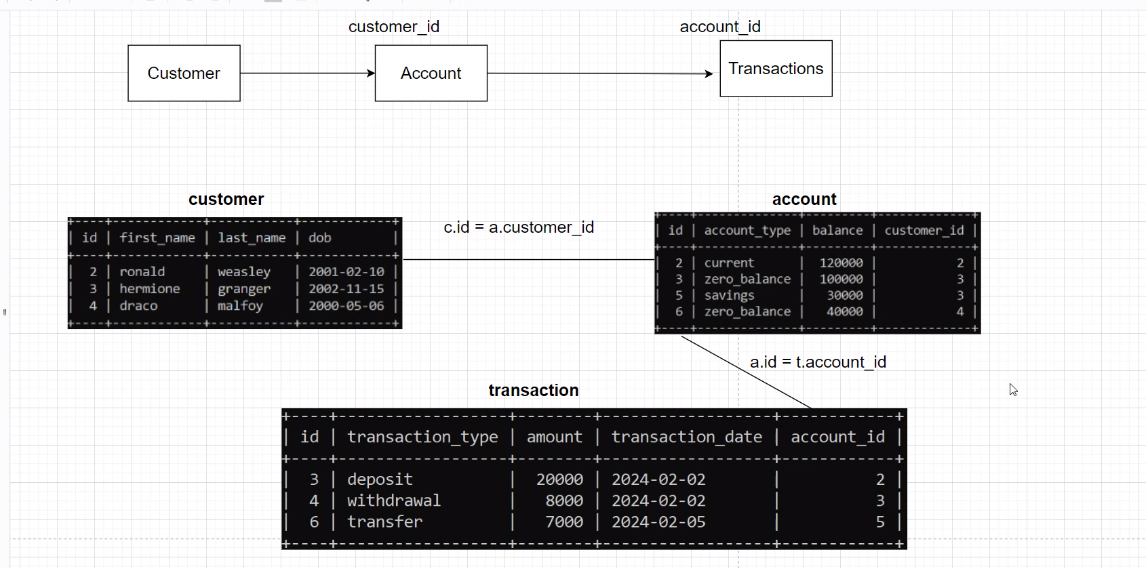
**ASSIGNMENT 2 : banking system**

TASK 1: DATABASE DESIGN



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

USE `BankingSys` ;

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

-- Table `BankingSys`.`customer`

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

CREATE TABLE IF NOT EXISTS `BankingSys`.`customer` (

`id` INT NOT NULL AUTO\_INCREMENT,

`first\_name` VARCHAR(45) NOT NULL,

`last\_name` VARCHAR(45) NOT NULL,

`dob` DATE NOT NULL,

PRIMARY KEY (`id`))

ENGINE = InnoDB;

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

-- Table `BankingSys`.`account`

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

CREATE TABLE IF NOT EXISTS `BankingSys`.`account` (

`id` INT NOT NULL AUTO\_INCREMENT,

`account\_type` VARCHAR(255) NOT NULL,

`balance` DOUBLE NULL,

`customer\_id` INT NOT NULL,

PRIMARY KEY (`id`),

INDEX `fk\_account\_customer\_idx` (`customer\_id` ASC) ,

CONSTRAINT `fk\_account\_customer`

FOREIGN KEY (`customer\_id`)

REFERENCES `BankingSys`.`customer` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

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

-- Table `BankingSys`.`transaction`

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

CREATE TABLE IF NOT EXISTS `BankingSys`.`transaction` (

`id` INT NOT NULL AUTO\_INCREMENT,

`transaction\_type` VARCHAR(45) NULL,

`transaction\_date` DATE NULL,

`amount` DOUBLE NULL,

`account\_id` INT NOT NULL,

PRIMARY KEY (`id`),

INDEX `fk\_transaction\_account1\_idx` (`account\_id` ASC) ,

CONSTRAINT `fk\_transaction\_account1`

FOREIGN KEY (`account\_id`)

REFERENCES `BankingSys`.`account` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

Use BankingSys;

insert into customer(first\_name,last\_name,dob) values

('harry','potter','2002-03-21'),

('ronald','weasley','2001-02-10'),

('hermione','granger','2002-11-15');

/\*

+----+------------+-----------+------------+

| id | first\_name | last\_name | dob |

+----+------------+-----------+------------+

| 1 | harry | potter | 2002-03-21 |

| 2 | ronald | weasley | 2001-02-10 |

| 3 | hermione | granger | 2002-11-15 |

+----+------------+-----------+------------+

\*/

insert into account(account\_type,balance,customer\_id) values

('savings',50000,1) ,

('current',120000,2) ,

('zero\_balance',100000,3),

('current',150000,1) ,

('savings',30000,3);

/\*

+----+--------------+---------+-------------+

| id | account\_type | balance | customer\_id |

+----+--------------+---------+-------------+

| 1 | savings | 50000 | 1 |

| 2 | current | 120000 | 2 |

| 3 | zero\_balance | 100000 | 3 |

| 4 | current | 150000 | 1 |

| 5 | savings | 30000 | 3 |

+----+--------------+---------+-------------+

\*/

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);

/\*

+----+------------------+--------+------------------+------------+

| id | transaction\_type | amount | transaction\_date | account\_id |

+----+------------------+--------+------------------+------------+

| 1 | deposit | 10000 | 2024-02-01 | 1 |

| 2 | withdrawal | 5000 | 2024-02-02 | 1 |

| 3 | deposit | 20000 | 2024-02-02 | 2 |

| 4 | withdrawal | 8000 | 2024-02-02 | 3 |

| 5 | transfer | 20000 | 2024-02-01 | 4 |

| 6 | transfer | 7000 | 2024-02-05 | 5 |

+----+------------------+--------+------------------+------------+

\*/

TASK 2:

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

**select c.first\_name, a.account\_type from customer c join account a on c.id=a.customer\_id group by c.id;**

/\*+------------+--------------+

| first\_name | account\_type |

+------------+--------------+

| harry | savings |

| ronald | current |

| hermione | zero\_balance |

+------------+--------------+\*/

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

**select c.first\_name, t.\* from customer c join account a on c.id=a.customer\_id join transaction t on a.id=t.account\_id;**

/\*+------------+----+------------------+------------------+--------+------------+

| first\_name | id | transaction\_type | transaction\_date | amount | account\_id |

+------------+----+------------------+------------------+--------+------------+

| harry | 1 | deposit | 2024-02-01 | 10000 | 1 |

| harry | 2 | withdrawal | 2024-02-02 | 5000 | 1 |

| harry | 5 | transfer | 2024-02-01 | 20000 | 4 |

| ronald | 3 | deposit | 2024-02-02 | 20000 | 2 |

| hermione | 4 | withdrawal | 2024-02-02 | 8000 | 3 |

| hermione | 6 | transfer | 2024-02-05 | 7000 | 5 |

+------------+----+------------------+------------------+--------+------------+\*/

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

**Update account**

**Set balance=60000**

**Where customer.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;**

/\*+------------------+

| full\_name |

+------------------+

| harry potter |

| ronald weasley |

| hermione granger |

+------------------+\*/

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

type is savings.

**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 dob='2001-02-10';**

/\*+----+------------+-----------+------------+

| id | first\_name | last\_name | dob |

+----+------------+-----------+------------+

| 2 | ronald | weasley | 2001-02-10 |

+----+------------+-----------+------------+\*/

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

**select balance from account where id = 2;**

/\*+---------+

| balance |

+---------+

| 120000 |

+---------+\*/

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

**select \* from account where account\_type='current' and balance>1000;**

/\*+----+--------------+---------+-------------+

| id | account\_type | balance | customer\_id |

+----+--------------+---------+-------------+

| 2 | current | 120000 | 2 |

| 4 | current | 150000 | 1 |

+----+--------------+---------+-------------+\*/

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

**select \* from transaction where amount = 20000;**

/\*+----+------------------+------------------+--------+------------+

| id | transaction\_type | transaction\_date | amount | account\_id |

+----+------------------+------------------+--------+------------+

| 3 | deposit | 2024-02-02 | 20000 | 2 |

| 5 | transfer | 2024-02-01 | 20000 | 4 |

+----+------------------+------------------+--------+------------+\*/

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

given interest rate.

**Not possible with given data**

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

overdraft limit.

**select \* from account where balance<50000;**

+----+--------------+---------+-------------+

| id | account\_type | balance | customer\_id |

+----+--------------+---------+-------------+

| 5 | savings | 30000 | 3 |

+----+--------------+---------+-------------+

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

-- not possible as city is not given in the data

TASK 3:

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

**select c.id , avg(a.balance) from customer c join account a on c.id=a.customer\_id group by c.id;**

/\*+----+----------------+

| id | avg(a.balance) |

+----+----------------+

| 1 | 100000 |

| 2 | 120000 |

| 3 | 65000 |

+----+----------------+\*/

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

**select \* from account order by balance desc limit 10;**

/\*+----+--------------+---------+-------------+

| id | account\_type | balance | customer\_id |

+----+--------------+---------+-------------+

| 4 | current | 150000 | 1 |

| 2 | current | 120000 | 2 |

| 3 | zero\_balance | 100000 | 3 |

| 1 | savings | 50000 | 1 |

| 5 | savings | 30000 | 3 |

+----+--------------+---------+-------------+\*/

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

**select c.id, sum(t.amount) from customer c left join account a on c.id=a.customer\_id left join transaction t on a.id=t.account\_id where transaction\_date ='2024-02-02'**

**and transaction\_type ='deposit';**

/\*+----+---------------+

| id | sum(t.amount) |

+----+---------------+

| 2 | 20000 |

+----+---------------+\*/

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

**select \* from customer order by dob;**

/\*+----+------------+-----------+------------+

| id | first\_name | last\_name | dob |

+----+------------+-----------+------------+

| 2 | ronald | weasley | 2001-02-10 |

| 1 | harry | potter | 2002-03-21 |

| 3 | hermione | granger | 2002-11-15 |

+----+------------+-----------+------------+\*/

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

**select t.\*, a.account\_type from transaction t join account a on a.id=t.account\_id;**

/\*+----+------------------+------------------+--------+------------+--------------+

| id | transaction\_type | transaction\_date | amount | account\_id | account\_type |

+----+------------------+------------------+--------+------------+--------------+

| 1 | deposit | 2024-02-01 | 10000 | 1 | savings |

| 2 | withdrawal | 2024-02-02 | 5000 | 1 | savings |

| 3 | deposit | 2024-02-02 | 20000 | 2 | current |

| 4 | withdrawal | 2024-02-02 | 8000 | 3 | zero\_balance |

| 5 | transfer | 2024-02-01 | 20000 | 4 | current |

| 6 | transfer | 2024-02-05 | 7000 | 5 | savings |

+----+------------------+------------------+--------+------------+--------------+\*/

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

**select \* from customer c left join account a on c.id=a.customer\_id;**

/\*+----+------------+-----------+------------+------+--------------+---------+-------------+

| id | first\_name | last\_name | dob | id | account\_type | balance | customer\_id |

+----+------------+-----------+------------+------+--------------+---------+-------------+

| 1 | harry | potter | 2002-03-21 | 1 | savings | 50000 | 1 |

| 1 | harry | potter | 2002-03-21 | 4 | current | 150000 | 1 |

| 2 | ronald | weasley | 2001-02-10 | 2 | current | 120000 | 2 |

| 3 | hermione | granger | 2002-11-15 | 3 | zero\_balance | 100000 | 3 |

| 3 | hermione | granger | 2002-11-15 | 5 | savings | 30000 | 3 |

+----+------------+-----------+------------+------+--------------+---------+-------------+\*/

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

specific account.

**select a.id,t.\*,c.\* from customer c left join account a on c.id=a.customer\_id left join transaction t on a.id=t.account\_id where a.id=4;**

/\*+----+------+------------------+------------------+--------+------------+----+------------+-----------+------------+

| id | id | transaction\_type | transaction\_date | amount | account\_id | id | first\_name | last\_name | dob |

+----+------+------------------+------------------+--------+------------+----+------------+-----------+------------+

| 4 | 5 | transfer | 2024-02-01 | 20000 | 4 | 1 | harry | potter | 2002-03-21 |

+----+------+------------------+------------------+--------+------------+----+------------+-----------+------------+\*/

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

**select customer\_id from account group by customer\_id having count(\*)>1;**

/\*+-------------+

| customer\_id |

+-------------+

| 1 |

| 3 |

+-------------+\*/

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

withdrawals.

**(select sum(amount) from transaction where transaction\_type='deposit')**

**UNION (select sum(amount) from transaction where transaction\_type ='withdrawal';**

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

period.

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

**select account\_type, sum(balance) from account group by account\_type;**

/\*+--------------+--------------+

| account\_type | sum(balance) |

+--------------+--------------+

| current | 270000 |

| savings | 80000 |

| zero\_balance | 100000 |

+--------------+--------------+\*/

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

**select a.id, count(\*) from account a join transaction t on a.id=t.account\_id group by a.id order by count(\*);**

/\*+----+----------+

| id | count(\*) |

+----+----------+

| 5 | 1 |

| 2 | 1 |

| 3 | 1 |

| 4 | 1 |

| 1 | 2 |

+----+----------+\*/

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

**select c.id,sum(a.balance) as aggregate, a.account\_type from customer c join account a on c.id=a.customer\_id group by c.id;**

/\*+----+-----------+--------------+

| id | aggregate | account\_type |

+----+-----------+--------------+

| 1 | 200000 | savings |

| 2 | 120000 | current |

| 3 | 130000 | zero\_balance |

+----+-----------+--------------+\*/

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

TASK 4:

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

**select c.\*,a.balance from customer c join account a on c.id=a.customer\_id order by a.balance desc limit 1;**

+----+------------+-----------+------------+---------+

| id | first\_name | last\_name | dob | balance |

+----+------------+-----------+------------+---------+

| 1 | harry | potter | 2002-03-21 | 150000 |

+----+------------+-----------+------------+---------+

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

select c.id, avg(a.balance) from customer c join account a on c.id=a.customer\_id group by c.id having count(a.id)>1;

+----+----------------+

| id | avg(a.balance) |

+----+----------------+

| 1 | 100000 |

| 3 | 65000 |

+----+----------------+

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

**select a.id from account a join transaction t on a.id=t.account\_id where t.amount>(select avg(t.amount) from transaction);**

/\*Empty set\*/

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

**select c.id from customer c join account a on c.id=a.customer\_id left join transaction t on a.id=t.account\_id where t.id is null;**

/\*Empty set\*/

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

**select \* from account where id NOT IN(select a.id from account a join transaction t on a.id=t.account\_id);**

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

select t.\*,a.balance from account a join transaction t on a.id=t.account\_id order by a.balance desc limit 1;

+----+------------------+------------------+--------+------------+---------+

| id | transaction\_type | transaction\_date | amount | account\_id | balance |

+----+------------------+------------------+--------+------------+---------+

| 5 | transfer | 2024-02-01 | 20000 | 4 | 150000 |

+----+------------------+------------------+--------+------------+---------+

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 c.id havingcount(a.account\_type)>1;

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

9. Retrieve all transactions for a customer with a given customer\_id.

**select \* from customer c join account a on c.id=a.customer\_id join transaction t on a.id=t.account\_id where c.id=1;**

/\*+----+------------+-----------+------------+----+--------------+---------+-------------+----+------------------+------------------+--------+------------+

| id | first\_name | last\_name | dob | id | account\_type | balance | customer\_id | id | transaction\_type | transaction\_date | amount | account\_id |

+----+------------+-----------+------------+----+--------------+---------+-------------+----+------------------+------------------+--------+------------+

| 1 | harry | potter | 2002-03-21 | 1 | savings | 50000 | 1 | 1 | deposit | 2024-02-01 | 10000 | 1 |

| 1 | harry | potter | 2002-03-21 | 1 | savings | 50000 | 1 | 2 | withdrawal | 2024-02-02 | 5000 | 1 |

| 1 | harry | potter | 2002-03-21 | 4 | current | 150000 | 1 | 5 | transfer | 2024-02-01 | 20000 | 4 |

+----+------------+-----------+------------+----+--------------+---------+-------------+----+------------------+------------------+--------+------------+\*/

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

clause.

**select account\_type, sum(balance) as total\_balance from account group by account\_type;**

+--------------+---------------+

| account\_type | total\_balance |

+--------------+---------------+

| current | 270000 |

| savings | 80000 |

| zero\_balance | 100000 |