Questions

1. Find the total balance held in accounts of each Account_Type across all branches.

```
mysql> SELECT Account_Type, SUM(Balance) AS Total_Balance
    -> FROM account
    -> GROUP BY Account_Type;
+--------------+---------------+
| Account_Type | Total_Balance |
+--------------+---------------+
| Savings      |     387004.25 |
| Current      |    3170000.75 |
+--------------+---------------+
2 rows in set (0.08 sec)
```

2. List the branch names and the total number of accounts in each branch.

```
mysql> SELECT b.Name AS Branch_Name, COUNT(a.Id) AS Total_Accounts
    -> FROM branch b
    -> JOIN account a ON b.BCode = a.BCode
    -> GROUP BY b.Name;
+--------------+----------------+
| Branch_Name  | Total_Accounts |
+--------------+----------------+
| Main Branch  |              5 |
| South Branch |              5 |
| East Branch  |              5 |
| West Branch  |              5 |
+--------------+----------------+
4 rows in set (0.05 sec)
```

3. Retrieve the top 3 accounts with the highest balance.

```
mysql> SELECT Name, Account_Number, Balance
```

```
    -> FROM account
    -> ORDER BY Balance DESC
    -> LIMIT 3;
+--------------+----------------+---------+
| Name         | Account_Number | Balance |
+--------------+----------------+---------+
| Sneha Kapoor | ACC001237      |  800000 |
| Harish Yadav | ACC001246      |  700000 |
| Meera Iyer   | ACC001243      |  600000 |
+--------------+----------------+---------+
3 rows in set (0.01 sec)
```

4. Find the total number of transactions for each account and the average transaction amount per account.

```
mysql> SELECT Account_Num, COUNT(Id) AS Total_Transactions, AVG(Amount) AS Avg_Transaction_Amount
    -> FROM transaction
    -> GROUP BY Account_Num;
+-------------+--------------------+------------------------+
| Account_Num | Total_Transactions | Avg_Transaction_Amount |
+-------------+--------------------+------------------------+
| ACC001234   |                  8 |                   8750 |
| ACC001235   |                  8 |                  14000 |
| ACC001236   |                  8 |                   7125 |
| ACC001237   |                  8 |                   8500 |
| ACC001238   |                  8 |                   5250 |
| ACC001239   |                  8 |                12687.5 |
| ACC001240   |                  8 |                   9250 |
| ACC001241   |                  8 |                   8750 |
| ACC001242   |                  8 |                   8500 |
| ACC001243   |                  8 |                   8500 |
| ACC001244   |                  8 |                  11750 |
| ACC001245   |                  8 |                  15375 |
| ACC001246   |                  8 |                   7375 |
| ACC001247   |                  8 |                   7875 |
| ACC001248   |                  8 |                   6250 |
| ACC001249   |                  6 |     11416.666666666666 |
```

```
| ACC001250   |                  6 |                   7000 |
| ACC001251   |                  6 |                  11000 |
| ACC001252   |                  6 |      5833.333333333333 |
| ACC001253   |                  6 |     10833.333333333334 |
+-------------+--------------------+------------------------+
20 rows in set (0.03 sec)
```

5. List all services that were provided in the past 6 months, along with their transaction details.

```
mysql> SELECT s.Service_Type, s.Service_Amount, s.Service_Date, t.Account_Num, t.Transaction_Type
    -> FROM service s
    -> JOIN transaction t ON s.Transaction_Id = t.Id
    -> WHERE s.Service_Date >= CURDATE() - INTERVAL 6 MONTH;
+--------------------+----------------+--------------+-------------+------------------+
| Service_Type       | Service_Amount | Service_Date | Account_Num | Transaction_Type |
+--------------------+----------------+--------------+-------------+------------------+
| Account Setup      |         100.00 | 2024-11-01   | ACC001234   | Deposit          |
| Account Maintenance |          50.00 | 2024-11-02   | ACC001235   | Withdrawal       |
| Account Setup      |         100.00 | 2024-11-03   | ACC001236   | Deposit          |
| Account Maintenance |          50.00 | 2024-11-04   | ACC001237   | Withdrawal       |
| ATM Withdrawal     |          20.00 | 2024-11-05   | ACC001238   | Deposit          |
| Transfer Fee       |          15.00 | 2024-11-06   | ACC001234   | Transfer         |
| Account Setup      |         100.00 | 2024-11-07   | ACC001235   | Deposit          |
| ATM Withdrawal     |          20.00 | 2024-11-08   | ACC001236   | Withdrawal       |
| Account Maintenance |          50.00 | 2024-11-09   | ACC001237   | Deposit          |
| Transfer Fee       |          15.00 | 2024-11-10   | ACC001238   | Transfer         |
| Account Setup      |         100.00 | 2024-11-11   | ACC001239   | Deposit          |
| Account Maintenance |          50.00 | 2024-11-12   | ACC001240   | Withdrawal       |
| Account Setup      |         100.00 | 2024-11-13   | ACC001241   | Deposit          |
| Account Maintenance |          50.00 | 2024-11-14   | ACC001242   | Withdrawal       |
| ATM Withdrawal     |          20.00 | 2024-11-15   | ACC001243   | Deposit          |
| Transfer Fee       |          15.00 | 2024-11-16   | ACC001244   | Transfer         |
| Account Setup      |         100.00 | 2024-11-17   | ACC001245   | Withdrawal       |
| ATM Withdrawal     |          20.00 | 2024-11-18   | ACC001246   | Deposit          |
| Account Maintenance |          50.00 | 2024-11-19   | ACC001247   | Withdrawal       |
```

```
| Transfer Fee        |          15.00 | 2024-11-20   | ACC001248   | Transfer         |
| Account Setup       |         100.00 | 2024-11-21   | ACC001249   | Deposit          |
| Account Maintenance |          50.00 | 2024-11-22   | ACC001250   | Withdrawal       |
| Account Setup       |         100.00 | 2024-11-23   | ACC001251   | Deposit          |
| Account Maintenance |          50.00 | 2024-11-24   | ACC001252   | Withdrawal       |
| ATM Withdrawal      |          20.00 | 2024-11-25   | ACC001253   | Deposit          |
| Transfer Fee        |          15.00 | 2024-11-26   | ACC001234   | Transfer         |
| Account Setup       |         100.00 | 2024-11-27   | ACC001235   | Deposit          |
| ATM Withdrawal      |          20.00 | 2024-11-28   | ACC001236   | Withdrawal       |
| Account Maintenance |          50.00 | 2024-11-29   | ACC001237   | Deposit          |
| Transfer Fee        |          15.00 | 2024-11-30   | ACC001238   | Transfer         |
+---------------------+----------------+--------------+-------------+------------------+
30 rows in set (0.01 sec)
```

6. Find employees working at branches that have accounts with a balance greater than 1,00,000.

```
mysql> SELECT DISTINCT e.Name AS Employee_Name, e.Branch
    -> FROM employee e
    -> JOIN branch b ON e.Branch = b.Name
    -> JOIN account a ON b.BCode = a.BCode
    -> WHERE a.Balance > 100000;
+-----------------+--------------+
| Employee_Name   | Branch       |
+-----------------+--------------+
| Anil Mehta      | Main Branch  |
| Deepika Rao     | South Branch |
| Rina Banerjee   | West Branch  |
| Manoj Sharma    | Main Branch  |
| Alka Singh      | South Branch |
| Shweta Deshmukh | West Branch  |
| Vikram Chauhan  | Main Branch  |
| Neha Dubey      | South Branch |
+-----------------+--------------+
8 rows in set (0.02 sec)
```

7. Retrieve all accounts that have never availed any services.

```
mysql> SELECT a.Account_Number, a.Name
    -> FROM account a
    -> WHERE a.Account_Number NOT IN (
    ->   SELECT DISTINCT t.Account_Num
    ->   FROM transaction t
    ->   JOIN service s ON t.Id = s.Transaction_Id
    -> );
Empty set (0.00 sec)
```

8. Calculate the total service amount provided by each branch.

```
mysql> SELECT b.Name AS Branch_Name, SUM(s.Service_Amount) AS Total_Service_Amount
    -> FROM branch b
    -> JOIN account a ON b.BCode = a.BCode
    -> JOIN transaction t ON a.Account_Number = t.Account_Num
    -> JOIN service s ON t.Id = s.Transaction_Id
    -> GROUP BY b.Name;
+--------------+----------------------+
| Branch_Name  | Total_Service_Amount |
+--------------+----------------------+
| Main Branch  |               300.00 |
| South Branch |               470.00 |
| East Branch  |               320.00 |
| West Branch  |               470.00 |
+--------------+----------------------+
4 rows in set (0.01 sec)
```

9. List accounts along with the number of services they have availed and the total service amount.

```
mysql> SELECT a.Account_Number, a.Name, COUNT(s.Id) AS Total_Services, SUM(s.Service_Amount) AS
```

```
Total_Service_Amount
    -> FROM account a
    -> JOIN transaction t ON a.Account_Number = t.Account_Num
    -> JOIN service s ON t.Id = s.Transaction_Id
    -> GROUP BY a.Account_Number, a.Name;
```

Account_Number	Name	Total_Services	Total_Service_Amount
ACC001234	Rahul Sharma	3	130.00
ACC001235	Priya Nair	3	250.00
ACC001236	Arjun Das	3	140.00
ACC001237	Sneha Kapoor	3	150.00
ACC001238	Amit Joshi	3	50.00
ACC001239	Neha Verma	1	100.00
ACC001240	Rohan Gupta	1	50.00
ACC001241	Sunita Roy	1	100.00
ACC001242	Vikas Singh	1	50.00
ACC001243	Meera Iyer	1	20.00
ACC001244	Nitin Kumar	1	15.00
ACC001245	Anjali Mehta	1	100.00
ACC001246	Harish Yadav	1	20.00
ACC001247	Pooja Reddy	1	50.00
ACC001248	Siddharth Chatterjee	1	15.00
ACC001249	Kavita Mishra	1	100.00
ACC001250	Arvind Khanna	1	50.00
ACC001251	Divya Narayan	1	100.00
ACC001252	Rajesh Sen	1	50.00
ACC001253	Smita Kulkarni	1	20.00

```
20 rows in set (0.01 sec)
```

10. Find the youngest account holder in each branch and their account balance.

```
mysql> SELECT b.Name AS Branch_Name, a.Name AS Account_Holder, a.Balance, a.DOB
    -> FROM branch b
```

```
    -> JOIN account a ON b.BCode = a.BCode
    -> WHERE (b.BCode, a.DOB) IN (
    ->   SELECT BCode, MIN(DOB)
    ->   FROM account
    ->   GROUP BY BCode
    -> );
+--------------+----------------+---------+------------+
| Branch_Name  | Account_Holder | Balance | DOB        |
+--------------+----------------+---------+------------+
| South Branch | Priya Nair     |  500000 | 1988-03-12 |
| East Branch  | Neha Verma     |   32000 | 1993-11-22 |
| West Branch  | Sunita Roy     |   45000 | 1975-03-30 |
| Main Branch  | Arvind Khanna  |  450000 | 1978-08-08 |
+--------------+----------------+---------+------------+
4 rows in set (0.01 sec)
```

11. Update the balance of a specific account.

```
mysql> UPDATE account
    -> SET Balance = Balance + 5000
    -> WHERE Account_Number = 'ACC001234';
Query OK, 1 row affected (0.06 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

12. Mark inactive accounts with a balance below 1000.

```
mysql> ALTER TABLE account
    -> ADD COLUMN Status VARCHAR(15) DEFAULT 'Active';
Query OK, 0 rows affected (0.10 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
mysql> UPDATE account
    -> SET Status = 'Inactive'
```

```
    -> WHERE Balance < 1000;
Query OK, 0 rows affected (0.00 sec)
Rows matched: 0  Changed: 0  Warnings: 0
```

13. Delete accounts that have no associated transactions.

```
DELETE FROM account
    -> WHERE Account_Number NOT IN (
    ->   SELECT DISTINCT Account_Num FROM transaction
    -> );
Query OK, 0 rows affected (0.01 sec)
```

14. Add a new column to track the last transaction date for each account.

```
mysql> ALTER TABLE account
    -> ADD COLUMN Last_Transaction_Date DATE;
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
mysql> UPDATE account a
    -> SET Last_Transaction_Date = (
    ->   SELECT MAX(Date)
    ->   FROM transaction t
    ->   WHERE t.Account_Num = a.Account_Number
    -> );
Query OK, 20 rows affected (0.03 sec)
Rows matched: 20  Changed: 20  Warnings: 0
```

15. Update service charges for all transactions of a specific type.

```
mysql> UPDATE service s
    -> JOIN transaction t ON s.Transaction_Id = t.Id
    -> SET s.Service_Amount = s.Service_Amount * 1.1
```

```
    -> WHERE t.Transaction_Type = 'Withdrawal';
Query OK, 10 rows affected (0.01 sec)
Rows matched: 10  Changed: 10  Warnings: 0
```

16. Remove employees from branches with no accounts.

```
mysql> DELETE FROM employee
    -> WHERE Branch NOT IN (
    ->   SELECT DISTINCT b.Name
    ->   FROM branch b
    ->   JOIN account a ON b.BCode = a.BCode
    -> );
Query OK, 0 rows affected (0.01 sec)
```

17. Reset all balances to zero for accounts marked as inactive.

```
mysql> UPDATE account
    -> SET Balance = 0
    -> WHERE Status = 'Inactive';
Query OK, 0 rows affected (0.01 sec)
Rows matched: 0  Changed: 0  Warnings: 0
```

18. Rename the Service_Type column to Service_Name.

```
mysql> ALTER TABLE service
    -> CHANGE COLUMN Service_Type Service_Name VARCHAR(255);
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

19. Delete all services older than 1 year.

```
mysql> DELETE FROM service
```

```
    -> WHERE Service_Date < CURDATE() - INTERVAL 1 YEAR;
Query OK, 0 rows affected (0.00 sec)
```

20. Add a column to the transaction table to track transaction status.

```
mysql> ALTER TABLE transaction
    -> ADD COLUMN Transaction_Status VARCHAR(15) DEFAULT 'Pending';
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
mysql> UPDATE transaction
    -> SET Transaction_Status = 'Completed'
    -> WHERE Date < CURDATE() - INTERVAL 7 DAY;
Query OK, 132 rows affected (0.02 sec)
Rows matched: 132  Changed: 132  Warnings: 0
```