Islamic University of Technology

Lab 08

CSE 4308 - DBMS Lab

Submitted To :

Zannatun Naim Sristy
Lecturer, CSE Department
Islamic University of Technology

Submitted By :

Rhidwan Rashid
ID : 200042149
Prog. : SWE
Dept. : CSE

Task 1:

Analysis: This problem requires counting the total number of transactions conducted under account 45.

Working code:

```
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="1234";
    public static void main (String args[])
    {
        Connection conn=null;
        Statement stmt=null;
        try
        {
            Class.forName(JDBC_DRIVER);
            System.out.println("Connecting to
database");
            conn=DriverManager.getConnection(DB_URL,
USER, PASS);
            System.out.println("Creating statement");
            stmt=conn.createStatement();
            String sql;
            sql="SELECT COUNT(T_ID) TOTAL FROM ACCOUNT
A, TRANSACTIONS T WHERE A.A_ID = T.A_ID AND A.A_ID =
```

```
45";
            System.out.println("Executing the query: " +
sql);
            ResultSet rs=stmt.executeQuery(sql);
            while(rs.next())
            {
                System.out.println(rs.getInt("TOTAL"));
            }

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

Explanation:
Connection to oracle database is required for every problem. The required SQL query is written in a string variable first. Then the query is executed and stored in a variable type of ResultSet. Then a loop is conducted until it reaches end of ResultSet. As the query gives only one row, so, the result is printed inside the loop.

Problems:
Connecting to the database was the hardest part. Understanding ResultSet and how the code inside loop works was quite difficult too.

Task 2:

Analysis: This problem requires counting number of debits.

Working code:

```
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="1234";
    public static void main (String args[])
    {
        Connection conn=null;
        Statement stmt=null;
        try
        {
            Class.forName(JDBC_DRIVER);
            System.out.println("Connecting to
database");
            conn=DriverManager.getConnection(DB_URL,
USER, PASS);
            System.out.println("Creating statement");
            stmt=conn.createStatement();
```

```
            String sql;
            sql="SELECT COUNT(T_ID) DEBITS\n" +
                    "FROM TRANSACTIONS T\n" +
                    "WHERE T.TYPE = '1'";
            System.out.println("Executing the query: " +
sql);
            ResultSet rs=stmt.executeQuery(sql);
            while(rs.next())
            {
                System.out.println(rs.getInt("DEBITS"));
            }

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

Explanation: Same as task 1. In query, transaction ids are counted when transaction type is 1 (debit). This query also returns only row. So, the result is printed inside loop.

Problems: Not much problems occurred as the task is same as task 1. Only query is different.

Task 3:

Analysis: This task requires finding transactions that occurred in year 2020.

Working code:

```
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="1234";
    public static void main (String args[])
    {
        Connection conn=null;
        Statement stmt=null;
        try
        {
            Class.forName(JDBC_DRIVER);
            System.out.println("Connecting to
database");
            conn=DriverManager.getConnection(DB_URL,
USER, PASS);
            System.out.println("Creating statement");
            stmt=conn.createStatement();
            String sql;
```

```
            sql="SELECT T_ID, A_ID, AMOUNT, TYPE\n" +
                    "FROM TRANSACTIONS T\n" +
                    "WHERE EXTRACT(YEAR FROM T.DTM) =
'2020'";
            System.out.println("Executing the query: " +
sql);
            ResultSet rs=stmt.executeQuery(sql);
            while(rs.next())
            {
                String tid = rs.getString("T_ID");
                String aid = rs.getString("A_ID");
                String amount = rs.getString("AMOUNT");
                String type = rs.getString("TYPE");
                System.out.println(tid+ "," + aid + ","
+ amount + "," + type);
            }

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

```
    }
}
```

Explanation: In the SQL query, transaction id, account id, amount and transaction types are selected from transaction table. In where statement "EXTRACT" is used to extract the year from date time of transaction and it is compared with 2020. The result in stored in a ResultSet variable. In this case, the query returns multiple rows. So, the Resultset variable is iterated untill the end and all the selected attributes are printed in comma separated way.

Problems: No problems have been faced here.

Task 4:

Analysis: This problem requires finding number of CIP, VIP, OP and others who don't fall under any category.

Working code:

```
import java.sql.*;
import java.util.HashMap;
import java.util.Locale;

public class Task4
{
    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="1234";
    public static void main (String args[])
```

```
    {
        Connection conn=null;
        Statement stmt=null;
        try
        {
            Class.forName(JDBC_DRIVER);
            System.out.println("Connecting to
database");
            conn=DriverManager.getConnection(DB_URL,
USER, PASS);
            System.out.println("Creating statement");
            stmt=conn.createStatement();
            String sql;
            sql="SELECT A_ID, AMOUNT, TYPE FROM
TRANSACTIONS";
            System.out.println("Executing the query: " +
sql);
            ResultSet rs=stmt.executeQuery(sql);

            int CIP = 0;
            int VIP = 0;
            int OP = 0;
            int Others = 0;

            HashMap<Integer, Long> transactions = new
HashMap<>();
            HashMap<Integer, Long> accBalance = new
HashMap<>();
            while(rs.next())
            {
                int id = rs.getInt("A_ID");
                Long amount = rs.getLong("AMOUNT");
```

```
                String type = rs.getString("TYPE");

                if(transactions.get(id) == null)
                    transactions.put(id, amount);
                else
                    transactions.replace(id,
transactions.get(id) + amount);

                if(accBalance.get(id) == null)
                    accBalance.put(id, amount);
                else
                {
                    if(type.equals("0"))
                        accBalance.replace(id,
accBalance.get(id) + amount);
                    else
                        accBalance.replace(id,
accBalance.get(id) - amount);
                }
            }
            for(int i=1; i<98; i++)
            {
                if(accBalance.get(i) > 1000000 &&
transactions.get(i) > 5000000)
                    CIP++;
                else if(accBalance.get(i) > 500000 &&
accBalance.get(i) < 900000 && transactions.get(i) >
2500000 && transactions.get(i) < 4500000)
                    VIP++;
                else if(accBalance.get(i) < 100000 &&
transactions.get(i) < 1000000)
                    OP++;
```

```
                else
                    Others++;
            }
            System.out.println("CIP: " + CIP);
            System.out.println("VIP: " + VIP);
            System.out.println("OP: " + OP);
            System.out.println("Others: " + Others);
            rs.close();
            stmt.close();
            conn.close();
            System.out.println("Thank you for banking
with us!");
        }
        catch(SQLException se)
        {
            se.printStackTrace();
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

Explanation: The solution of this task is more dependent on java programming than SQL query. A simple query which selects account id, amount and type is conducted and the result is stored in the ResultSet variable. This query returns multiple rows. Two java hashmaps is declared here. One named transaction which has account id as key and amount as value. And the other names account balance which keep tracks of balance in account with account id as key and balance as value. And four counters named CIP, VIP, OP and Others are

declared also. Hashmap keys are unique which is used in this prospect. Inside loop, we get the account id and put it in the transaction map, as well as the new balance. Then based on type of transaction updated account balance is stored in accBalance hashmap. After the end of the loop, all account ids are iterated using for loop again. Inside this loop, hashmaps for that particular id is accessed using the key and conditions for CIP, VIP, OP, and Others and the counters are incremented as conditions are met. At the end the value of the counters are printed.

Problems: It took some time for me to understand that, this problem can't be solved using sql alone. After googling, the solution using java hashmaps came to my mind. Then, after some brainstorming and some trial and errors, I came to this solution.