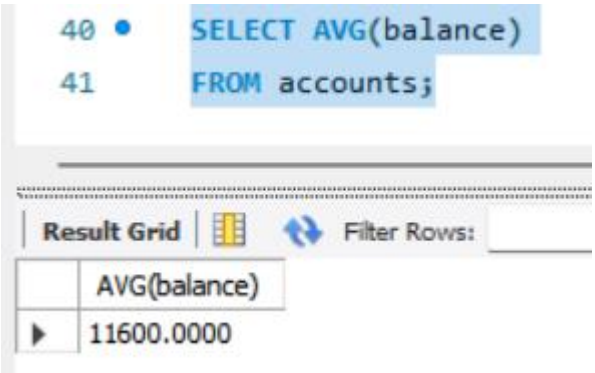
Task 3

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

Query:

```
SELECT AVG(balance)
FROM accounts;
```

Output:



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

Query:

```
SELECT * FROM accounts
ORDER BY balance
LIMIT 10;
```

Output:

```
SELECT * FROM accounts
ORDER BY balance
LIMIT 10;
```

Result Grid | Filter Rows: | Ed

account_id	customer_id	account_type	balance
10002	4	zero_balance	1000
10007	8	zero_balance	2000
10005	3	current	8000
10010	10	savings	9000
10001	1	savings	10000
10003	2	savings	12000
10006	6	savings	13000
10008	7	savings	16000
10004	5	current	20000
10009	9	current	25000

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

Query:

```
SELECT SUM(amount) AS Total Deposits
FROM transaction
WHERE transaction_type='deposit'
AND transaction_date='2024-04-08';
```

Output:

transaction_id	account_id	transaction_type	amount	transaction_date
3132024500	10001	deposit	2000	2024-04-01
3132024501	10002	withdrawl	500	2024-04-03
3132024502	10004	transfer	5000	2024-04-05
3132024503	10003	withdrawl	2000	2024-03-29
3132024504	10006	deposit	5000	2024-04-01
3132024505	10005	transfer	4000	2024-04-04
3132024506	10010	deposit	2000	2024-04-07
3132024507	10007	withdrawl	2000	2024-04-06
3132024508	10008	deposit	4000	2024-04-09
3132024509	10009	transfer	6000	2024-04-08
3132024511	10001	transfer	2000	2024-04-08

```
SELECT SUM(amount) AS 'Total Deposits'
FROM transaction
WHERE transaction_type='deposit'
AND transaction_date='2024-04-01';
```

Result Grid | Filter Rows: | Export:

Total Deposits
7000

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

Query:

SELECT MIN(join_date) AS Oldest customer, MAX(join_date) AS Newest customer
FROM customers;

Output:

customer_id	first_name	last_name	DOB	email	phone_number	address	city	join_date
1	hari	prakash	2000-02-08	hariom@gmail.com	8964232589	13th street	delhi	2020-05-18
2	mahesh	kannan	2000-10-08	mahesh09@gmail.com	8456785456	15th north street	mumbai	2020-07-11
3	suresh	roopan	2001-11-18	sureshk@gmail.com	9765456788	West side street	chennai	2019-07-06
4	dayananth	selvan	2000-09-20	dayasel@gmail.com	8456789768	22B baker street	delhi	2020-12-08
5	sri	dharshan	2002-07-04	sridhard@gmail.com	9876543456	uptown 18th street	bengaluru	2021-09-30
6	viswa	nathan	2000-05-01	viswa05@gmail.com	9765456787	17th downtown	mumbai	2019-01-30
7	vignesh	prasath	2001-12-09	vickyp@gmail.com	8765467898	hillside point	hyderabad	2021-11-03
8	dinesh	kumar	2001-10-07	dineshk07@gmail.com	9876545678	7th lake point	pune	2021-10-17
9	akash	prasath	2000-10-17	akashp@gmail.com	9876545679	23V Krypto street	noida	2020-04-13
10	sanjeev	kumar	2002-10-14	sanjeevk@gmail.com	9876543456	43 crvo street	trivandrum	2021-06-09

```
49 •  SELECT MIN(join_date) AS 'Oldest customer', MAX(join_date) AS 'Newest customer'
50    FROM customers;
```

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

Oldest customer	Newest customer
2019-01-30	2021-11-03

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

Query:

```
SELECT T.transaction_id,
T.transaction_type,T.amount,T.transaction_date,
A.account_type
FROM transaction T
RIGHT JOIN accounts A
ON T.account_id=A.account_id
ORDER BY account_type;
```

Output:

```
SELECT T.transaction_id, T.transaction_type,T.amount,T.transaction_date,
A.account_type
FROM transaction T
RIGHT JOIN accounts A
ON T.account_id=A.account_id
ORDER BY account_type;
```

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

transaction_id	transaction_type	amount	transaction_date	account_type
3132024502	transfer	5000	2024-04-05	current
3132024505	transfer	4000	2024-04-04	current
3132024509	transfer	6000	2024-04-08	current
3132024500	deposit	2000	2024-04-01	savings
3132024511	transfer	2000	2024-04-08	savings
3132024503	withdrawl	2000	2024-03-29	savings
3132024504	deposit	5000	2024-04-01	savings
3132024508	deposit	4000	2024-04-09	savings
3132024506	deposit	2000	2024-04-07	savings
3132024501	withdrawl	500	2024-04-03	zero_balance
3132024507	withdrawl	2000	2024-04-06	zero_balance

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

Query:

```
SELECT C.customer_id, CONCAT(first_name,last_name) AS
name, A.account_id,A.account_type,A.balance
FROM customers C
JOIN accounts A
ON C.customer_id=A.customer_id
ORDER BY C.customer_id;
```

Output:

```
SELECT C.customer_id, CONCAT(first_name,last_name) AS name, A.account_id,A.account_type,A.balance
FROM customers C
JOIN accounts A
ON C.customer_id=A.customer_id
ORDER BY C.customer_id;
```

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

customer_id	name	account_id	account_type	balance
1	hariprakash	10001	savings	10000
2	maheshkannan	10003	savings	12000
3	sureshroopan	10005	current	8000
4	dayananthselvan	10002	zero_balance	1000
5	sridharshan	10004	current	20000
6	viswanathan	10006	savings	13000
7	vigneshprasath	10008	savings	16000
8	dineshkumar	10007	zero_balance	2000
9	akashprasath	10009	current	25000
10	sanjeevkumar	10010	savings	9000

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

Query:

```
SELECT C.customer_id,CONCAT(C.first_name,C.last_name)
AS name,T.transaction_id,T.account_id,T.transaction_type,
T.amount,T.transaction_date
FROM transaction 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=10001;
```

Output:

```
SELECT C.customer_id,CONCAT(C.first_name,C.last_name)
AS name,T.transaction_id,T.account_id,T.transaction_type,
T.amount,T.transaction_date
FROM transaction 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=10001;
```

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

customer_id	name	transaction_id	account_id	transaction_type	amount	transaction_date
1	hariprakash	3132024500	10001	deposit	2000	2024-04-01
1	hariprakash	3132024511	10001	transfer	2000	2024-04-08

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

Query:

```
SELECT customer_id,COUNT(*) AS 'Count'
FROM accounts
GROUP BY customer_id
HAVING COUNT(*)>1;
```

Output:

account_id	customer_id	account_type	balance
10001	1	savings	10000
10002	4	zero_balance	1000
10003	2	savings	12000
10004	5	current	20000
10005	3	current	8000
10006	6	savings	13000
10007	8	zero_balance	2000
10008	7	savings	16000
10009	9	current	25000
10010	10	savings	9000
10012	10	zero_balance	5000
10013	8	savings	9000

```
SELECT customer_id,COUNT(*) AS 'Count'
FROM accounts
GROUP BY customer_id
HAVING COUNT(*)>1;
```

Result Grid | Filter Rows: | Export: | W

customer_id	Count
8	2
10	2

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

Query:

```
SELECT
        SUM(CASE
                WHEN transaction_type='deposit' THEN amount
ELSE 0 END)   -

         SUM(CASE
                WHEN transaction_type='withdrawl' THEN
amount ELSE 0 END)
AS total_Difference
FROM transaction;
```

Output:

```
SELECT
    SUM(CASE
            WHEN transaction_type='deposit' THEN amount ELSE 0 END)
     -
    SUM(CASE
            WHEN transaction_type='withdrawl' THEN amount ELSE 0 END)
    AS total_Difference
    FROM transaction;
```

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

total_Difference
8500

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

Query:

```
SELECT account_id,
AVG(amount) AS avg_daily_balance
FROM transaction
WHERE transaction_date BETWEEN '2024-04-01' AND '2024-04-03'
GROUP BY account_id;
```

Output:

transaction_id	account_id	transaction_type	amount	transaction_date
3132024500	10001	deposit	2000	2024-04-02
3132024501	10002	withdrawl	-1000	2024-04-03
3132024502	10004	transfer	-5000	2024-04-05
3132024503	10003	withdrawl	-2000	2024-04-03
3132024504	10006	deposit	5000	2024-04-01
3132024505	10005	transfer	-7000	2024-04-04
3132024506	10010	deposit	2000	2024-04-07
3132024507	10007	withdrawl	-2000	2024-04-06
3132024508	10008	deposit	4000	2024-04-09
3132024509	10009	transfer	-6000	2024-04-08
3132024511	10001	transfer	-2000	2024-04-02
NULL	10015	NULL	NULL	NULL
31320245012	10002	deposit	2000	2024-04-03

```
SELECT account_id,
AVG(amount) AS avg_daily_balance
FROM transaction
WHERE transaction_date BETWEEN '2024-04-01' AND '2024-04-03'
GROUP BY account_id;
```

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

account_id	average_daily_balance
10001	0.0000
10002	500.0000
10003	-2000.0000
10006	5000.0000

11. Calculate the total balance for each account type.

Query:

SELECT account_type, SUM(balance) AS ‘total balance’
FROM accounts
GROUP BY account_type;

Output:

```
SELECT account_type, SUM(balance) AS 'total balance'
FROM accounts
GROUP BY account_type;
```

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

account_type	total balance
savings	69000
zero_balance	8000
current	53000

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

Query:

SELECT account_id, COUNT(*) AS ‘count’
FROM transaction
GROUP BY account_id
ORDER BY ‘count’ DESC;

Output:

```
83 •  SELECT account_id, COUNT(*) AS 'count'
84    FROM transaction
85    GROUP BY account_id
86    ORDER BY 'count' DESC;
```

Result Grid | Filter Rows: | Export: | W

account_id	count
10001	2
10002	1
10003	1
10004	1
10005	1
10006	1
10007	1
10008	1
10009	1
10010	1

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

Query:

```
With TotalBalance AS(
SELECT customer_id, SUM(balance) AS tot_bal
FROM accounts
GROUP BY customer_id)
SELECT C.customer_id,
CONCAT(C.first_name,C.last_name)AS name,
A.account_type,T.tot_bal
FROM customers C
JOIN TotalBalance T
ON C.customer_id = T.customer_id
JOIN accounts A
ON A.customer_id=C.customer_id
WHERE tot_bal>15000;
```

Output:

```
158 • With TotalBalance AS(
159     SELECT customer_id, SUM(balance) AS tot_bal
160     FROM accounts
161     GROUP BY customer_id)
162     SELECT C.customer_id, CONCAT(C.first_name,C.last_name)AS name,
163     A.account_type,T.tot_bal
164     FROM customers C
165     JOIN TotalBalance T
166     ON C.customer_id = T.customer_id
167     JOIN accounts A
168     ON A.customer_id=C.customer_id
169     WHERE tot_bal>15000;
170
```

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

customer_id	name	account_type	tot_bal
5	sridharshan	current	20000
7	vigneshprasath	savings	16000
9	akashprasath	current	25000

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

Query:

SELECT account_id, amount, transaction_date ,COUNT(*) AS dupcount
FROM transaction
GROUP BY amount, transaction_date, account_id HAVING COUNT(*) > 1;

Output:

```
238 •  SELECT account_id, amount, transaction_date ,COUNT(*) AS dupcount
239    FROM transaction
240    GROUP BY amount, transaction_date, account_id HAVING COUNT(*) > 1;
241
```

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

account_id	amount	transaction_date	dupcount
10010	2000	2024-04-07	2
10001	-2000	2024-04-02	2