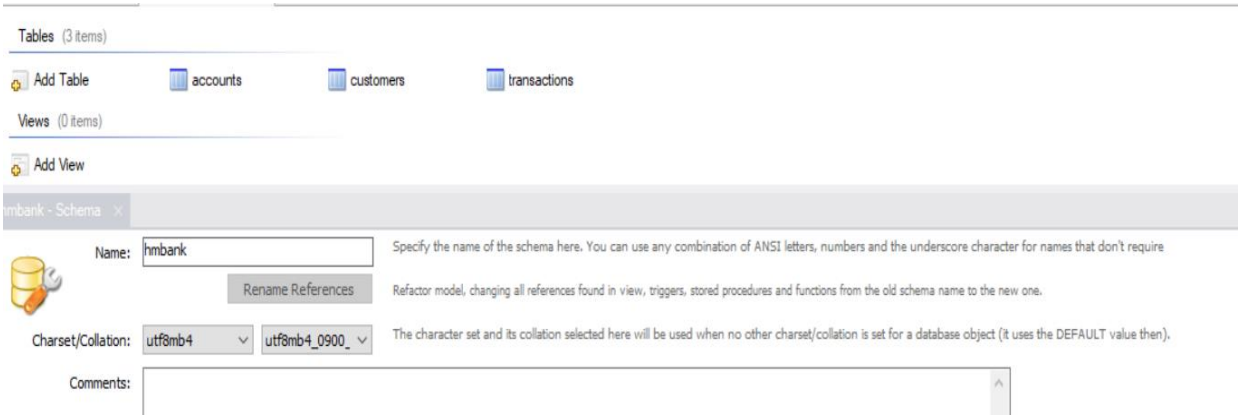
Task 1:

Database Design

1. Create the database named "HMBank"

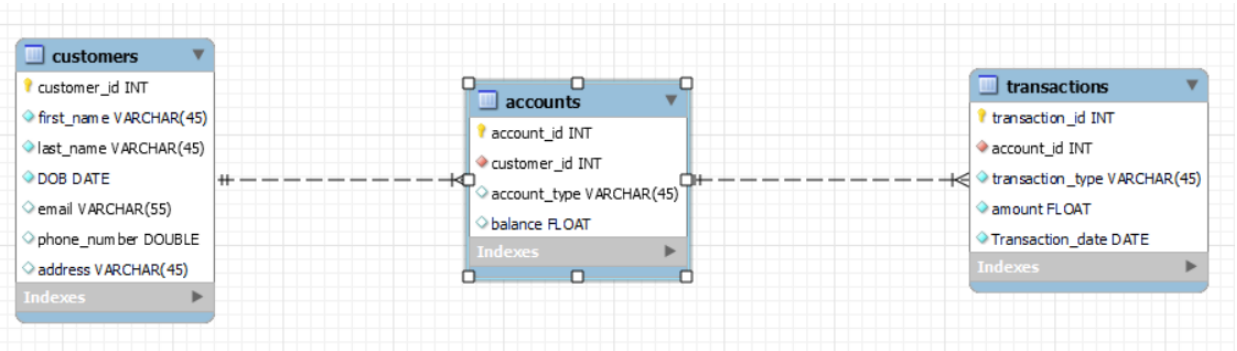
A. create database hmbank;

2. Define the schema for the Customers, Accounts, and Transactions tables based on the provided schema.



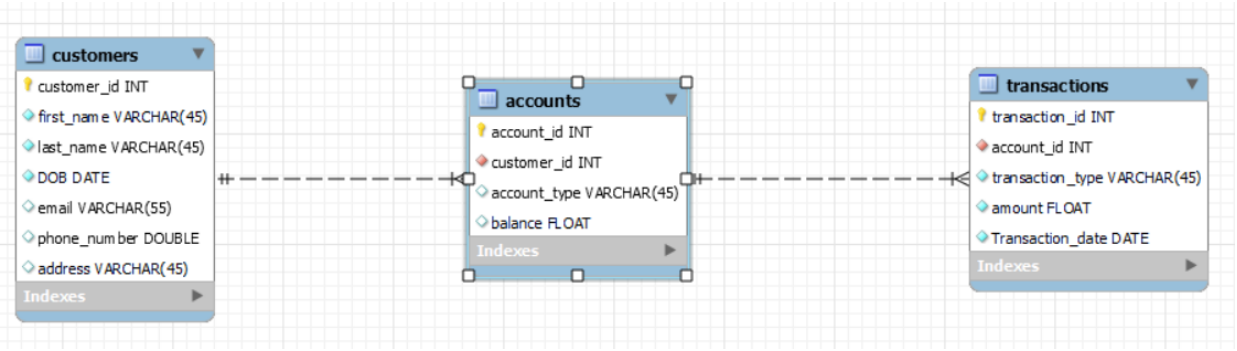
3. Create an ERD (Entity Relationship Diagram) for the database.

A.



4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.

A.



5. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.

- **Customers**

CREATE TABLE Customers (customer_id int(10) primary key,first_name varchar(10), last_name varchar(10), DOB Date, email varchar(10),phone number int(20), address varchar(10));

- **FOR ACCOUNTS:**

Create table Accounts (account_id int(10) primary key, customer_id varchar(10) foreign key, account_type varchar(10), balance float(10));

- **FOR TRANSACTIONS:**

Create table Transactions (transaction_id int(10) Primary key, account_id int(10) , transactions_type varchar(10), amount varchar(10) , transaction_date date);

Tasks 2:

Select, Where, Between, AND, LIKE:

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

- **Customers**
- **Accounts**
- **Transactions**

A. 1. Inserting into Customers Table:

INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’, `phone_number’, `address’)
VALUES ('1001', 'sai', 'pavan', `2001-02-20’, `sai@gmail.com’, `9182927904’, `kdkr’);

INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’, `phone_number’, `address’)
VALUES ('1002', 'bitra', 'venkatesh', `2002-08-23’, `bitra@gmail.com’, `9440523165’, `ogl’);

INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’, `phone_number’, `address’)
VALUES ('1003', 'naga', 'jayanth', `2000-07-23’, `naga@gmail.com’, `7456984521’, `kvl’);

INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’, `phone_number’, `address’)
VALUES ('1004', 'hari', 'krishna', `2002-05-14’, `hari@gmail.com’, `7845213698’, `hyd’);

```
INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’,
`phone_number’, `address’)
VALUES ('1005', 'sai', 'ram', `2003-05-15’, `ram@gmail.com’, `8956231247’, `kdkr’);

INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’,
`phone_number’, `address’)
VALUES ('1006', 'sai', 'meghana', `2003-09-16’, `sm@gmail.com’, `9856412378’, `nlr’);

INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’,
`phone_number’, `address’)
VALUES ('1007', 'raghu', 'nadh', `2003-03-07’, `raghu@gmail.com’, `8796541236’, `ogl’);

INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’,
`phone_number’, `address’)
 VALUES ('1008', 'meera', 'vali', `1999-02-18’, `mvali@gmail.com’, `9988774561’, `mas’);

INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’,
`phone_number’, `address’)
VALUES ('1009', 'mule', 'priya', `2004-06-28’, `priya@gmail.com’, `8523697451’, `tpt’);

INSERT INTO `hmbank`.`customers` (`customer_id`, `first_name’, `last_name’, `DOB’, `email’,
`phone_number’, `address’)
VALUES ('1010', 'sandhya', 'rani', `1999-12-15’, `rani@gmail.com’, `9456123879’, `kvl’);
```

- **Inserting into Accounts table:**

```
INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
VALUES ('123001', '1001', 'savings', '18500');

INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
 VALUES ('123002', '1002', 'zero_account', '0');

INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
VALUES ('123003', '1003', 'savings', '17850');

INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
VALUES ('123004', '1004', 'current', '5400');

INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
VALUES ('123005', '1005', 'savings', '15300');

INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
VALUES ('123006', '1006', 'current', '8650');

INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
VALUES ('123007', '1007', 'savings', '12620');

INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
VALUES ('123008', '1008', 'current', '6480');

INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
VALUES ('123009', '1009', 'zero_account', '800');
```

```
INSERT INTO `hmbank`.`accounts` (`account_id`, `customer_id`, `account_type`, `balance`)
VALUES ('123010', '1010', 'savings', '20200');
```

- **Insert into Transaction table:**

```
INSERT INTO `hmbank`.`transactions` (`transaction_id`, `account_id`, `transaction_type`, `amount`,
`Transactions_date`)
VALUES ('101007891', '123001', 'withdrawl', '12500', '2023-09-14');

INSERT INTO `hmbank`.`transactions` (`transaction_id`, `account_id`, `transaction_type`, `amount`,
`Transactions_date`)
 VALUES ('101007892', '123002', 'deposit', '1500', '2024-01-02');

INSERT INTO `hmbank`.`transactions` (`transaction_id`, `account_id`, `transaction_type`, `amount`,
`Transactions_date`)
VALUES ('101007893', '123007', 'withdrawl', '3600', '2023-02-21');

INSERT INTO `hmbank`.`transactions` (`transaction_id`, `account_id`, `transaction_type`, `amount`,
`Transactions_date`)
 VALUES ('101007894', '123008', 'deposit', '4500', '2023-11-09');

INSERT INTO `hmbank`.`transactions` (`transaction_id`, `account_id`, `transaction_type`, `amount`,
`Transactions_date`)
VALUES ('101007895', '123010', 'transfer', '10200', '2024-01-06');

INSERT INTO `hmbank`.`transactions` (`transaction_id`, `account_id`, `transaction_type`, `amount`,
`Transactions_date`)
VALUES ('1010078', '1010', 'transfer', '12200', '2023-12-09')
```

2. Write SQL queries for the following tasks:

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

A.

```
SELECT concat(c.first_name, c.last_name) as customer_name, a.account_type, c.email
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	account_type	email
saipavan	savings	sai@gmail.com
bitravenkatesh	zero_account	bitra@gmail.com
nagajayanth	savings	naga@gmail.com
harikrishna	current	hari@gmail.com
sairam	savings	ram@gmail.com
sai meghana	current	sm@gmail.com
raghunadh	savings	raghu@gmail.com
meeravali	current	mvali@gmail.com
mulepriya	zero_account	priya@gmail.com
sandhya rani	savings	rani@gmail.com

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

A.

```
SELECT c.customer_id,concat(c.first_name, c.last_name) as
customer_name,t.transaction_id,t.transaction_type, t.amount, t.transaction_date
FROM Customers c
JOIN Accounts a ON c.customer_id = a.customer_id
JOIN Transactions t ON a.account_id = t.account_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	customer_name	transaction_id	transaction_type	amount	transaction_date
1001	saipavan	101007891	withdrawl	12500	2023-09-14
1002	bitravenkatesh	101007892	deposit	1500	2024-01-02
1007	raghunadh	101007893	withdrawl	3600	2023-02-21
1008	meeravali	101007894	deposit	4500	2023-11-09
1010	sandhya rani	101007895	transfer	10200	2024-01-06

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

A.

```
UPDATE Accounts
SET balance = balance + 550
WHERE account_id =123006;
```

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

A.

```
SELECT concat(first_name,last_name) as Full_name from customers;
```

Result Grid | Filter Rows:

Full_name
saipavan
bitravenkatesh
nagajayanth
harikrishna
sairam
sai meghana
raghunadh
meeravali
mulepriya
sandhya rani

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

A. DELETE FROM Accounts WHERE balance = 0 AND account_type = ‘Savings’;

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

A. SELECT * FROM Customers WHERE address = 'kdkr';

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_id	first_name	last_name	DOB	email	phone_number	address
1001	sai	pavan	2001-02-20	sai@gmail.com	9182927904	kdkr
1005	sai	ram	2003-05-15	ram@gmail.com	8956231247	kdkr
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

A. SELECT account_id, balance FROM Accounts WHERE account_id = 123003;

Result Grid | Filter Row

account_id	balance
123003	17850
NULL	NULL

9.Write a SQL query to List all current accounts with a balance greater than 10,000.

A. SELECT * FROM Accounts WHERE account_type = 'current' AND balance > 10000;

Result Grid | Filter Rows: | Edit:

account_id	customer_id	account_type	balance
123006	1006	current	12550
NULL	NULL	NULL	NULL

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

A. SELECT * FROM Transactions WHERE account_id = 123010;

Result Grid | Filter Rows: | Edit: | Export/

transaction_id	account_id	transaction_type	amount	Transaction_date
101007895	123010	transfer	10200	2024-01-06
NULL	NULL	NULL	NULL	NULL

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

A. SELECT account_id, account_type, balance, 0.05 AS interest_rate, balance * 0.05 AS interest_accrued FROM Accounts WHERE account_type = 'Savings';

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

account_id	account_type	balance	interest_rate	interest_accrued
123001	savings	23750.1	0.05	1187.50703125
123003	savings	17850	0.05	892.5
123005	savings	15300	0.05	765
123007	savings	12620	0.05	631
123010	savings	20200	0.05	1010

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

A. SELECT * FROM Accounts WHERE balance <1000;

Result Grid | Filter Rows: | Edit:

account_id	customer_id	account_type	balance
123002	1002	zero_account	0
123009	1009	zero_account	800
NULL	NULL	NULL	NULL

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

A. SELECT * FROM Customers WHERE address <> 'kdkr';

Result Grid | Filter Rows: | Edit: | Export/Import: | Wr

customer_id	first_name	last_name	DOB	email	phone_number	address
1002	bitra	venkatesh	2002-08-23	bitra@gmail.com	9440523165	ogl
1003	naga	jayanth	2000-07-23	naga@gmail.com	7456984521	kvl
1004	hari	krishna	2002-05-14	hari@gmail.com	7845213698	hyd
1006	sai	meghana	2003-09-16	sm@gmail.com	9856412378	nlr
1007	raghu	nadh	2003-03-07	raghu@gmail.com	8796541236	ogl
1008	meera	vali	1999-02-18	mvali@gmail.com	9988774561	mas
1009	mule	priya	2004-06-28	priya@gmail.com	8523697451	tpt
1010	sandhya	rani	1999-12-15	rani@gmail.com	9456123879	kvl
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Tasks 3:

Aggregate functions, Having, Order By, GroupBy and Joins:

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

A. SELECT AVG(balance) AS avg_balance FROM Accounts;

Result Grid | Filter Rows:

avg_balance
11495.0140625

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

A. SELECT * FROM Accounts ORDER BY balance DESC LIMIT 10;

Result Grid | Filter Rows: | Edit:

account_id	customer_id	account_type	balance
123001	1001	savings	23750.1
123010	1010	savings	20200
123003	1003	savings	17850
123005	1005	savings	15300
123007	1007	savings	12620
123006	1006	current	12550
123008	1008	current	6480
123004	1004	current	5400
123009	1009	zero_account	800
123002	1002	zero_account	0
NULL	NULL	NULL	NULL

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

A. SELECT SUM(amount) AS total_deposits FROM Transactions WHERE transaction_type

= 'deposit' AND transaction_date = '2023-11-09';

Result Grid

total_deposits
4500

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

A. SELECT customer_id,concat(first_name,last_name) as customer_name,DOB FROM Customers ORDER BY DOB ASC LIMIT 10;

Result Grid | Filter Rows:

customer_id	customer_name	DOB
1008	meeravali	1999-02-18
1010	sandhya rani	1999-12-15
1003	nagajayanth	2000-07-23
1001	saipavan	2001-02-20
1004	harikrishna	2002-05-14
1002	bitravenkatesh	2002-08-23
1007	raghunadh	2003-03-07
1005	sairam	2003-05-15
1006	sai meghana	2003-09-16
1009	mulepriya	2004-06-28

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

A.SELECTt.transaction_id,t.account_id,t.transaction_type,t.amount,t.transaction_date,a.account_type FROM Transactions t JOIN Accounts a ON t.account_id = a.account_id;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

transaction_id	account_id	transaction_type	amount	transaction_date	account_type
101007891	123001	withdrawl	12500	2023-09-14	savings
101007892	123002	deposit	1500	2024-01-02	zero_account
101007893	123007	withdrawl	3600	2023-02-21	savings
101007894	123008	deposit	4500	2023-11-09	current
101007895	123010	transfer	10200	2024-01-06	savings

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

A. SELECT c.customer_id,c.first_name,c.last_name,c.DOB,c.email,c.phone_number,c.address,a.account_id,a.account_type,a.balance FROM Customers c JOIN Accounts a ON c.customer_id = a.customer_id;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	first_name	last_name	DOB	email	phone_number	address	account_id	account_type	balance
1001	sai	pavan	2001-02-20	sai@gmail.com	9182927904	kdkr	123001	savings	23750.1
1002	bitra	venkatesh	2002-08-23	bitra@gmail.com	9440523165	ogl	123002	zero_account	0
1003	naga	jayanth	2000-07-23	naga@gmail.com	7456984521	kvl	123003	savings	17850
1004	hari	krishna	2002-05-14	hari@gmail.com	7845213698	hyd	123004	current	5400
1005	sai	ram	2003-05-15	ram@gmail.com	8956231247	kdkr	123005	savings	15300
1006	sai	meghana	2003-09-16	sm@gmail.com	9856412378	nlr	123006	current	12550
1007	raghu	nadh	2003-03-07	raghu@gmail.com	8796541236	ogl	123007	savings	12620
1008	meera	vali	1999-02-18	mvali@gmail.com	9988774561	mas	123008	current	6480
1009	mule	priya	2004-06-28	priya@gmail.com	8523697451	tpt	123009	zero_account	800
1010	sandhya	rani	1999-12-15	rani@gmail.com	9456123879	kvl	123010	savings	20200

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

A.SELECT c.customer_id,c.first_name,c.last_name,c.DOB,c.email,c.phone_number,c.address,t.transaction_id,t.transaction_type,t.amount,t.transaction_date FROM Customers c JOIN Accounts a ON c.customer_id = a.customer_id JOIN Transactions t ON a.account_id = t.account_id WHERE a.account_id = 123001;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	first_name	last_name	DOB	email	phone_number	address	transaction_id	transaction_type	amount	transaction_date
1001	sai	pavan	2001-02-20	sai@gmail.com	9182927904	kdkr	101007891	withdrawl	12500	2023-09-14

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

A. SELECT customer_id FROM Accounts GROUP BY customer_id

HAVING COUNT(DISTINCT account_id) > 1;

There are no multiple accounts for any customer,so the return value is Null.

Result Grid | Filter Rov

customer_id

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

A. SELECT account_id, SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE -amount END) AS transaction_difference FROM Transactions GROUP BY account_id;

Result Grid | Filter Rows:

account_id	transaction_difference
123001	-12500
123002	1500
123007	-3600
123008	4500
123010	-10200

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

A. SELECT account_id, AVG(amount) AS average_daily_balance FROM transactions WHERE transaction_date BETWEEN '2023-01-01' AND '2024-01-07' GROUP BY account_id;

Result Grid | Filter Rows:

account_id	average_daily_balance
123001	12500
123002	1500
123007	3600
123008	4500
123010	10200

11.Calculate the total balance for each account type.

A. SELECT account_type, SUM(balance) AS total_balance FROM Accounts

GROUP BY account_type;

Result Grid | Filter Rows:

account_type	total_balance
savings	89720.140625
zero_account	800
current	24430

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

A. SELECT account_id, COUNT(transaction_id) AS transaction_count FROM Transactions GROUP BY account_id ORDER BY transaction_count DESC;

Result Grid | Filter Rows:

account_id	transaction_count
123001	1
123002	1
123007	1
123008	1
123010	1

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

A. SELECT c.customer_id, c.first_name, c.last_name, a.account_type, SUM(a.balance) AS aggregate_balance FROM Customers c JOIN Accounts a ON c.customer_id = a.customer_id GROUP BY c.customer_id, c.first_name, c.last_name, a.account_type ORDER BY aggregate_balance DESC;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	first_name	last_name	account_type	aggregate_balance
1001	sai	pavan	savings	23750.140625
1010	sandhya	rani	savings	20200
1003	naga	jayanth	savings	17850
1005	sai	ram	savings	15300
1007	raghu	nadh	savings	12620
1006	sai	meghana	current	12550
1008	meera	vali	current	6480
1004	hari	krishna	current	5400
1009	mule	priya	zero_account	800
1002	bitra	venkatesh	zero_account	0

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

A.

```
SELECT transaction_id, account_id, transaction_type, amount, transaction_date FROM
Transactions WHERE (amount, transaction_date, account_id) IN (
  SELECT amount, transaction_date, account_id FROM Transactions
  GROUP BY amount, transaction_date, account_id HAVING COUNT(*) > 1
  )
ORDER BY amount, transaction_date, account_id;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

transaction_id	account_id	transaction_type	amount	transaction_date
NULL	NULL	NULL	NULL	NULL

There are no duplicate transactions, therefore no data is returned.

Tasks 4:

Subquery and its type:

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

A. SELECT * FROM accounts WHERE balance = (SELECT MAX(balance) FROM accounts);

Result Grid | Filter Rows: | Edit:

account_id	customer_id	account_type	balance
123001	1001	savings	23750.1
NULL	NULL	NULL	NULL

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

A. SELECT C.customer_id, C.first_name, C.last_name, AVG(A.balance) AS average_balance FROM Customers C JOIN Accounts A ON C.customer_id = A.customer_id GROUP BY C.customer_id, C.first_name, C.last_name HAVING COUNT(A.account_id) > 1;

Result Grid | Filter Rows: | Export:

customer_id	first_name	last_name	average_balance

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

A. SELECT a.account_id, a.account_type, a.balance, t.transaction_id, t.transaction_type, t.amount, t.transaction_date FROM Accounts a JOIN Transactions t ON a.account_id = t.account_id WHERE t.amount > (SELECT AVG(amount) FROM Transactions);

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

account_id	account_type	balance	transaction_id	transaction_type	amount	transaction_date
123001	savings	23750.1	101007891	withdrawl	12500	2023-09-14
123010	savings	20200	101007895	transfer	10200	2024-01-06

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

A. SELECT c.customer_id, c.first_name, c.last_name FROM Customers c LEFT JOIN Accounts a ON c.customer_id = a.customer_id LEFT JOIN Transactions t ON a.account_id = t.account_id WHERE t.transaction_id IS NULL;

Result Grid | Filter Rows:

customer_id	first_name	last_name
1003	naga	jayanth
1004	hari	krishna
1005	sai	ram
1006	sai	meghana
1009	mule	priya

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

A. SELECT COALESCE(SUM(a.balance), 0) AS total_balance FROM Accounts a LEFT JOIN Transactions t ON a.account_id = t.account_id WHERE t.transaction_id IS NULL;

Result Grid

total_balance
51900

6. Retrieve transactions for accounts with the lowest balance.

A. SELECT t.transaction_id, t.account_id, t.transaction_type, t.amount, t.transaction_date FROM Transactions t JOIN Accounts a ON t.account_id = a.account_id WHERE a.balance = (SELECT MIN(balance) FROM Accounts);

Result Grid | Filter Rows: | Export: | Wrap Cell Conten

transaction_id	account_id	transaction_type	amount	transaction_date
101007892	123002	deposit	1500	2024-01-02

7.Identify customers who have accounts of multiple types.

A. SELECT C.customer_id, C.first_name, C.last_name FROM Customers C

JOIN Accounts A ON C.customer_id = A.customer_id GROUP BY

C.customer_id HAVINGCOUNT(DISTINCT A.account_type) > 1;

Result Grid | Filter Rows: | E

customer_id	first_name	last_name

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

A. SELECT account_type, COUNT(*) AS account_count,(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM Accounts)) AS percentage FROM Accounts GROUP BY account_type;

Result Grid | Filter Rows:

account_type	account_count	percentage
savings	5	50.0000
zero_account	2	20.0000
current	3	30.0000

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

A. SELECT t.transaction_id, t.account_id, t.transaction_type, t.amount, t.transaction_date FROM Transactions t JOIN Accounts a ON t.account_id = a.account_id JOIN Customers c ON a.customer_id = c.customer_id WHERE c.customer_id = 1001;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

transaction_id	account_id	transaction_type	amount	transaction_date
101007891	123001	withdrawl	12500	2023-09-14

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

A. SELECT DISTINCT account_type, (SELECT SUM(balance) FROM Accounts a WHERE a.account_type = ac.account_type) AS total_balance FROM Accounts ac;

Result Grid | Filter Rows:

account_type	total_balance
savings	89720.140625
zero_account	800
current	24430