# Lab 8 Report

Name : Mashrur Ahsan

ID : 200042115

Program : SWE

Department : CSE

Course : CSE 4308

The Scenario:

Consider the (partial) schema of a Banking Management System shown in Figure 1:

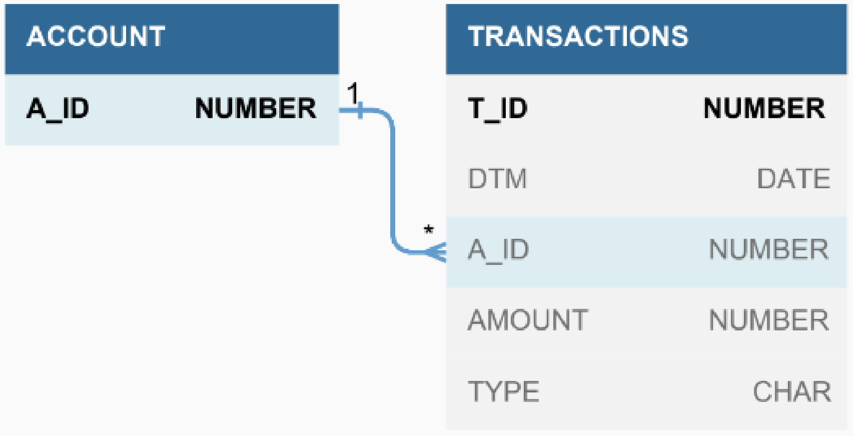


Figure 1: Schema for Lab Task

For simplicity, the ACCOUNT table contains only the account ID. The TRANSACTIONS table contains the information related to each transactions that occurs in the bank. The table stores transaction ID, date of the transaction, account involved in the transaction, the amount transacted, and the type of the transaction. The TYPE column is set to 0 if the money is credited to the account, i.e., the money is added to the account, and 1 if the money is debited from the account, i.e., the money is subtracted from the account. There are 3 types of accounts in the bank:

1. **Commercially Important Person (CIP):** If the person has a balance of more than 1, 000, 000 and all the transactions that they have made totals more than 5, 000, 000.
2. **Very Important Person (VIP):** If the person has a balance of more than 500, 000 but less than 900, 000 and all the transactions that they have made totals more than 2, 500, 000 but less than 4, 500, 000.
3. **Ordinary Person (OP):** If the user has a balance less than 100, 000 and all the transactions they have made totals less than 1, 000, 000.

Problem Statements:

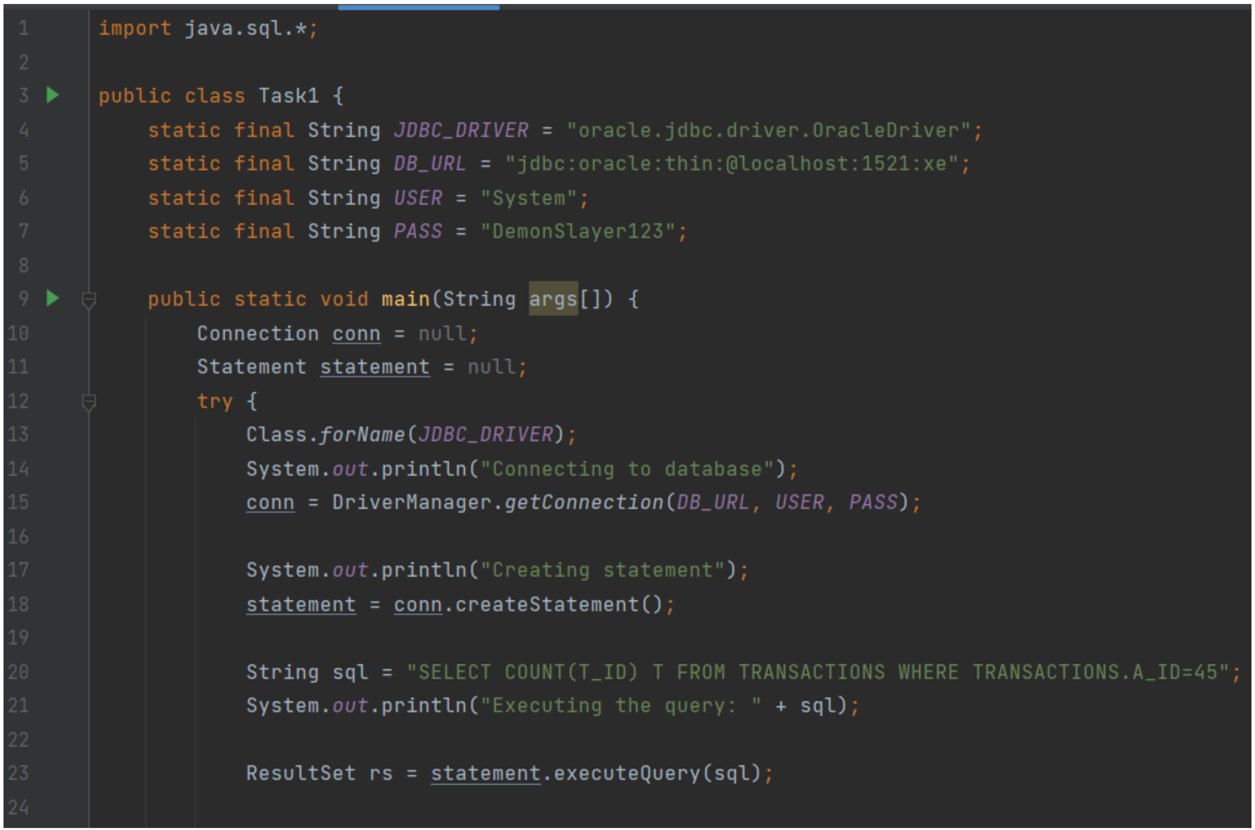
Write a JAVA code to:

1. Count the total number of transactions conducted under account 45.
2. Count the number of debits.
3. List the transactions that occurred in the year 2020.
4. Count the number of CIP, VIP, and OPs. Also show the number of people that do not fall in any of the categories.

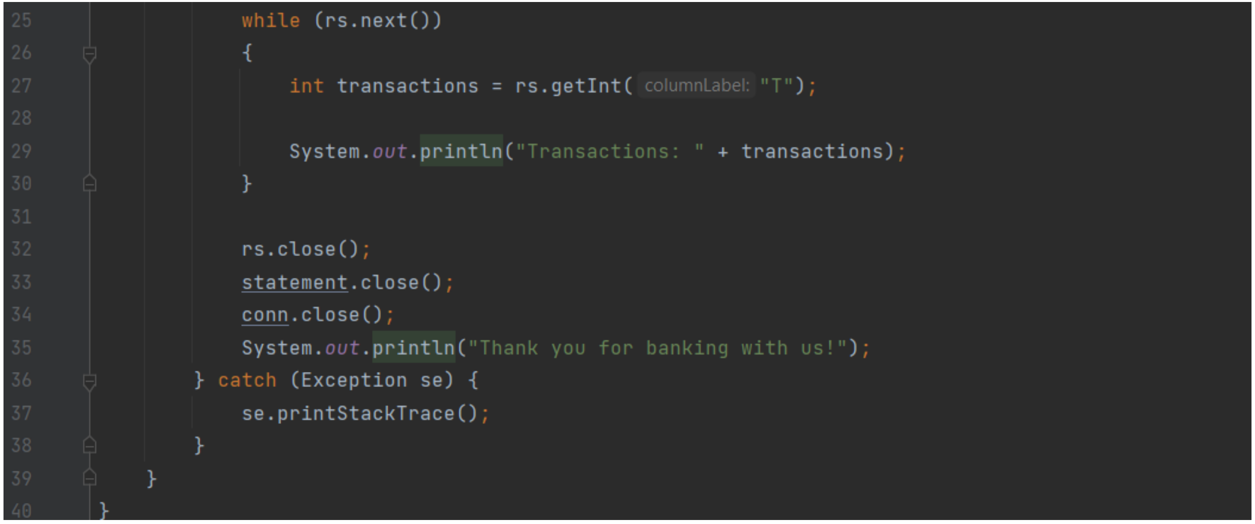
# Task 1

1. Count the total number of transactions conducted under account 45.

The Solution code:



```java
import java.sql.*;

public class Task1 {
    static final String JDBC_DRIVER = "oracle.jdbc.driver.OracleDriver";
    static final String DB_URL = "jdbc:oracle:thin:@localhost:1521:xe";
    static final String USER = "System";
    static final String PASS = "DemonSlayer123";

    public static void main(String args[]) {
        Connection conn = null;
        Statement statement = null;
        try {
            Class.forName(JDBC_DRIVER);
            System.out.println("Connecting to database");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);

            System.out.println("Creating statement");
            statement = conn.createStatement();

            String sql = "SELECT COUNT(T_ID) T FROM TRANSACTIONS WHERE TRANSACTIONS.A_ID=45";
            System.out.println("Executing the query: " + sql);

            ResultSet rs = statement.executeQuery(sql);

```



```java
            while (rs.next())
            {
                int transactions = rs.getInt( columnLabel: "T");

                System.out.println("Transactions: " + transactions);
            }

            rs.close();
            statement.close();
            conn.close();
            System.out.println("Thank you for banking with us!");
        } catch (Exception se) {
            se.printStackTrace();
        }
    }
}
```

Explanation:

Line 20: The SQL query has been written down

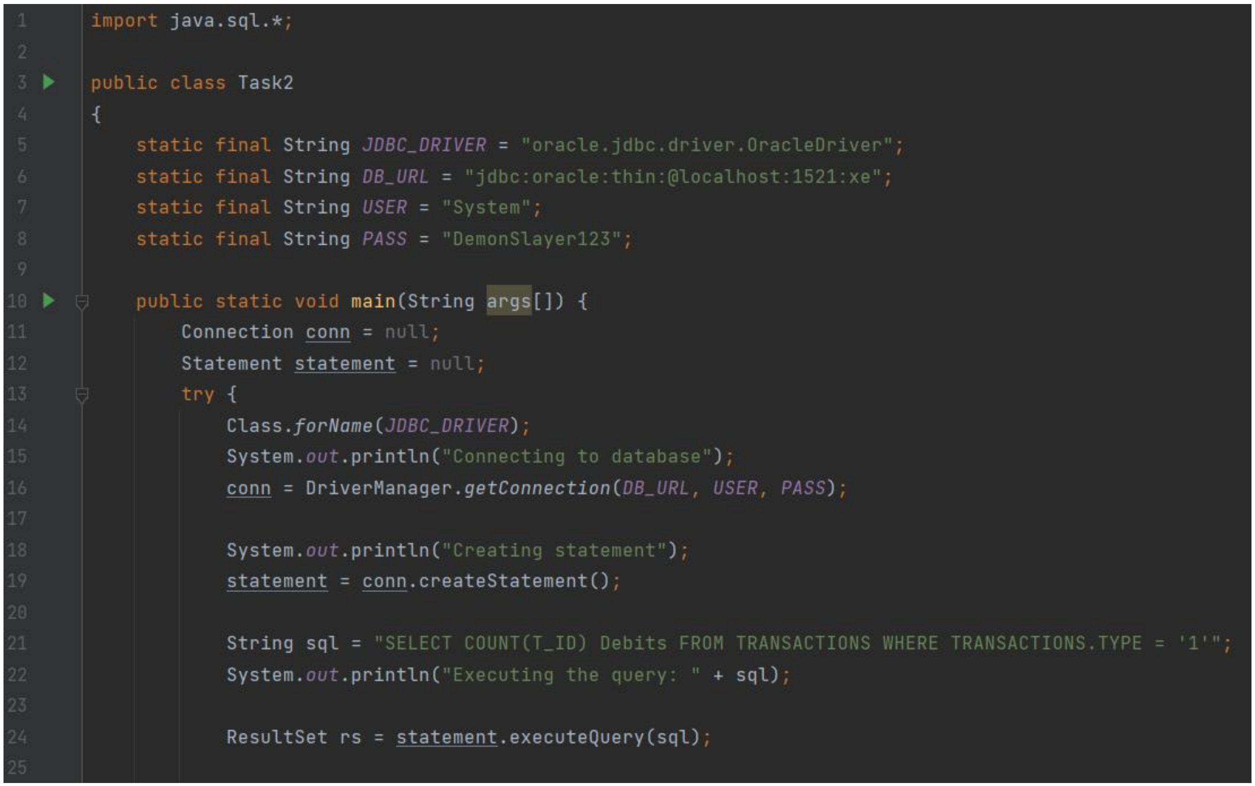
Line 27: Storing the number of transactions made under the account 45 into an int variable. Then showing the number as output.

Problems: Had to spend more time connecting the database with JAVA code than doing the task itself.

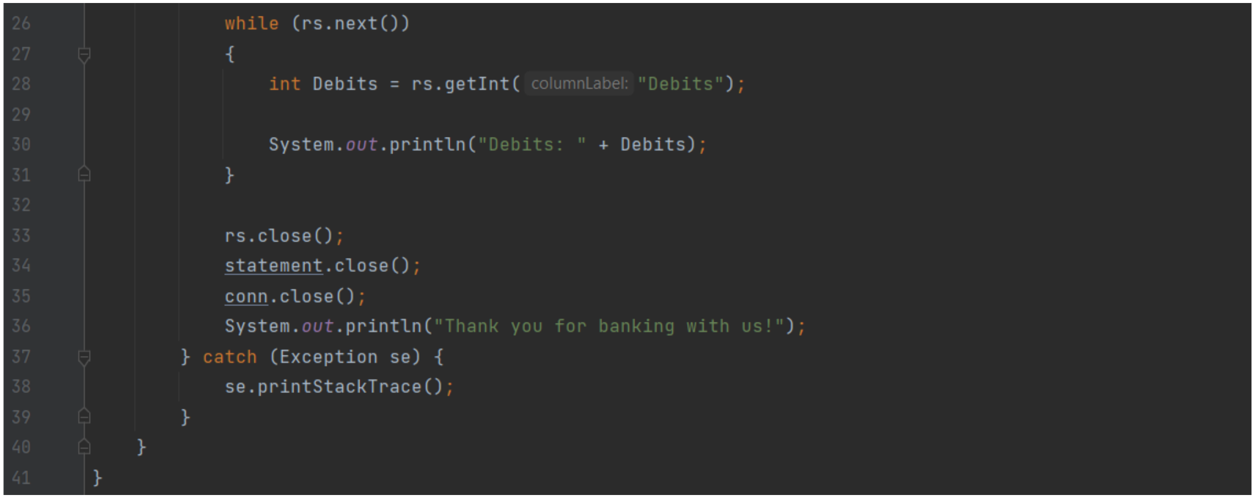
## Task 2

2. Count the number of debits.

The Solution code:



```java
import java.sql.*;

public class Task2
{
    static final String JDBC_DRIVER = "oracle.jdbc.driver.OracleDriver";
    static final String DB_URL = "jdbc:oracle:thin:@localhost:1521:xe";
    static final String USER = "System";
    static final String PASS = "DemonSlayer123";

    public static void main(String args[]) {
        Connection conn = null;
        Statement statement = null;
        try {
            Class.forName(JDBC_DRIVER);
            System.out.println("Connecting to database");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);

            System.out.println("Creating statement");
            statement = conn.createStatement();

            String sql = "SELECT COUNT(T_ID) Debits FROM TRANSACTIONS WHERE TRANSACTIONS.TYPE = '1'";
            System.out.println("Executing the query: " + sql);

            ResultSet rs = statement.executeQuery(sql);

```



```java
            while (rs.next())
            {
                int Debits = rs.getInt( columnLabel: "Debits");

                System.out.println("Debits: " + Debits);
            }

            rs.close();
            statement.close();
            conn.close();
            System.out.println("Thank you for banking with us!");
        } catch (Exception se) {
            se.printStackTrace();
        }
    }
}
```

Explanation:

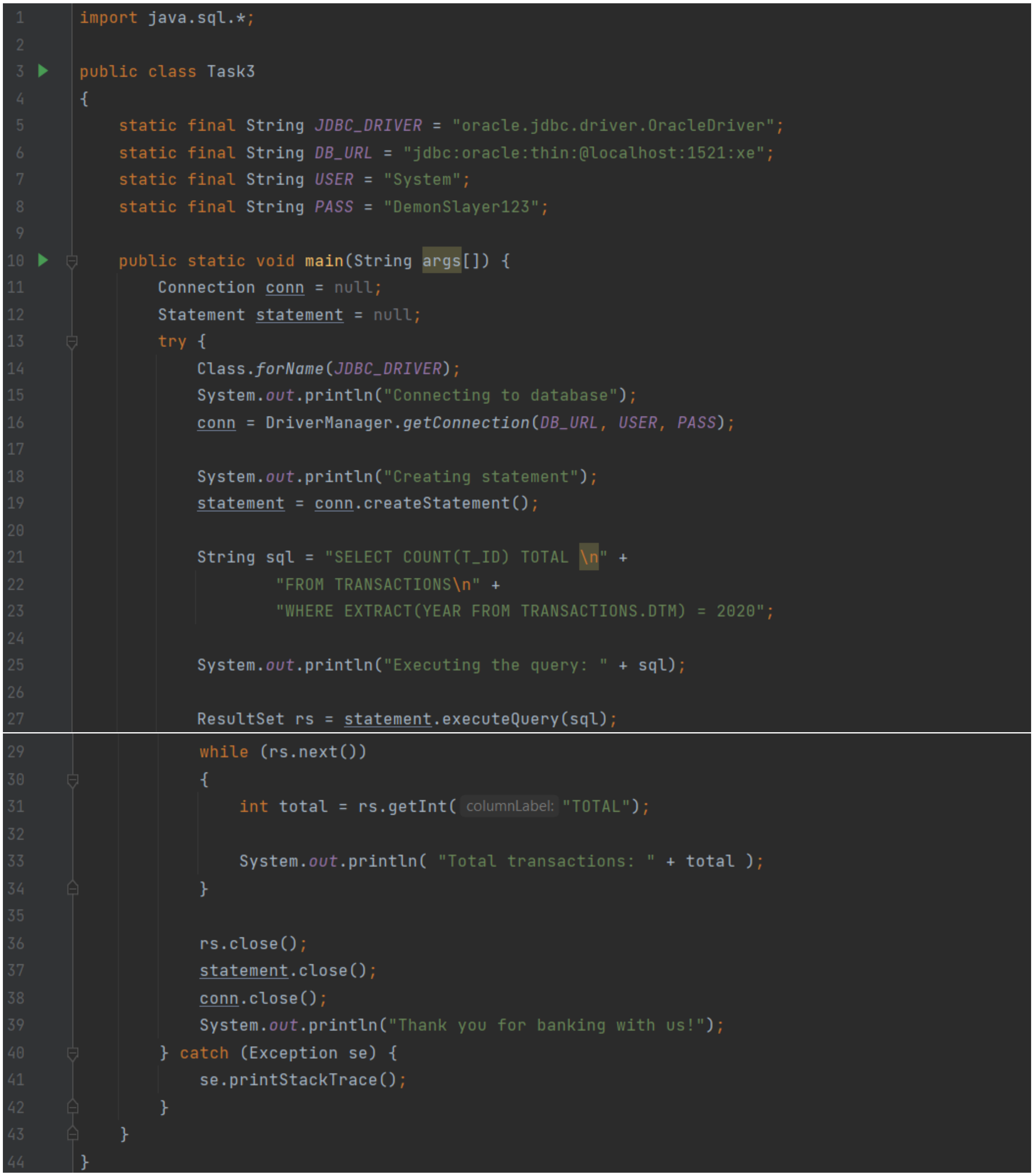
Line 21: The SQL query has been written down

Line 28: Storing the number of debits into an int variable. Then showing the number as output.

Problems: Didn’t face any problems.

# Task 3

The Solution code:



```java
import java.sql.*;

public class Task3
{
    static final String JDBC_DRIVER = "oracle.jdbc.driver.OracleDriver";
    static final String DB_URL = "jdbc:oracle:thin:@localhost:1521:xe";
    static final String USER = "System";
    static final String PASS = "DemonSlayer123";

    public static void main(String args[]) {
        Connection conn = null;
        Statement statement = null;
        try {
            Class.forName(JDBC_DRIVER);
            System.out.println("Connecting to database");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);

            System.out.println("Creating statement");
            statement = conn.createStatement();

            String sql = "SELECT COUNT(T_ID) TOTAL \n" +
                    "FROM TRANSACTIONS\n" +
                    "WHERE EXTRACT(YEAR FROM TRANSACTIONS.DTM) = 2020";

            System.out.println("Executing the query: " + sql);

            ResultSet rs = statement.executeQuery(sql);
            while (rs.next())
            {
                int total = rs.getInt( columnLabel: "TOTAL");

                System.out.println( "Total transactions: " + total );
            }

            rs.close();
            statement.close();
            conn.close();
            System.out.println("Thank you for banking with us!");
        } catch (Exception se) {
            se.printStackTrace();
        }
    }
}
```

Explanation:

Line 21 to 23: The SQL query has been written down

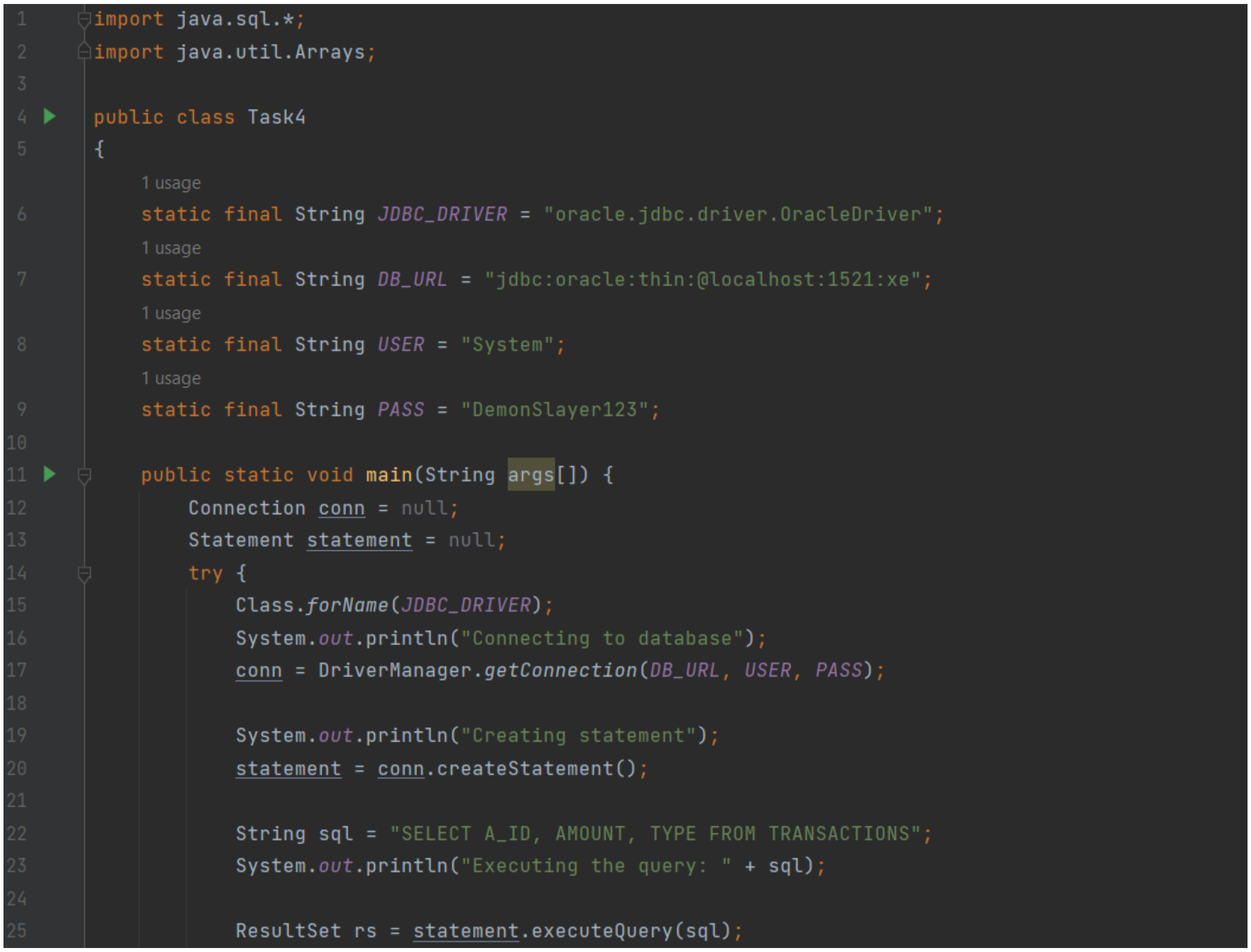
Line 23: The EXTRACT() function extracts a part from a given date

Line 28: Storing the number of Transactions that happened in 2020 into an int variable. Then showing the number as output.

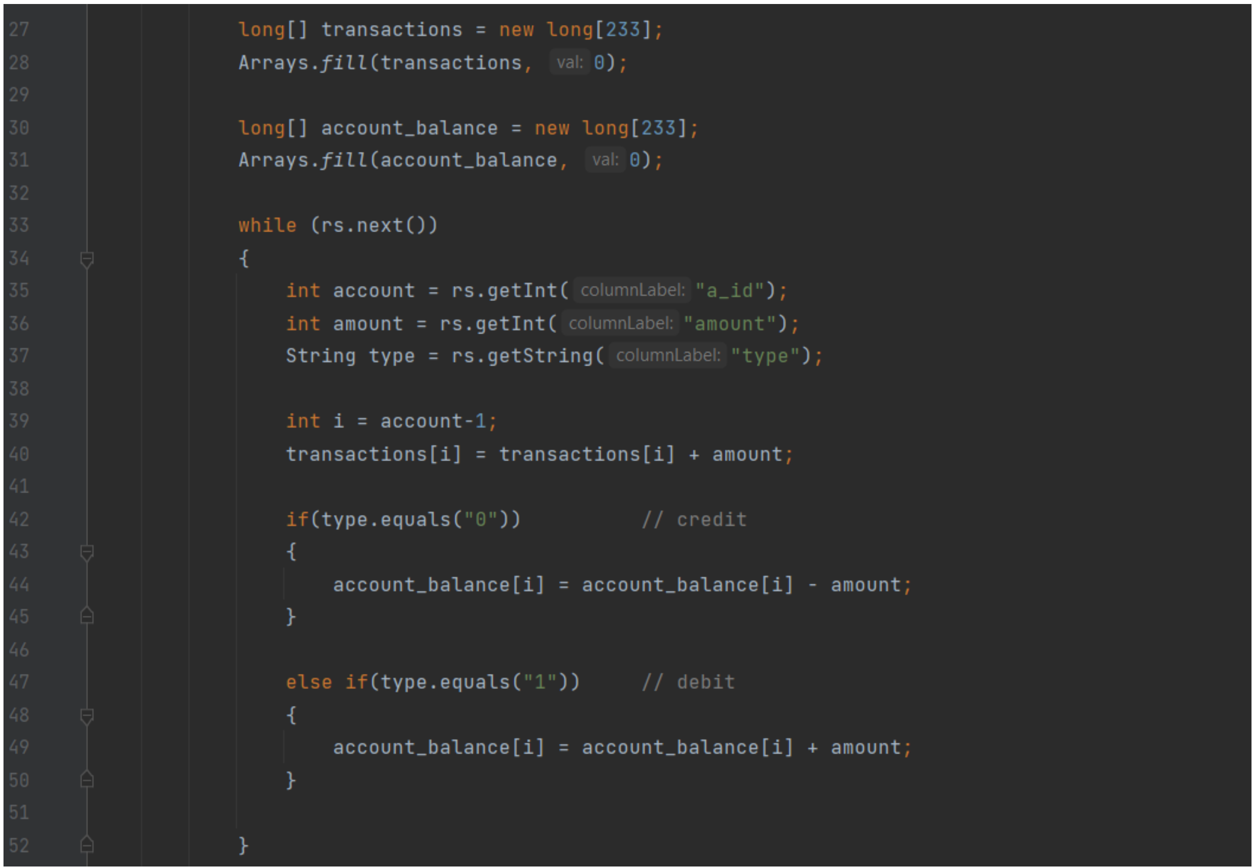
Problems: Didn't know how to extract date from a certain DATE data type.

## Task 4

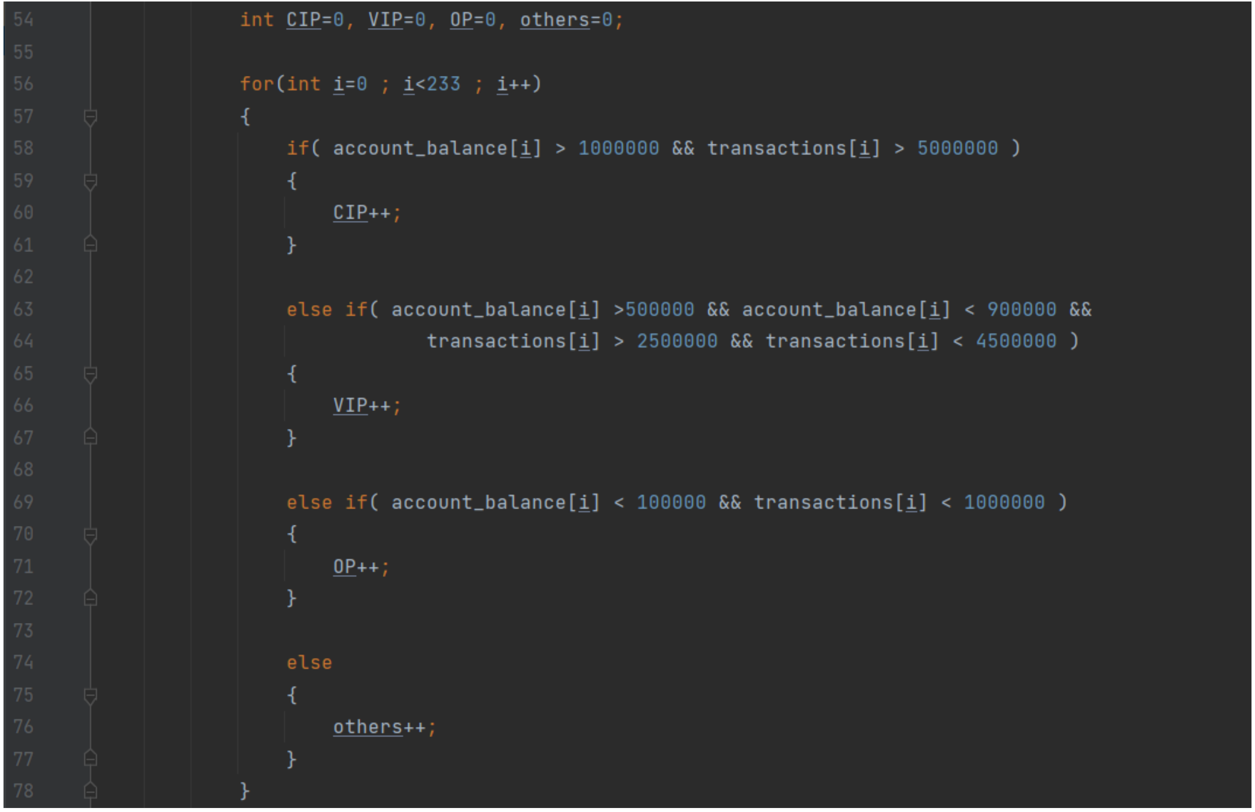
The Solution code:



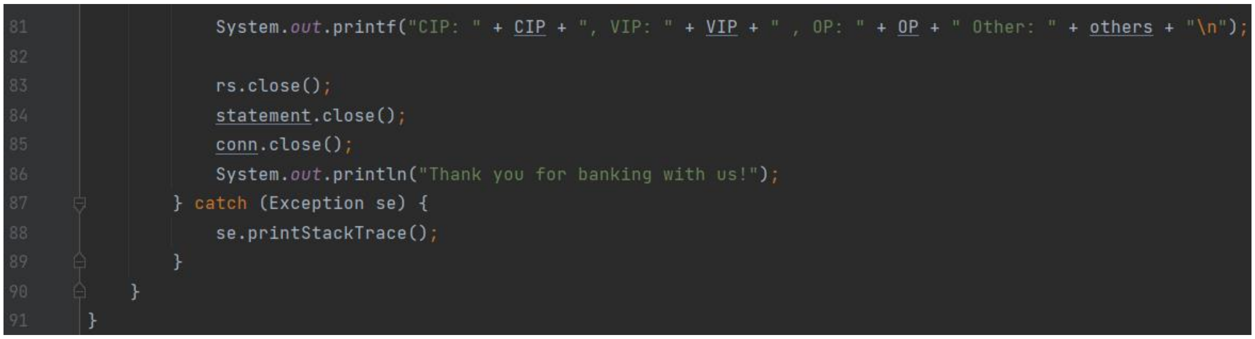
```
1   import java.sql.*;
2   import java.util.Arrays;
3
4   public class Task4
5   {
        1 usage
6       static final String JDBC_DRIVER = "oracle.jdbc.driver.OracleDriver";
        1 usage
7       static final String DB_URL = "jdbc:oracle:thin:@localhost:1521:xe";
        1 usage
8       static final String USER = "System";
        1 usage
9       static final String PASS = "DemonSlayer123";
10
11      public static void main(String args[]) {
12          Connection conn = null;
13          Statement statement = null;
14          try {
15              Class.forName(JDBC_DRIVER);
16              System.out.println("Connecting to database");
17              conn = DriverManager.getConnection(DB_URL, USER, PASS);
18
19              System.out.println("Creating statement");
20              statement = conn.createStatement();
21
22              String sql = "SELECT A_ID, AMOUNT, TYPE FROM TRANSACTIONS";
23              System.out.println("Executing the query: " + sql);
24
25              ResultSet rs = statement.executeQuery(sql);
```



```java
        long[] transactions = new long[233];
        Arrays.fill(transactions,  val: 0);

        long[] account_balance = new long[233];
        Arrays.fill(account_balance,  val: 0);

        while (rs.next())
        {
            int account = rs.getInt( columnLabel: "a_id");
            int amount = rs.getInt( columnLabel: "amount");
            String type = rs.getString( columnLabel: "type");

            int i = account-1;
            transactions[i] = transactions[i] + amount;

            if(type.equals("0"))          // credit
            {
                account_balance[i] = account_balance[i] - amount;
            }

            else if(type.equals("1"))     // debit
            {
                account_balance[i] = account_balance[i] + amount;
            }

        }
```



```java
        int CIP=0, VIP=0, OP=0, others=0;

        for(int i=0 ; i<233 ; i++)
        {
            if( account_balance[i] > 1000000 && transactions[i] > 5000000 )
            {
                CIP++;
            }

            else if( account_balance[i] >500000 && account_balance[i] < 900000 &&
                        transactions[i] > 2500000 && transactions[i] < 4500000 )
            {
                VIP++;
            }

            else if( account_balance[i] < 100000 && transactions[i] < 1000000 )
            {
                OP++;
            }

            else
            {
                others++;
            }
        }
```



```
81              System.out.printf("CIP: " + CIP + ", VIP: " + VIP + " , OP: " + OP + " Other: " + others + "\n");
82
83              rs.close();
84              statement.close();
85              conn.close();
86              System.out.println("Thank you for banking with us!");
87          } catch (Exception se) {
88              se.printStackTrace();
89          }
90      }
91  }
```

Explanation:

Line 22: The SQL query selects all account id, amount and type of all transactions

Line 27: Created a static array to store the transaction of all accounts. There were 233 accounts. That's why the size of the array is 233

Line 30: Created a static array to store the balance of all accounts. There were 233 accounts. That's why the size of the array is 233

Line 28 and 31: Filling the arrays with initial value of zero

Line 39: Initializing the variable "i" as "account-1". Since array indexing starts from 0

Line 40: Every time an iteration happens the amount transacted to the account "i" is stored in the $i^{th}$ index of the array.

Line 42 to 50: Based on the type of transaction, the balance of the $i^{th}$ account is adjusted

Line 56 to 78: The loop finds out the number of CIP(s), VIP(s), OP(s) and other accounts that are present in the bank according to the scenario

Line 81: Showing the number of CIP(s), VIP(s), OP(s) and other accounts as output

Problems:

The concept is pretty straightforward. It just took me quite a bit of time to find a proper solution and then implement it in code.