Task 4

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

Query:

```
SELECT customer_id , CONCAT(first_name,last_name) AS name
FROM customers
WHERE customer_id=(SELECT customer_id FROM accounts
WHERE balance=(SELECT MAX(balance) FROM accounts));
```

Output:

```
242 •  SELECT customer_id , CONCAT(first_name,last_name) AS name
243    FROM customers
244    WHERE customer_id=(SELECT customer_id FROM accounts
245    WHERE balance=(SELECT MAX(balance) FROM accounts));
246
```

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

customer_id	name
9	akashprasath

account_id	customer_id	account_type	balance
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
10014	8	savings	4000
10015	12	savings	4000

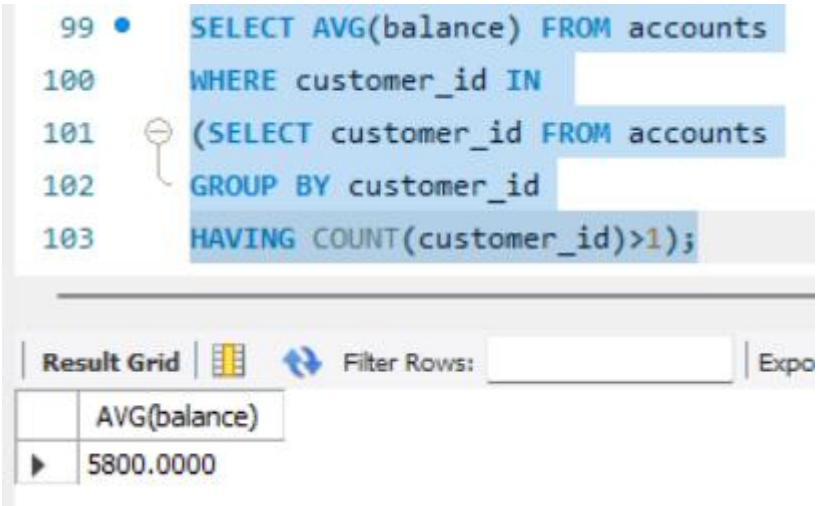
2. Calculate the average account balance for customers who have more than one account.

Query:

```
SELECT AVG(balance) FROM accounts
WHERE customer_id IN
(SELECT customer_id FROM accounts
GROUP BY customer_id
HAVING COUNT(customer_id)>1);
```

Output:

account_id	customer_id	account_type	balance
10007	8	zero_balance	2000
10013	8	savings	9000
10014	8	savings	4000
10010	10	savings	9000
10012	10	zero_balance	5000

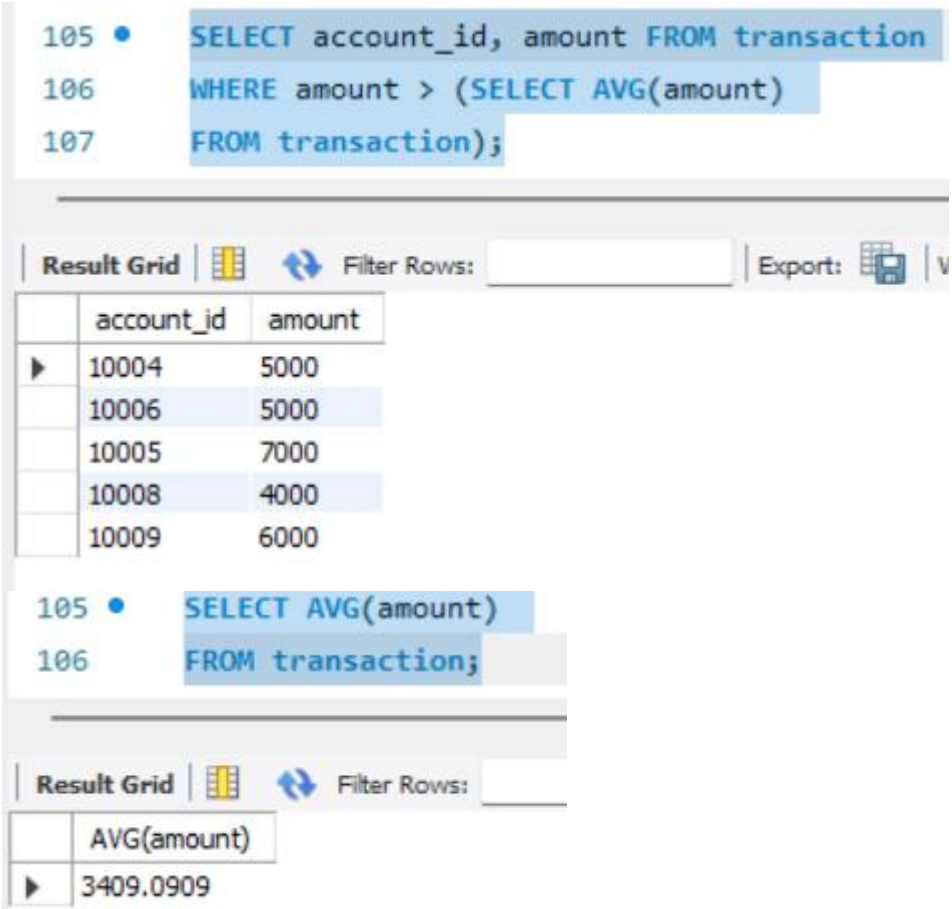


3. Retrieve accounts with transactions whose amounts exceed the average transaction amount.

Query:

```
SELECT account_id FROM transaction
WHERE amount > (SELECT AVG(amount)
FROM transaction);
```

Output:



```
105 •   SELECT account_id, amount FROM transaction
106     WHERE amount > (SELECT AVG(amount)
107     FROM transaction);
```

Result Grid | Filter Rows: | Export:

account_id	amount
10004	5000
10006	5000
10005	7000
10008	4000
10009	6000

```
105 •   SELECT AVG(amount)
106     FROM transaction;
```

Result Grid | Filter Rows:

AVG(amount)
3409.0909

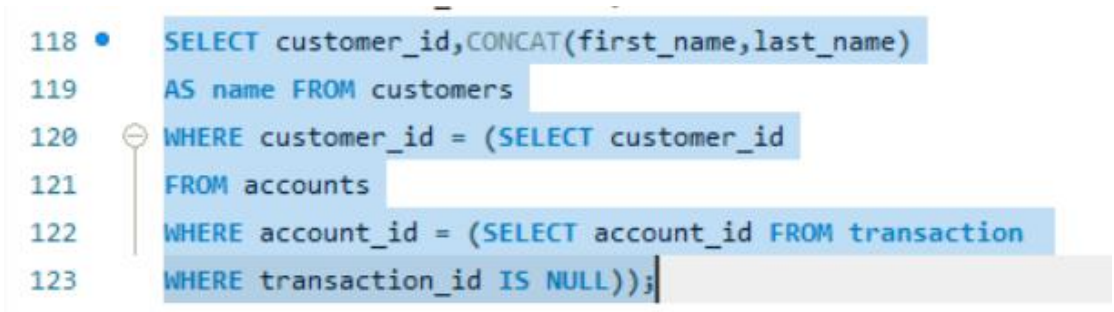
4. Identify customers who have no recorded transactions.

Query:

```
SELECT customer_id,CONCAT(first_name,last_name) AS
name
FROM customers
WHERE customer_id = (SELECT customer_id
FROM accounts
WHERE account_id = (SELECT account_id FROM transaction
WHERE transaction_id IS NULL));
```

Output:

transaction_id	account_id	transaction_type	amount	transaction_date
3132024502	10004	transfer	5000	2024-04-05
3132024503	10003	withdrawl	2000	2024-03-29
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	2024-04-08
NULL	10015	NULL	NULL	NULL



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

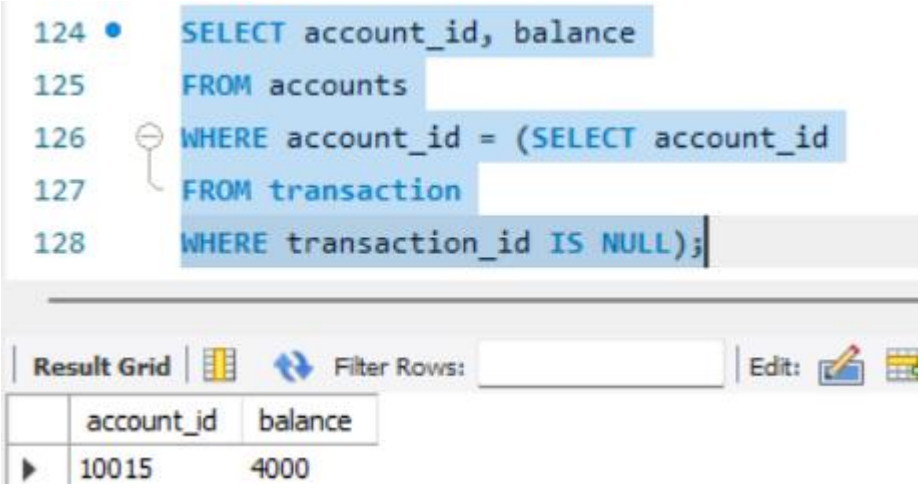
customer_id	name
12	jitheshsharma

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

Query:

```
SELECT balance
FROM accounts
WHERE account_id = (SELECT account_id
FROM transaction
WHERE transaction_id IS NULL);
```

Output:



6. Retrieve transactions for accounts with the lowest balance.

Query:

```
SELECT transaction_id,transaction_type,amount,
transaction_date
FROM transaction
WHERE account_id = (SELECT account_id
FROM accounts
WHERE balance =(SELECT MIN(balance)
FROM accounts));
```

Output:

```
129 •  SELECT account_id,transaction_id,transaction_type,amount,
130    transaction_date
131    FROM transaction
132    WHERE account_id = (SELECT account_id
133    FROM accounts
134    WHERE balance =(SELECT MIN(balance)
135    FROM accounts));
```

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

account_id	transaction_id	transaction_type	amount	transaction_date
10002	3132024501	withdrawl	500	2024-04-03

7. Identify customers who have accounts of multiple types.

Query:

```
SELECT customer_id,CONCAT(first_name,last_name)
AS PeopleWithMultipleTypeAccounts
FROM customers
WHERE customer_id IN (SELECT DISTINCT(customer_id)
FROM accounts
WHERE customer_id IN (SELECT customer_id
FROM accounts
GROUP BY customer_id
HAVING COUNT(customer_id)>1));
```

Output:

customer_id	account_type
8	zero_balance
10	savings
10	zero_balance
8	savings
8	savings

customer_id	account_type
8	zero_balance
10	savings
10	zero_balance
8	savings

```
136  SELECT customer_id,CONCAT(first_name,last_name)
137  AS PeopleWithMultipleTypeAccounts FROM customers
138  WHERE customer_id IN (SELECT DISTINCT(customer_id)
139  FROM accounts
140  WHERE customer_id IN (SELECT customer_id
141  FROM accounts
142  GROUP BY customer_id
143  HAVING COUNT(customer_id)>1));
```

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

customer_id	PeopleWithMultipleTypeAccounts
8	dineshkumar
10	sanjeevkumar

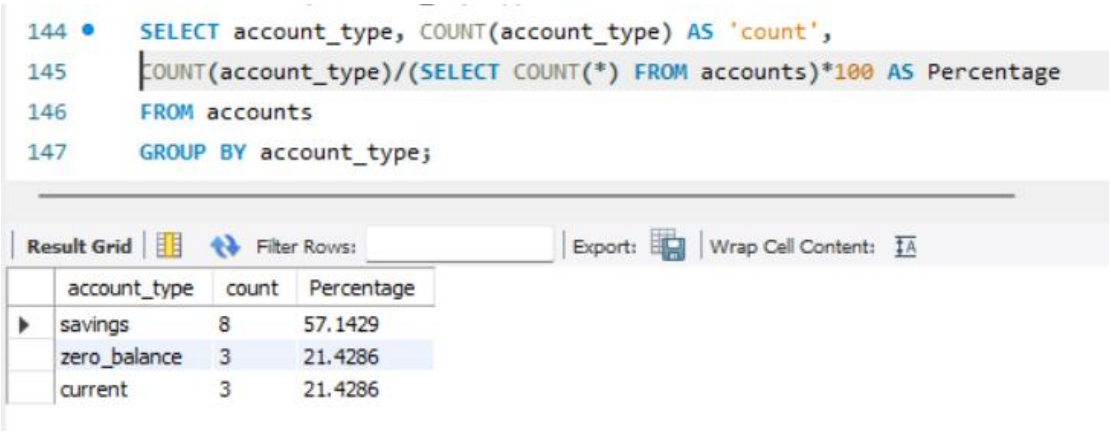
8. Calculate the percentage of each account type out of the total number of accounts.

Query:

SELECT account_type, COUNT(account_type) AS 'count', COUNT(account_type)/(SELECT COUNT(*) FROM accounts)*100 AS Percentage

FROM accounts
GROUP BY account_type;

Output:



```
144 •  SELECT account_type, COUNT(account_type) AS 'count',
145    COUNT(account_type)/(SELECT COUNT(*) FROM accounts)*100 AS Percentage
146    FROM accounts
147    GROUP BY account_type;
```

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

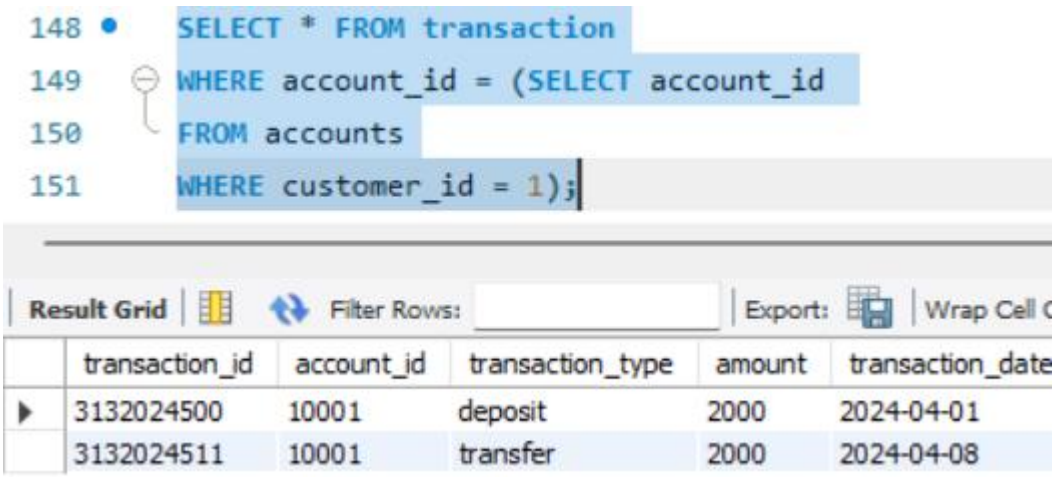
account_type	count	Percentage
savings	8	57.1429
zero_balance	3	21.4286
current	3	21.4286

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

Query:

SELECT * FROM transaction
WHERE account_id = (SELECT account_id
FROM accounts
WHERE customer_id = 1);

Output:



```
148 •  SELECT * FROM transaction
149    WHERE account_id = (SELECT account_id
150    FROM accounts
151    WHERE customer_id = 1);
```

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

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

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

Query:

```
SELECT account_type,SUM(balance) AS total_balance,
(SELECT COUNT(*) FROM accounts A
WHERE A.account_type=accounts.account_type)
AS account_type_count
FROM accounts
GROUP BY account_type;
```

Output: