, 'Karthika', 'Kalimuthu', '2003-02-19', 'karthika.kalimuthu@example.com', 998877665, 'bangalore')

insert into Customers values (4, 'Ravi', 'Sharma', '1992-11-08', 'ravi.sharma@example.com', 978563421, 'delhi')

insert into Customers values (5, 'Sneha', 'Patel', '1995-09-30', 'sneha.patel@example.com', 956183867, 'chennai')

insert into Customers values (6, 'Meena', 'Kumari', '1993-07-25', 'meena.kumari@example.com', 918273645, 'chennai')

insert into Customers values (7, 'Priya', 'Singh', '1991-05-27', 'priya.singh@example.com', 986745231, 'madurai')

insert into Customers values (8, 'Arjun', 'Varma', '1988-12-12', 'arjun.varma@example.com', 918273654, 'bangalore')

insert into Customers values (9, 'Divya', 'Bharathi', '1999-07-02', 'divya.bharathi@example.com', 912348765, 'hydrabad')

insert into Customers values (10, 'Yoga', 'Priya', '1997-03-12', 'yoga.priya@example.com', 902151922, 'salem')

select * from Customers

## Accounts table

insert into Accounts values (1001, 1, 'savings', 25000)

insert into Accounts values (1002, 2, 'current', 50000)

insert into Accounts values (1003, 2, 'zero_balance', 30000)

insert into Accounts values (1004, 4, 'current', 15000)

insert into Accounts values (1005, 5, 'savings', 40000)

insert into Accounts values (1006, 6, 'zero_balance', 10000)

insert into Accounts values (1007, 7, 'current', 20000)

insert into Accounts values (1008, 1, 'savings', 35000)

insert into Accounts values (1009, 9, 'zero_balance', 27000)

insert into Accounts values (1010, 5, 'savings', 22000)

select * from Accounts

## Transactions table

insert into Transactions values (1, 1001, 'deposit', 5000, '2025-06-01')

insert into Transactions values (2, 1001, 'withdraw', 3000, '2025-06-02')

insert into Transactions values (3, 1003, 'transfer', 7000, '2025-06-03')

insert into Transactions values (4, 1004, 'withdraw', 1500, '2025-06-04') insert into Transactions values (5, 1005, 'deposit', 10000, '2025-06-05')

insert into Transactions values (6, 1006, 'transfer', 2000, '2025-06-06')

insert into Transactions values (7, 1003, 'deposit', 8000, '2025-06-07')

insert into Transactions values (8, 1008, 'withdraw', 2500, '2025-06-08')

insert into Transactions values (9, 1007, 'transfer', 6000, '2025-06-09')

insert into Transactions values (10, 1010, 'withdraw', 1000, '2025-06-10')

select * from Transactions

## 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 first_name as name, email from Customers

select account_type from Accounts

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

select * from Customers, Accounts, Transactions

where Customers.customer_id= Accounts.customer_id and Accounts.account_id = Transactions.account_id

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

update Accounts set balance = balance + 1000 where account_id = 1004

### 4. Write a SQL query to combine first and last names of customers as a full_name.

select customer_id, first_name + last_name as full_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 customer_id, first_name, address from Customers

where address= 'chennai'
```

## 7. Write a SQL query to get the accounts balance for a specific account.

```
select account_id, balance from Accounts

where account_id = 1001
```

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

```
select account_type, balance from Accounts

where account_type='current' and balance > 1000
```

## 9. Write a SQL query to retrieve all transactions for a specific account

```
select account_id, transaction_type from Transactions

where account_id=1001
```

## 10. Write a SQL query to Calculate the interest accrued on savings account based on a given interest rate.

```
select account_id, balance, balance*0.05 as interest from Accounts

where account_type='savings'
```

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

```
select account_id, balance from Accounts

where balance < 20000     ---overdraft limit = 20000
```

## 12. Write a SQL query to find customers not living in a specific city.

```
select customer_id, first_name, address from Customers

where address != 'chennai'
```

**--------------------------------Task 3: Aggregate functions, Having, Order by, Group by and joins------------------------------**

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

```sql
select avg (balance) as Cus_Avg_balance from Accounts
```

2. Write a SQL query to retrieve the top 10 highest account balance.

```sql
select top (10) balance from Accounts
order by balance desc
```

3. Write a SQL query to calculate total deposits for all customers in specific date.

```sql
select sum (amount) as tot_dep from Transactions
where transaction_type = 'deposit' and transaction_date = '2025-06-05'
```

4. Write a SQL query to find the oldest and newest customers.

```sql
select top (1) first_name, DOB from Customers
order by DOB
select top (1) first_name, DOB from Customers
order by DOB desc
```

5. Write a SQL query to retrieve transactions details along with the account type.

```sql
select t.transaction_id, t.account_id, t.transaction_type, t.amount, t.transaction_date, a.account_type from Transactions as t
inner join Accounts as a
on t.transaction_id = a.customer_id
```

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

```sql
select c.first_name, a.account_id, a.customer_id, a.account_type, a.balance from Accounts as a
inner join Customers as c
on a.customer_id = c.customer_id
```

7. Write a SQL query to retrieve transaction details along with customer information for a specific amount.

```
select * from Customers as c
inner join Transactions as t
on c.customer_id = t.transaction_id
where amount=8000
```

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

```
select c.customer_id, c.first_name, count (a.account_id) as no_of_acc from Customers as c
inner join Accounts as a
on c.customer_id = a.customer_id
group by a.customer_id, c.customer_id, c.first_name
having count (a.account_id) > 1
```

9. Write a SQL query to calculate the difference in transaction amounts between deposits and withdrawls.

```
select sum ( case when transaction_type = 'deposit' then amount else 0 end) - sum ( case when transaction_type =
'withdrawl' then amount else 0 end) as diff_amount from Transactions
```

11. Calculate the total balance for each account type.

```
select a.account_type, sum (a.balance) as tot_bal from Accounts as a
group by a.account_type
```

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

```
select count ( transaction_id ) trans_acc, account_id from Transactions
group by account_id
order by trans_acc desc
```

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

```
select top 1 (sum ( balance )) highest_bal, account_type from Accounts
group by account_type
order by highest_bal desc
```

**-------------------------------------------------------Task 4: Subquery and its types**: -------------------------------------------------------

1. Retrieve the customers with the highest account balance.

```
select * from Customers as c

inner join Accounts as a

on c.customer_id = a.customer_id

where a.balance = (select max (balance) Highest_bal from Accounts)
```

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

```
select avg (a.balance) as avg_bal from Accounts a

where a.customer_id in (select customer_id from Accounts group by customer_id having count (account_id) > 1)
```

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

```
select account_id , amount from Transactions

where amount > (select avg (amount) from Transactions)
```

4. Identify customers who have no recorded transactions.

```
select distinct * from Customers

where customer_id not in (select a.customer_id from Accounts a join Transactions t on a.account_id = t.account_id)
```

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

```
select sum (a.balance) as tot_bal from Accounts a

where a.account_id not in (select distinct account_id from Transactions)
```

6. Retrieve transactions for accounts with the lowest balance.

```
select t.transaction_id, t.account_id,t.transaction_type, a.balance from Transactions as t

join Accounts as a

on t.account_id = a.account_id

where balance = (select min (balance) from Accounts)
```

7. Identify customers who have accounts of multiple types.

```
select c.customer_id, c.first_name, c.last_name from Customers as c
join ( select customer_id from Accounts
    group by customer_id
          having count(distinct account_type) >1) as multi_acc
on c.customer_id = multi_acc.customer_id
```

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

```
select account_type, count (*)*100/ (select count (*) from Accounts) as percentage from Accounts
group by account_type
```

9. Retrieve all transaction for a customer with a given customer_id.

```
select t.transaction_id, t.amount from Transactions as t
join Accounts as a
on t.account_id = a.account_id
where a.customer_id = (select customer_id from Customers 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 Accounts a2 where a2.account_type = a1.account_type) as tot_bal from
Accounts a1
group by account_type
```