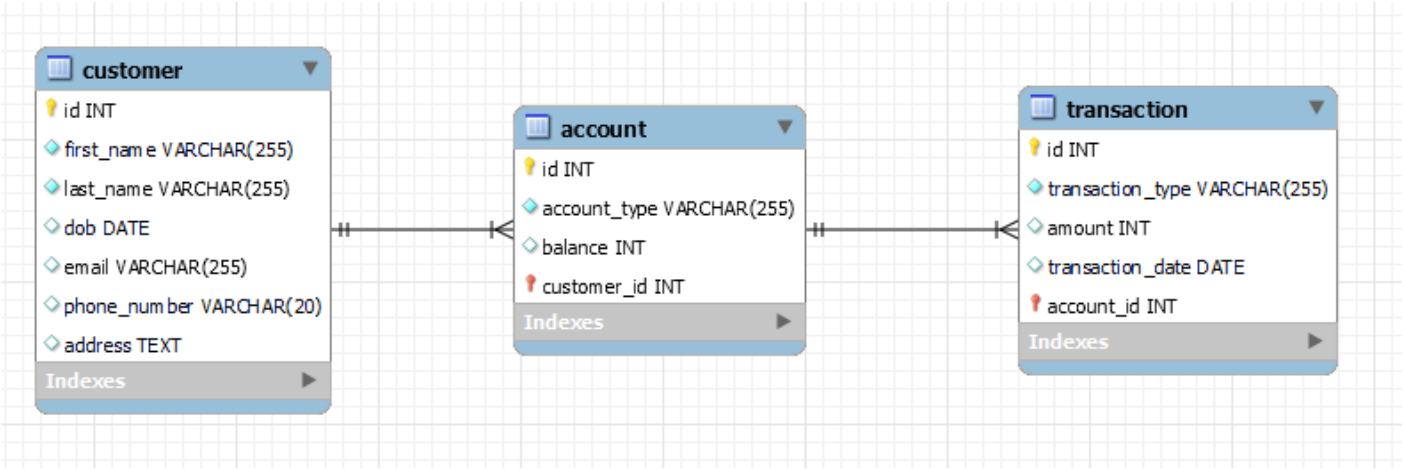
ASSIGNMENT 3

BANKING SYSTEM

ER DIAGRAM:



DATABASE DESIGN:

```
-- MySQL Workbench Forward Engineering

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

-- Schema banking_db

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


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

-- Schema banking_db

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

CREATE SCHEMA IF NOT EXISTS `banking_db` DEFAULT CHARACTER SET utf8 ;

USE `banking_db` ;


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

-- Table `banking_db`.`customer`

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

CREATE TABLE IF NOT EXISTS `banking_db`.`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 `banking_db`.`account`
-- -----------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_db`.`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 `banking_db`.`customer` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `banking_db`.`transaction`
-- -----------------------------------------------
CREATE TABLE IF NOT EXISTS `banking_db`.`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 `banking_db`.`account` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

INSERTION:

```
-- customer insertion
insert into customer(first_name,last_name,dob,email,phone_number,address)values
('dhana','lakshmi','2003-03-30','dhana@gmail.com','9360805403','ranipet'),
('dhivya','lakshmi','2002-08-25','dhivya@gmail.com','9786205333','mudhaliarpet'),
('dhivya','bharathy','2002-08-25','bharathy@gmail.com','9787333355','muthialpet'),
('bhavana','ranganathan','2003-06-12','bhavana@gmail.com','8220681348','john  paul'),
('pradheesha','sivakumar','2001-07-12','pradheesha@gmail.com','8072607205','anna  nagar'),
('kalis','dharani','1998-08-11','kalis@gmail.com','9443500160','T   nagar'),
('devibala','ponnusamy','1997-12-12','devibala@gmail.com','8610248302','rainbow  nagar'),
('dheepika','chellamuthu','2004-06-14','dheepika@gmail.com','9787333394','anna  nagar'),
('roshini','mani','2006-09-20','roshini@gmail.com','9787333394','ranipet'),
('hema','kannan','2005-04-11','hema@gmail.com','9787333364','muthialpet');

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

```
('current',120000,2),
('zero_balance',100000,3),
('savings',50000,4),
('savings',500000,5),
('savings',20000,6),
('savings',30000,7),
('savings',40000,8),
('savings',70000,9),
('savings',80000,10),
('current',150000,1),
('savings',30000,3),
('zero_balance',100000,6),
('zero_balance',200000,10),
('zero_balance',300000,9);

-- transaction insertion
insert into transaction(transaction_type,amount,transaction_date,account_id)
values
('deposit',10000,'2024-02-01',1),
('withdrawal',5000,'2024-02-02',1),
('deposit',20000,'2024-02-02',2),
('withdrawal',8000,'2024-02-02',3),
('transfer',20000,'2024-02-01',4),
('transfer',7000,'2024-02-05',5),
('deposit',20000,'2024-02-01',6),
('withdrawal',15000,'2024-02-02',7),
('transfer',2000,'2024-02-01',8),
('transfer',8000,'2024-02-05',9),
('deposit',30000,'2024-02-01',10);
```

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

-- 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.* from
customer c, transaction t ,account a
where a.customer_id=c.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+5000 where id=1;
```

-- 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. */

```
insert into account(account_type,balance,customer_id) values ('savings',0,9);

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

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

```
select * from customer where address='ranipet';

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

select id,balance from account where id=1;

-- 8. Write a SQL query to List all savings accounts with a balance greater than $100,000.

select * from account where balance>30000 and account_type='savings';

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

select * from transaction where account_id=1;

/* 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 interest 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<30000;

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

select * from customer where address!='ranipet';
```

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(customer_id) as average_account_balance from account
group by customer_id;

-- 2. Write a SQL query to Retrieve the top 5highest account balances.

select * from account
order by balance desc limit 0,5;

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

select * from transaction
where transaction_date='2024-02-01' and transaction_type='deposit';

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

(select first_name,last_name,dob, 'oldest_customer' as status from customer order by dob asc limit 0,1)
union all
(select first_name,last_name,dob,'newest_customer' 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.*,a.account_type from account a,transaction t
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 account a,customer c
where c.id=a.customer_id;
```

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

```
select c.first_name,c.last_name,a.account_type,a.balance,t.transaction_type,
t.transaction_date, t.amount from account a,customer c,transaction t
where c.id=a.customer_id and a.id=t.account_id;
```

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

```
select c.first_name,c.last_name, count(c.id) as no_of_accounts
from customer c,account a
where c.id=a.customer_id
group by c.id
having count(c.id)>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 difference_in_transaction_amounts_between_deposits_and_withdrawals;
```

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

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

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

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

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

select account_id ,count(account_id) as frequency from transaction
group by account_id
order by frequency desc;

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

select c.first_name,c.last_name,a.balance,a.account_type
from customer c ,account a
where c.id=a.customer_id
order by balance desc limit 0,1;

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

select amount,transaction_date,account_id,count(*)
from transaction
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 first_name,last_name
from customer where id =(select customer_id from account
                         where balance=(select max(balance) from account));
```

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

```
select customer_id,avg(balance) as avg_balance
from account
group by customer_id
having count(customer_id)>1;
```

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

```
select account_id from transaction
where amount>(select avg(amount) from transaction);
```

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

```
insert into customer values
(11,'priyan','sabarish','2003-03-30','sabarish@gmail.com', '9360805403','ranipet');

select first_name,last_name from
customer where id not in(select customer_id from account
where id in (select account_id from transaction));
```

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

```
insert into account values(16,'savings',500000,11);

select customer_id, (select first_name from customer
where customer.id = account.customer_id) first_name,id, balance from account
where id not in(select account_id from transaction) ;
```

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

```
select t.*
from transaction t where id =(select id from account
                              where balance=(select min(balance) from account));

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

select distinct account.customer_id from account
group by account.customer_id
having count(distinct account_type)>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 transaction.* from transaction where  account_id in (Select id from account where
customer_id = 6);

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

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