Task 3: Aggregate functions, Having, Order By, GroupBy and Joins

1) **SQL query to Find the average account balance for all customers.**

```
Use hmbank;
SELECT AVG(balance) AS average_balance FROM Accounts;
```

average_balance
2700.000000

2) **SQL query to Retrieve the top 10 highest account balances.**

```
SELECT * FROM Accounts ORDER BY balance DESC LIMIT 10;
```

account_id	customer_id	account_type	balance
104	4	current	8000.00
108	8	current	6000.00
102	2	current	3000.00
107	7	savings	2500.00
101	1	savings	2000.00
106	6	current	1200.00
110	10	current	900.00
103	3	savings	500.00
109	9	savings	200.00
NULL	NULL	NULL	NULL

3) **SQL query to Calculate Total Deposits for All Customers in specific date.**

```
SELECT SUM(amount) AS total_deposits
FROM Transactions
WHERE transaction_type = 'deposit' AND transaction_date = '2024-03-05';
```

total_deposits
300.00

4) **SQL query to Find the Oldest and Newest Customers**

```
SELECT * FROM Customers ORDER BY DOB ASC LIMIT 1;
```

customer_id	first_name	last_name	DOB	email	phone_number	address
1	John	Doe	1985-07-10	john.doe@example.com	9876543210	New York
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
SELECT * FROM Customers ORDER BY DOB DESC LIMIT 1;
```

customer_id	first_name	last_name	DOB	email	phone_number	address
7	Frank	Wilson	1997-02-17	frank.wilson@example.com	9898765432	San Diego
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

```
SELECT T.*, A.account_type
FROM Transactions T
JOIN Accounts A ON T.account_id = A.account_id;
```

transaction_id	account_id	transaction_type	amount	transaction_date	account_type
1	101	deposit	500.00	2024-01-15 00:00:00	savings
2	102	withdrawal	700.00	2024-02-10 00:00:00	current
3	103	deposit	300.00	2024-03-05 00:00:00	savings
4	104	withdrawal	1000.00	2024-04-20 00:00:00	current
6	106	withdrawal	400.00	2024-06-12 00:00:00	current
7	107	deposit	600.00	2024-07-18 00:00:00	savings
8	108	withdrawal	500.00	2024-08-22 00:00:00	current
9	109	deposit	350.00	2024-09-25 00:00:00	savings
10	110	withdrawal	200.00	2024-10-28 00:00:00	current

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

```
SELECT C.customer_id, C.first_name, C.last_name, 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	account_id	account_type	balance
1	John	Doe	101	savings	2000.00
2	Alice	Smith	102	current	3000.00
3	Bob	Johnson	103	savings	500.00
4	Charlie	Brown	104	current	8000.00
6	Eve	Miller	106	current	1200.00
7	Frank	Wilson	107	savings	2500.00
8	Grace	Moore	108	current	6000.00
9	Hannah	Taylor	109	savings	200.00
10	Isaac	Anderson	110	current	900.00

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

```
SELECT T.*, C.first_name, C.last_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
WHERE A.account_id = '103';
```

transaction_id	account_id	transaction_type	amount	transaction_date	first_name	last_name	email
3	103	deposit	300.00	2024-03-05 00:00:00	Bob	Johnson	bob.johnson@example.com

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

```
SELECT customer_id, COUNT(account_id) AS account_count
FROM Accounts
GROUP BY customer_id
HAVING COUNT(account_id) > 1;
```

customer_id	account_count
1	2
2	2

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

```
SELECT account_id,
       SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE 0 END) -
       SUM(CASE WHEN transaction_type = 'withdrawal' THEN amount ELSE 0 END) AS balance_difference
FROM Transactions
GROUP BY account_id;
```

account_id	balance_difference
101	500.00
102	-700.00
103	300.00
104	-1000.00
106	-400.00
107	600.00
108	-500.00
109	350.00
110	-200.00

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

```
SELECT account_id, AVG(balance) AS avg_daily_balance
FROM Accounts
WHERE account_id IN (
    SELECT DISTINCT account_id FROM Transactions
    WHERE transaction_date BETWEEN '2024-02-10' AND '2024-09-26'
)
GROUP BY account_id;
```

account_id	avg_daily_balance
102	3000.000000
103	500.000000
104	8000.000000
106	1200.000000
107	2500.000000
108	6000.000000
109	200.000000

11) Calculate the total balance for each account type.

```
SELECT account_type, SUM(balance) AS total_balance
FROM Accounts
GROUP BY account_type;
```

account_type	total_balance
savings	8700.00
current	21600.00

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

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

account_id	transaction_count
101	4
103	3
102	1
104	1
106	1
107	1
108	1
109	1
110	1

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

```
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, A.account_type
HAVING SUM(A.balance) > 1000;
```

customer_id	first_name	last_name	account_type	total_balance
1	John	Doe	savings	2000.00
2	Alice	Smith	current	3000.00
4	Charlie	Brown	current	8000.00
6	Eve	Miller	current	1200.00
7	Frank	Wilson	savings	2500.00
8	Grace	Moore	current	6000.00
1	John	Doe	current	2500.00
2	Alice	Smith	savings	3500.00

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

```
SELECT account_id, transaction_date, amount, COUNT(*) AS duplicate_count
FROM Transactions
GROUP BY account_id, transaction_date, amount
HAVING COUNT(*) > 1;
```

account_id	transaction_date	amount	duplicate_count
101	2024-01-15 00:00:00	500.00	2
102	2024-02-10 00:00:00	700.00	2