

SQL Queries - Joins and Window Functions

SQL Queries

1. Retrieve all details of clients from the client table.

```
SELECT * FROM client;
```

2. Get the account IDs and their corresponding opening dates from the account table.

```
SELECT account_id, date FROM account;
```

3. List all transactions where the amount is greater than 5000.

```
SELECT * FROM trans WHERE amount > 5000;
```

4. Count the total number of loans in the loan table.

```
SELECT COUNT(*) FROM loan;
```

5. Find the average amount of all loans.

```
SELECT AVG(amount) AS avg_loan_amount FROM loan;
```

6. Retrieve all clients and their associated accounts.

```
SELECT c.client_id, c.gender, a.account_id, a.date
FROM client c
JOIN disp d ON c.client_id = d.client_id
JOIN account a ON d.account_id = a.account_id;
```

7. Get all transactions with the corresponding account details.

```
SELECT t.trans_id, t.account_id, t.amount, a.date AS account_creation_date
FROM trans t
JOIN account a ON t.account_id = a.account_id;
```

8. Show loans along with the clients who took them.

```
SELECT l.loan_id, l.amount, l.status, c.client_id, c.gender
FROM loan l
JOIN account a ON l.account_id = a.account_id
JOIN disp d ON a.account_id = d.account_id
JOIN client c ON d.client_id = c.client_id;
```

9. Retrieve all orders and the clients who placed them.

```
SELECT o.order_id, o.amount, c.client_id, c.gender
FROM 'order' o
JOIN account a ON o.account_id = a.account_id
JOIN disp d ON a.account_id = d.account_id
JOIN client c ON d.client_id = c.client_id;
```

10. Get all cards issued to clients along with their account details.

```

SELECT card.card_id , card.type AS card_type , c.client_id , c.gender , a.account_id
FROM card
JOIN disp d ON card.disp_id = d.disp_id
JOIN client c ON d.client_id = c.client_id
JOIN account a ON d.account_id = a.account_id;

```

11. Find the number of accounts in each district.

```

SELECT district_id , COUNT(*) AS account_count
FROM account
GROUP BY district_id;

```

12. Get the total loan amount per district.

```

SELECT a.district_id , SUM(l.amount) AS total_loan_amount
FROM loan l
JOIN account a ON l.account_id = a.account_id
GROUP BY a.district_id;

```

13. Find the total number of transactions per transaction type.

```

SELECT type , COUNT(*) AS total_transactions
FROM trans
GROUP BY type;

```

14. Retrieve the maximum transaction amount per client.

```

SELECT c.client_id , MAX(t.amount) AS max_transaction_amount
FROM trans t
JOIN account a ON t.account_id = a.account_id
JOIN disp d ON a.account_id = d.account_id
JOIN client c ON d.client_id = c.client_id
GROUP BY c.client_id;

```

15. Rank clients by the total amount of their transactions.

```

SELECT c.client_id , SUM(t.amount) AS total_spent ,
       RANK() OVER (ORDER BY SUM(t.amount) DESC) AS spending_rank
FROM trans t
JOIN account a ON t.account_id = a.account_id
JOIN disp d ON a.account_id = d.account_id
JOIN client c ON d.client_id = c.client_id
GROUP BY c.client_id;

```

16. Assign a row number to each loan ordered by amount.

```

SELECT loan_id , account_id , amount ,
       ROWNUMBER() OVER (ORDER BY amount DESC) AS row_num
FROM loan;

```

17. Find the cumulative sum of transactions for each account.

```

SELECT account_id , trans_id , amount ,
       SUM(amount) OVER (PARTITION BY account_id ORDER BY date) AS cumulative_sum
FROM trans;

```

18. Compute the moving average of loan payments per account.

```

SELECT loan_id , account_id , payments ,
       AVG(payments) OVER (PARTITION BY account_id ORDER BY date ROWS BETWEEN 2
FROM loan;

```

19. Get the lead transaction amount per account.

```
SELECT account_id , trans_id , amount ,  
       LEAD(amount) OVER (PARTITION BY account_id ORDER BY date) AS next_transa  
FROM trans ;
```

20. Get the lag transaction amount per account.

```
SELECT account_id , trans_id , amount ,  
       LAG(amount) OVER (PARTITION BY account_id ORDER BY date) AS previous_tran  
FROM trans ;
```