Tasks 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.

A. 1. Creating Customers table:

```
create table Customers (
customer_id INT PRIMARY KEY,
customer_name VARCHAR(50) NOT NULL,
address VARCHAR(100),
contact_number VARCHAR(15)
);
```

2. Creating Accounts table:

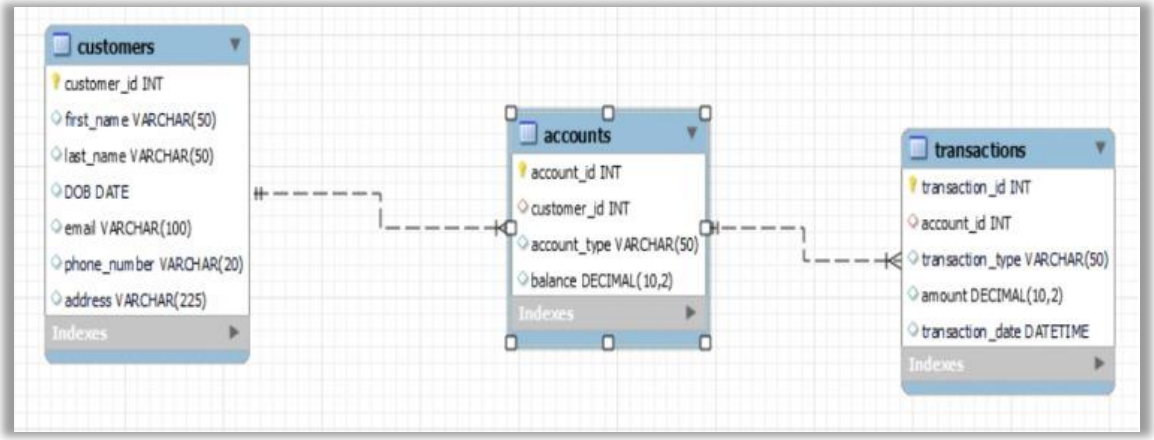
```
create table Accounts (
account_id INT PRIMARY KEY,
customer_id INT,
account_type VARCHAR(20) NOT NULL,
balance DECIMAL(10, 2) DEFAULT 0,
FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);
```

3. Creating Transactions table:

```
create table Transactions (
transaction_id INT PRIMARY KEY,
account_id INT,
transaction_type VARCHAR(20) NOT NULL,
amount DECIMAL(10, 2) NOT NULL,
transaction_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
);
```

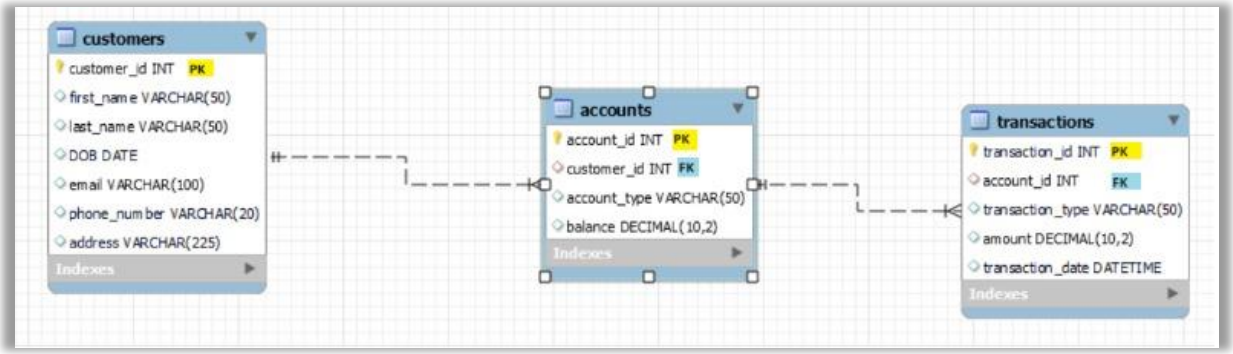
3. Create an ERD (Entity Relationship Diagram) for the database.

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

A.

1. Customers table:

```
create table Customers (
customer_id INT PRIMARY KEY,
customer_name VARCHAR(50) NOT NULL,
address VARCHAR(100),
contact_number VARCHAR(15)
```

```
);
```

2. Accounts table:

```
create table Accounts (
account_id INT PRIMARY KEY,
customer_id INT,
account_type VARCHAR(20) NOT NULL,
balance DECIMAL(10, 2) DEFAULT 0,
FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);
```

3. Transactions table:

```
create table Transactions (
transaction_id INT PRIMARY KEY,
account_id INT,
transaction_type VARCHAR(20) NOT NULL,
amount DECIMAL(10, 2) NOT NULL,
transaction_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
FOREIGN KEY (account_id) REFERENCES Accounts(account_id)
);
```

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 Customers (customer_id, first_name,last_name,DOB, email,
phone_number,address) VALUES
(1, 'Goutham','Kakarla','2001-10-31','gautham@gmail.com','99999999','Kovvur'),
(2, 'Preetham','Kakarla','2003-11-23','preetham@gmail.com','999999998','Kovvur'),
(3, 'Indhu','Jalem','1998-10-18','indhu@gmail.com','9999999997','Nidadavole'),
(4, 'Chandu','Jalem','1997-07-16','chandhu@gmail.com','9999999996','Nidadavole'),
(5, 'Chanti','Barthi', '2000-09-12','chanti@gmail.com','9999999995','Chennai'),
(6, 'Manju','Barthi','2002-09-04','manju@gmail.com','9999999994', 'Eluru'),
```

(7, 'Dinku','Barthi','2004-11-05','dinku@gmail.com','9999999993', 'Hyderabad'),

(8, 'Sonu','Bontha','2002-08-01','sonu@gmail.com','9999999992', 'Machilipatnam'),

(9, 'Minnu','Barthi' ,'2005-04-21','minnu@gmail.com','9999999991', 'Gopalapuram'),

(10, 'Binnu','Barthi' ,'2006-05-13','binnu@gmail.com','9999999990', 'Gopalapuram')

;

2. Inserting into Accounts Table:

A. insert into Accounts (account_id, customer_id, account_type, balance)VALUES

(101, 1, 'Savings', 5000.00),

(102, 2, 'Current', 1000.00),

(103, 3, 'Savings', 3000.00),

(104, 4, 'Current', 2000.00),

(105, 5, 'Savings', 7000.00),

(106, 6, 'zero_balance', 0.00),

(107, 7, 'Savings', 4500.00),

(108, 8, 'Current', 3000.00),

(109, 9, 'Savings', 6000.00),

(110, 10, 'zero_balance', 0.00),

3. Inserting into Transactions Table:

A. insert into Transactions (transaction_id, account_id, transaction_type, amount, transaction_date) VALUES

(1001, 101, 'Deposit', 1000.00, '2024-01-13 12:30:00'),

(1002, 101, 'Withdrawal', 500.00, '2024-01-14 10:45:00'),

(1003, 102, 'Deposit', 2000.00, '2024-01-15 15:20:00'),

(1004, 102, 'Withdrawal', 1000.00, '2024-01-16 09:10:00'),

(1005, 103, 'Deposit', 1500.00, '2024-01-17 14:45:00'),

(1006, 104, 'Withdrawal', 800.00, '2024-01-18 11:30:00'),

(1007, 105, 'Deposit', 3000.00, '2024-01-19 13:15:00'),

(1008, 106, 'Withdrawal', 1200.00, '2024-01-20 16:40:00'),

(1009, 107, 'Deposit', 2500.00, '2024-01-21 08:55:00'),

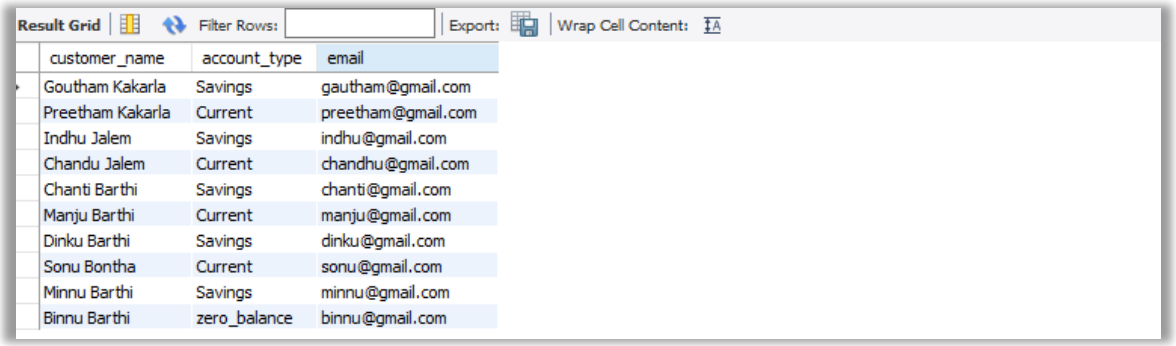
(1010, 108, 'Withdrawal', 900.00, '2024-01-22 10:25:00');

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
Goutham Kakarla	Savings	gautham@gmail.com
Preetham Kakarla	Current	preetham@gmail.com
Indhu Jalem	Savings	indhu@gmail.com
Chandu Jalem	Current	chandhu@gmail.com
Chanti Barthi	Savings	chanti@gmail.com
Manju Barthi	Current	manju@gmail.com
Dinku Barthi	Savings	dinku@gmail.com
Sonu Bontha	Current	sonu@gmail.com
Minnu Barthi	Savings	minnu@gmail.com
Binnu Barthi	zero_balance	binnu@gmail.com

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

A.

```
SELECT
    T.transaction_id,
    T.account_id,
    T.transaction_type,
    T.amount,
    T.transaction_date,
    C.customer_id,
    CONCAT(C.first_name, ' ', C.last_name) AS customer_name,
    C.email
FROM
    Transactions T
JOIN
    Accounts A ON T.account_id = A.account_id
JOIN
    Customers C ON A.customer_id = C.customer_id;
```

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

transaction_id	account_id	transaction_type	amount	transaction_date	customer_id	customer_name	email
1001	101	Deposit	1000.00	2024-01-13 12:30:00	1	Goutham Kakarla	gautham@gmail.com
1002	102	Withdrawal	500.00	2024-01-14 10:45:00	2	Preetham Kakarla	preetham@gmail.com
1003	103	Deposit	2000.00	2024-01-15 15:20:00	3	Indhu Jalem	indhu@gmail.com
1004	104	Withdrawal	1000.00	2024-01-16 09:10:00	4	Chandu Jalem	chandhu@gmail.com
1005	105	Deposit	1500.00	2024-01-17 14:45:00	5	Chanti Barthi	chanti@gmail.com
1006	106	Withdrawal	800.00	2024-01-18 11:30:00	6	Manju Barthi	manju@gmail.com
1007	107	Deposit	3000.00	2024-01-19 13:15:00	7	Dinku Barthi	dinku@gmail.com
1008	108	Withdrawal	1200.00	2024-01-20 16:40:00	8	Sonu Bontha	sonu@gmail.com
1009	109	Withdrawal	900.00	2024-01-22 10:25:00	9	Minnu Barthi	minnu@gmail.com
1010	110	Deposit	2500.00	2024-01-21 08:55:00	10	Binnu Barthi	binnu@gmail.com

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

A. UPDATE Accounts

SET balance = balance + 10000.00

WHERE account_id = 101;

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: | Export: | Wrap Cell Content:

full_name
Goutham Kakarla
Preetham Kakarla
Indhu Jalem
Chandu Jalem
Chanti Barthi
Manju Barthi
Dinku Barthi
Sonu Bontha
Minnu Barthi
Binnu Barthi

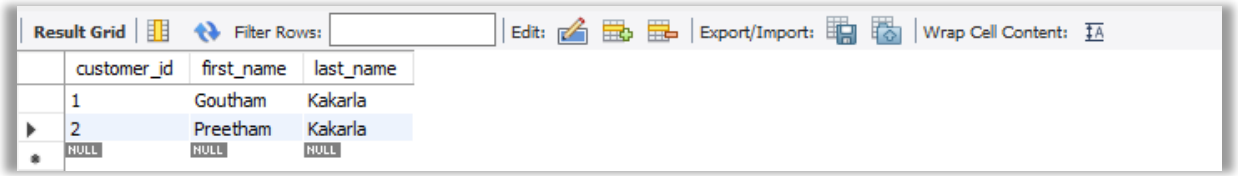
5. Write a SQL query to remove accounts with a balance of zero where the account type is savings.

A. DELETE FROM Accounts

WHERE balance = 0 AND account_type = 'Savings';

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

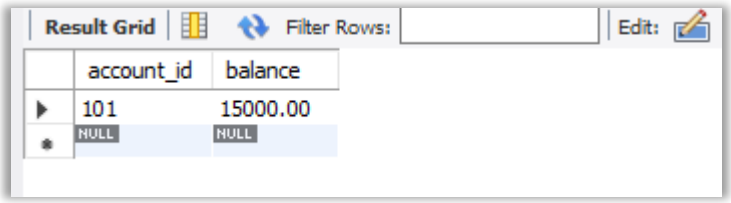
A. SELECT customer_id, first_name,last_name FROM Customers WHERE address = 'Kovvur';



customer_id	first_name	last_name
1	Goutham	Kakarla
2	Preetham	Kakarla
NULL	NULL	NULL

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

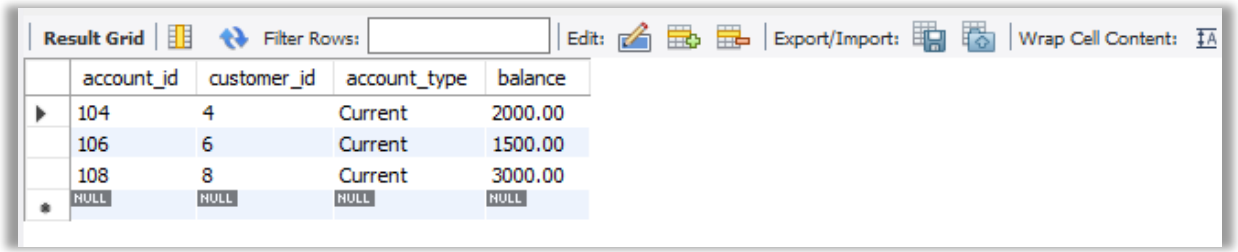
A. SELECT account_id, balance FROM Accounts WHERE account_id = 101;



account_id	balance
101	15000.00
NULL	NULL

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

A. SELECT account_id, customer_id, account_type, balance FROM Accounts WHERE account_type = 'Current' AND balance > 1000.00;

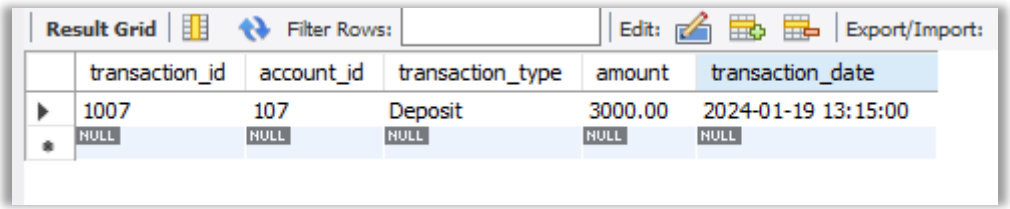


account_id	customer_id	account_type	balance
104	4	Current	2000.00
106	6	Current	1500.00
108	8	Current	3000.00
NULL	NULL	NULL	NULL

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

A. SELECT transaction_id, account_id, transaction_type, amount, transaction_date

FROM transactions WHERE account_id = 107;



transaction_id	account_id	transaction_type	amount	transaction_date
1007	107	Deposit	3000.00	2024-01-19 13:15:00
NULL	NULL	NULL	NULL	NULL

10. 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
101	Savings	15000.00	0.05	750.0000
103	Savings	3000.00	0.05	150.0000
105	Savings	7000.00	0.05	350.0000
107	Savings	4500.00	0.05	225.0000
109	Savings	6000.00	0.05	300.0000

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

A. SELECT account_id, account_type, balance FROM Accounts WHERE balance < 2000;

Result Grid | Filter Rows: | Edit:

account_id	account_type	balance
102	Current	1000.00
106	Current	1500.00
110	zero_balance	0.00
NULL	NULL	NULL

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

A. SELECT customer_id, first_name, last_name, address, FROM Customers WHERE city != 'Kovvur';

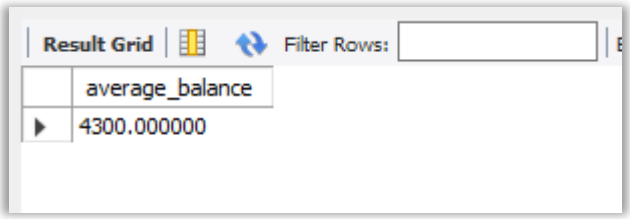
Result Grid | Filter Rows: | Edit:

customer_id	first_name	last_name	address
3	Indhu	Jalem	Nidadavole
4	Chandu	Jalem	Nidadavole
5	Chanti	Barthi	Chennai
6	Manju	Barthi	Eluru
7	Dinku	Barthi	Hyderabad
8	Sonu	Bontha	Machilipatnam
9	Minnu	Barthi	Gopalapuram
10	Binnu	Barthi	Gopalapuram
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 average_balance FROM Accounts;



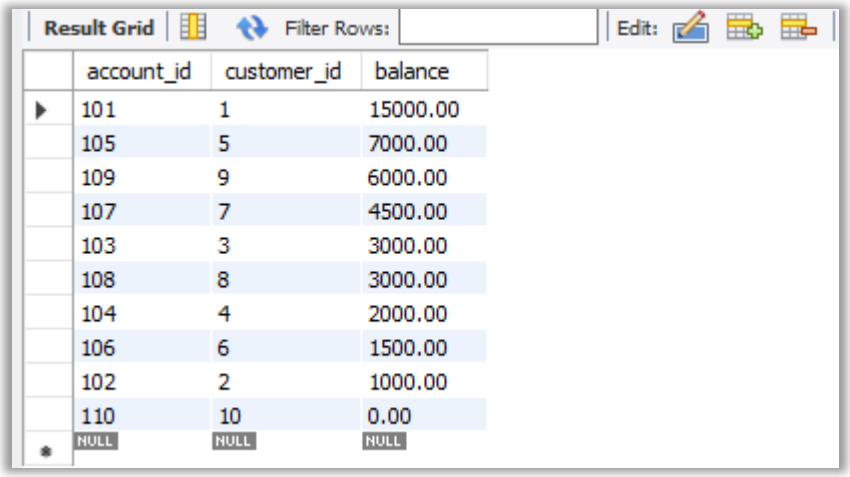
Result Grid | Filter Rows:

average_balance
4300.000000

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

A. SELECT account_id, customer_id,balance

FROM Accounts ORDER BY balance DESC LIMIT 10;



Result Grid | Filter Rows: | Edit:

account_id	customer_id	balance
101	1	15000.00
105	5	7000.00
109	9	6000.00
107	7	4500.00
103	3	3000.00
108	8	3000.00
104	4	2000.00
106	6	1500.00
102	2	1000.00
110	10	0.00
NULL	NULL	NULL

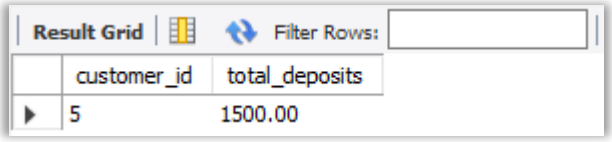
3. Write a SQL query to Calculate Total Deposits for All Customers in specific date.

A. SELECT A.customer_id, SUM(T.amount) AS total_deposits FROM Transactions T

JOIN

Accounts A ON T.account_id = A.account_id WHERE T.transaction_type = 'deposit'

AND DATE(T.transaction_date) = '2024-01-17' GROUP BY A.customer_id;



Result Grid | Filter Rows:

customer_id	total_deposits
5	1500.00

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

A. Oldest Customer:

```
SELECT customer_id, first_name, last_name,DOB FROM Customers
ORDER BY DOB ASC LIMIT 1;
```

Result Grid | Filter Rows: | Edit:

customer_id	first_name	last_name	DOB
4	Chandu	Jalem	1997-07-16
NULL	NULL	NULL	NULL

Newest Customer:

```
SELECT  customer_id,first_name, last_name, DOB FROM Customers
ORDER BY DOB DESC LIMIT 1;
```

Result Grid | Filter Rows: | Edit:

customer_id	first_name	last_name	DOB
10	Binnu	Barthi	2006-05-13
NULL	NULL	NULL	NULL

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

A.

```
SELECT T.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
1001	101	Deposit	1000.00	2024-01-13 12:30:00	Savings
1002	102	Withdrawal	500.00	2024-01-14 10:45:00	Current
1003	103	Deposit	2000.00	2024-01-15 15:20:00	Savings
1004	104	Withdrawal	1000.00	2024-01-16 09:10:00	Current
1005	105	Deposit	1500.00	2024-01-17 14:45:00	Savings
1006	106	Withdrawal	800.00	2024-01-18 11:30:00	Current
1007	107	Deposit	3000.00	2024-01-19 13:15:00	Savings
1008	108	Withdrawal	1200.00	2024-01-20 16:40:00	Current
1009	109	Withdrawal	900.00	2024-01-22 10:25:00	Savings
1010	110	Deposit	2500.00	2024-01-21 08:55:00	zero_balance

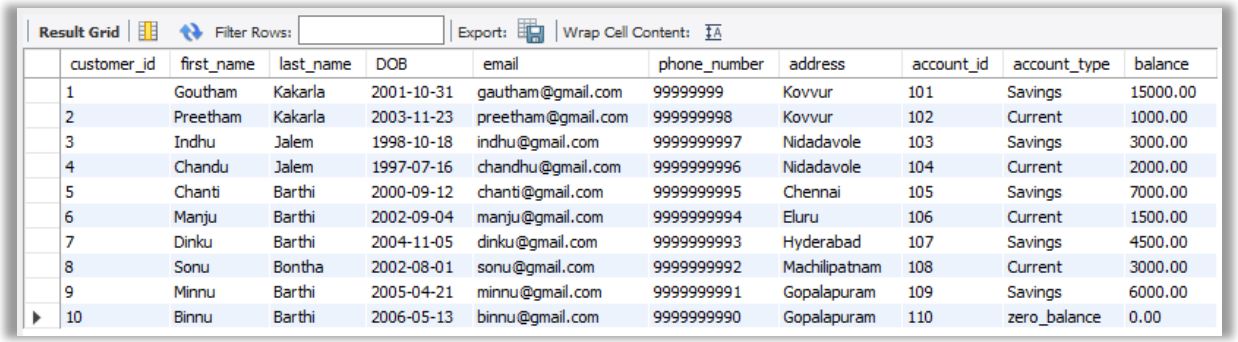
6. Write a SQL query to Get a list of customers along with their account details.

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;



customer_id	first_name	last_name	DOB	email	phone_number	address	account_id	account_type	balance
1	Goutham	Kakarla	2001-10-31	gautham@gmail.com	99999999	Kovvur	101	Savings	15000.00
2	Preetham	Kakarla	2003-11-23	preetham@gmail.com	999999998	Kovvur	102	Current	1000.00
3	Indhu	Jalem	1998-10-18	indhu@gmail.com	9999999997	Nidadavole	103	Savings	3000.00
4	Chandu	Jalem	1997-07-16	chandhu@gmail.com	9999999996	Nidadavole	104	Current	2000.00
5	Chanti	Barthi	2000-09-12	chanti@gmail.com	9999999995	Chennai	105	Savings	7000.00
6	Manju	Barthi	2002-09-04	manju@gmail.com	9999999994	Eluru	106	Current	1500.00
7	Dinku	Barthi	2004-11-05	dinku@gmail.com	9999999993	Hyderabad	107	Savings	4500.00
8	Sonu	Bontha	2002-08-01	sonu@gmail.com	9999999992	Machilipatnam	108	Current	3000.00
9	Minnu	Barthi	2005-04-21	minnu@gmail.com	9999999991	Gopalapuram	109	Savings	6000.00
10	Binnu	Barthi	2006-05-13	binnu@gmail.com	9999999990	Gopalapuram	110	zero_balance	0.00

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

A. SELECT T.transaction_id, T.account_id, T.transaction_type, T.amount, T.transaction_date

C.customer_id, C.first_name, C.last_name, C.DOB, C.email, C.phone_number, C.address

FROM Transactions T

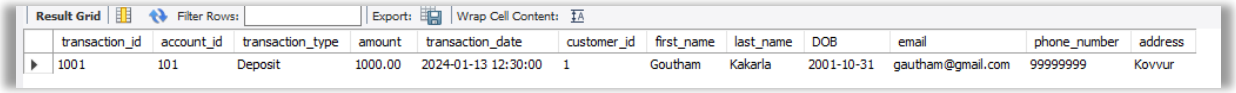
JOIN

Accounts A ON T.account_id = A.account_id

JOIN

Customers C ON A.customer_id = C.customer_id

WHERE T.account_id = 101;



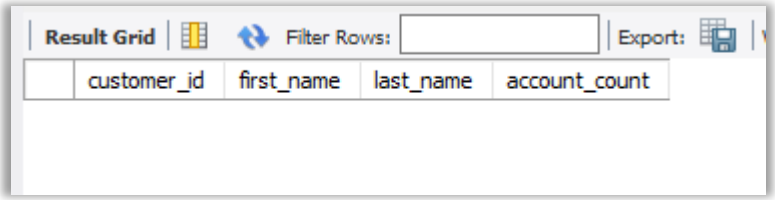
transaction_id	account_id	transaction_type	amount	transaction_date	customer_id	first_name	last_name	DOB	email	phone_number	address
1001	101	Deposit	1000.00	2024-01-13 12:30:00	1	Goutham	Kakarla	2001-10-31	gautham@gmail.com	99999999	Kovvur

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

A. SELECT C.customer_id, C.first_name, C.last_name, COUNT(A.account_id) AS account_count FROM

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

HAVING COUNT(A.account_id) > 1;



customer_id	first_name	last_name	account_count

Since there are no multiple accounts for any customer, no data is returned.

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

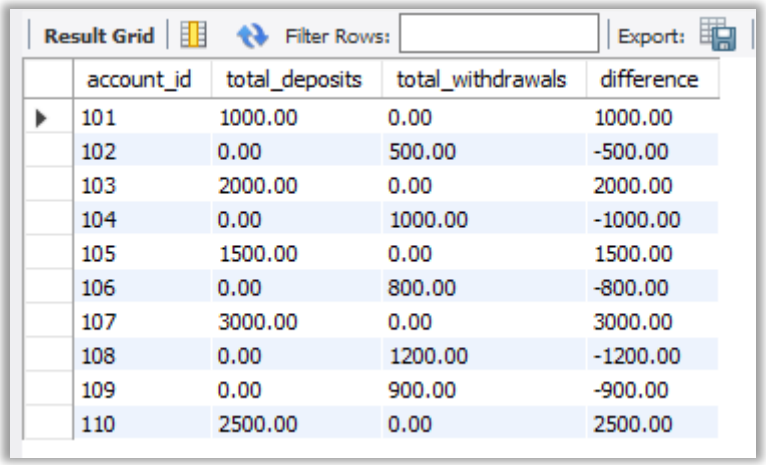
A. SELECT account_id,

SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE 0 END) AS total_deposits,

SUM(CASE WHEN transaction_type = 'withdrawal' THEN amount ELSE 0 END) AS total_withdrawals,

SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE -amount END) AS difference

FROM Transactions GROUP BY account_id;



Result Grid | Filter Rows: | Export:

account_id	total_deposits	total_withdrawals	difference
101	1000.00	0.00	1000.00
102	0.00	500.00	-500.00
103	2000.00	0.00	2000.00
104	0.00	1000.00	-1000.00
105	1500.00	0.00	1500.00
106	0.00	800.00	-800.00
107	3000.00	0.00	3000.00
108	0.00	1200.00	-1200.00
109	0.00	900.00	-900.00
110	2500.00	0.00	2500.00

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

A.

```
SELECT
account_id,
AVG(balance) AS average_daily_balance
FROM (
SELECT
A.account_id,
T.transaction_date,
SUM(CASE WHEN T.transaction_type = 'deposit' THEN amount
    WHEN T.transaction_type = 'withdrawal' THEN -amount
    ELSE 0 END) AS balance
FROM
Accounts A
```

```
    JOIN
        Transactions T ON A.account_id = T.account_id
    WHERE
        T.transaction_date BETWEEN '2024-01-17' AND '2024-01-22'
    GROUP BY
        A.account_id,
        T.transaction_date
) AS DailyBalances
GROUP BY
    account_id;
```

Result Grid | Filter Rows: | Export:

account_id	average_daily_balance
105	1500.000000
106	-800.000000
107	3000.000000
108	-1200.000000
110	2500.000000

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	35500.00
Current	7500.00
zero_balance	0.00

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
101	1
102	1
103	1
104	1
105	1
106	1
107	1
108	1
109	1
110	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 total_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, C.DOB, C.email, C.phone_number, C.address,
A.account_type ORDER BY total_balance DESC;
```

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

customer_id	first_name	last_name	account_type	total_balance
1	Goutham	Kakarla	Savings	15000.00
5	Chanti	Barthi	Savings	7000.00
9	Minnu	Barthi	Savings	6000.00
7	Dinku	Barthi	Savings	4500.00
3	Indhu	Jalem	Savings	3000.00
8	Sonu	Bontha	Current	3000.00
4	Chandu	Jalem	Current	2000.00
6	Manju	Barthi	Current	1500.00
2	Preetham	Kakarla	Current	1000.00
10	Binnu	Barthi	zero_balance	0.00

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

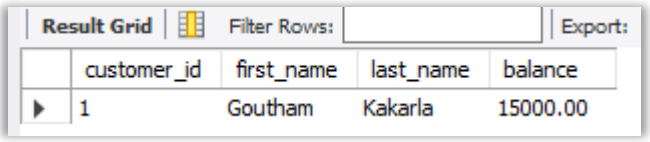
There are no duplicate transactions, therefore no data is returned.

Tasks 4: Subquery and its type:

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

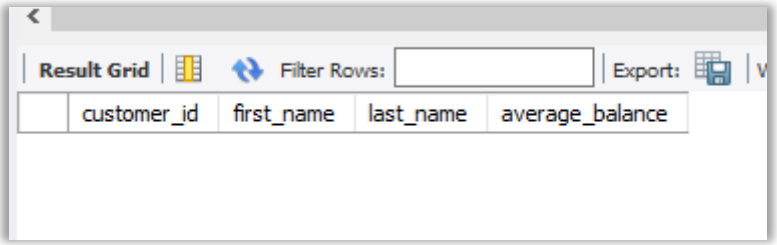
A.

```
WITH RankedCustomers AS (
SELECT C.customer_id, C.first_name, C.last_name, A.balance, RANK() OVER (ORDER BY
A.balance DESC) AS balance_rank FROM Customers C
JOIN Accounts A ON C.customer_id = A.customer_id )
SELECT customer_id, first_name, last_name, balance FROM RankedCustomers
WHERE  balance_rank = 1;
```



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;

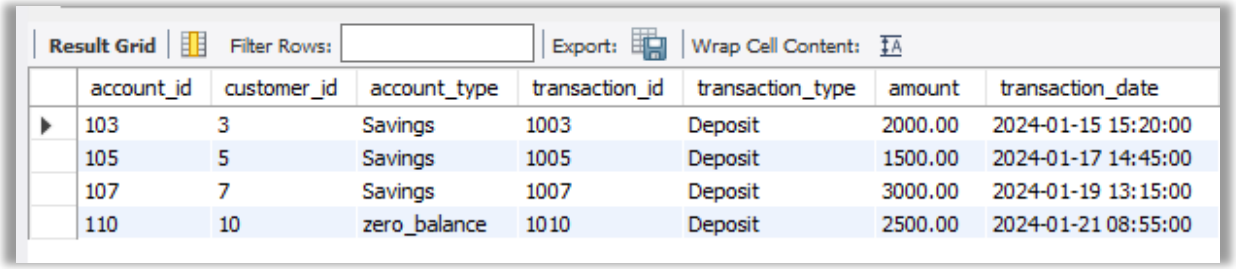


3. Retrieve accounts with transactions whose amounts exceed the average transaction amount.

A.

```
WITH TransactionAverages AS (
SELECT AVG(amount) AS average_transaction_amount FROM Transactions)
SELECT A.account_id, A.customer_id, A.account_type, T.transaction_id, T.transaction_type,
T.amount, T.transaction_date FROM Accounts A
JOIN
Transactions T ON A.account_id = T.account_id
CROSS JOIN
TransactionAverages
WHERE
```

T.amount > TransactionAverages.average_transaction_amount;

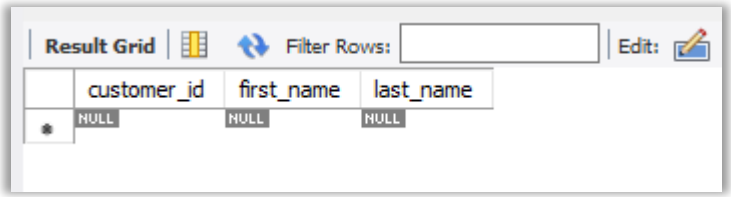


account_id	customer_id	account_type	transaction_id	transaction_type	amount	transaction_date
103	3	Savings	1003	Deposit	2000.00	2024-01-15 15:20:00
105	5	Savings	1005	Deposit	1500.00	2024-01-17 14:45:00
107	7	Savings	1007	Deposit	3000.00	2024-01-19 13:15:00
110	10	zero_balance	1010	Deposit	2500.00	2024-01-21 08:55:00

4. Identify customers who have no recorded transactions.

A.

```
SELECT C.customer_id, C.first_name, C.last_name FROM Customers C WHERE
C.customer_id NOT IN (
   SELECT DISTINCT customer_id
   FROM Transactions
);
```

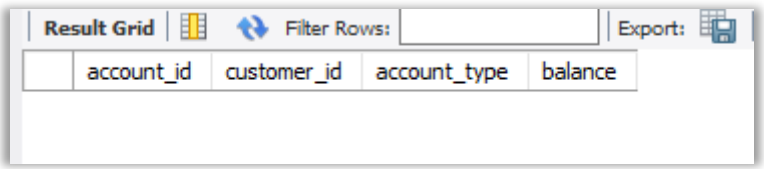


customer_id	first_name	last_name
NULL	NULL	NULL

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

A.

```
SELECT A.account_id, A.customer_id, A.account_type, A.balance FROM Accounts A
LEFT JOIN
Transactions T ON A.account_id = T.account_id WHERE T.account_id IS NULL;
```



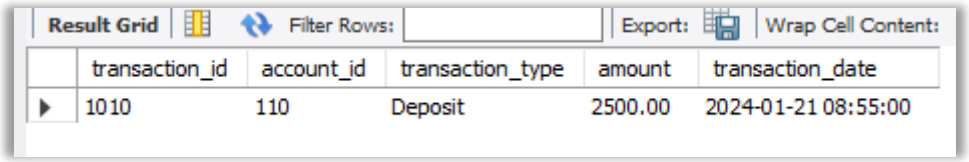
account_id	customer_id	account_type	balance

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 (
   SELECT account_id, RANK() OVER (ORDER BY balance) AS balance_rank FROM
   Accounts) AS RankedAccounts ON T.account_id = RankedAccounts.account_id
```

```
WHERE balance_rank = 1;
```



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

transaction_id	account_id	transaction_type	amount	transaction_date
1010	110	Deposit	2500.00	2024-01-21 08:55:00

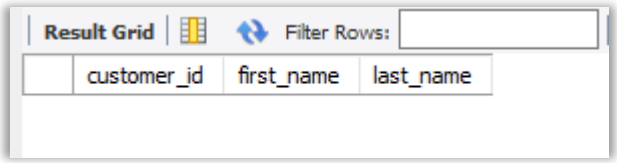
7. Identify customers who have accounts of multiple types.

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 HAVING

COUNT(DISTINCT A.account_type) > 1;

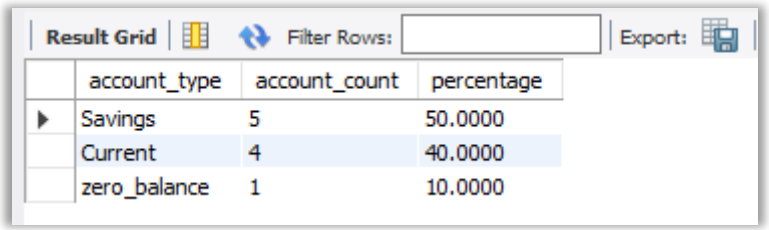


Result Grid | Filter Rows:

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(*) / (SELECT COUNT(*) FROM Accounts)) * 100 AS percentage FROM Accounts GROUP BY account_type;



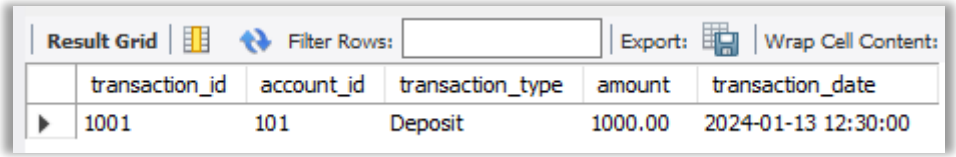
Result Grid | Filter Rows: | Export:

account_type	account_count	percentage
Savings	5	50.0000
Current	4	40.0000
zero_balance	1	10.0000

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 = 1;
```

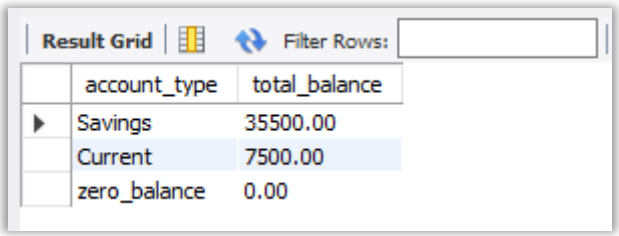


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

transaction_id	account_id	transaction_type	amount	transaction_date
1001	101	Deposit	1000.00	2024-01-13 12:30:00

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

A. SELECT account_type, (SELECT SUM(balance) FROM Accounts A WHERE A.account_type = Accounts.account_type) AS total_balance FROM Accounts GROUP BY account_type;



Result Grid | Filter Rows:

account_type	total_balance
Savings	35500.00
Current	7500.00
zero_balance	0.00