
| Course Number and Name:<br>CSE 4308<br>Database Management Systems Lab | |
|---|---|
| **Student Name:**<br>Nafisa Maliyat | **Student ID:**<br>200042133 |
| **Report Submission Date:**<br>28 October, 2022 | **Name of Lab Instructor:**<br>Md. Bakhtiar Hasan, Lecturer, CSE<br>Zannatun Naim Sristy, Lecturer, CSE |

# Lab 8: Java database connectivity

## Overview:

This lab provided us a manual for connecting to database using programming language (Java, in this case). Different SQL queries had to be carried out in Java to get the information stated in the problem statement.

On the next pages, I have mentioned the following :

- the problem statement
- analysis of the problem,
- java code written to solve the problem,
- problems faced (if any) during solution of the tasks,
- the results.

General format of the Java code:

```
import java.sql.*;

public class Task01 {
   static final String JDBC_DRIVER = "oracle.jdbc.driver.OracleDriver";
   static final String DB_URL= "jdbc:oracle:thin:@localhost:1521:xe";
   static final String USER="user";
   static final String PASS="password";

   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 QUERIES AND OUTPUT PRINTING DONE IN THIS SEGMENT
        //THIS PART WILL BE GIVEN IN THE CODE SECTOIN OF EACH TASK
         rs.close();
         stmt.close();
         conn.close();
         System.out.println("Query of task n completed successfully!");
      } catch (SQLException se) {
         se.printStackTrace();
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

Task 1:

### Problem Statement:

Count the total number of transactions conducted under account 45.

### Analysis of the problem:

This can be solved by grouping account ID and adding an condition to choose account with account number 45. Then a count is taken of all the transaction IDs that meet the condition.

### Any problems faced and how it was solved:

There were no problems faced since the query was straightforward and simple.

### Code:

```
sql = "SELECT COUNT(t_id) AS T_COUNT FROM TRANSACTIONS " +
      "GROUP BY A_ID HAVING A_ID = 45";
System.out.println("Executing the query: " + sql);
ResultSet rs = stmt.executeQuery(sql);
while (rs.next()) {
   //get date
   int count = rs.getInt("T_COUNT");

   //printing
   System.out.print("Total number of "
         + count );
   if(count==1)
      System.out.print(" transaction");
   else
      System.out.print(" transactions");
   System.out.print(" were conducted under account 45.\n");
}
```

### Results:

```
Connecting to database
Creating statement
Executing the query: SELECT COUNT(t_id) AS T_COUNT FROM TRANSACTIONS
GROUP BY A_ID HAVING A_ID = 45
Total number of 1 transaction were conducted under account 45.
Query of task 1 completed successfully!
```

## Task 2:

### Problem Statement:

Count the number of debits.

### Analysis of the problem:

This requires selecting the number of transaction IDs that fulfill the condition of having the attribute TYPE equals to 1.

### Any problems faced and how it was solved:

There were no problems faced since the query was straightforward and simple.

### Code:

```
sql="SELECT COUNT(t_id) AS COUNT_DEBIT FROM TRANSACTIONS " +
      "WHERE TYPE = 1";
System.out.println("Executing the query: " + sql);
ResultSet rs = stmt.executeQuery(sql);
while(rs.next())
{
   //get date
   int count =rs.getInt("COUNT_DEBIT");

   //printing
   System.out.print("Total number of debits is " + count+ "\n");
}
```

### Results:

```
Connecting to database
Creating statement
Executing the query: SELECT COUNT(t_id) AS COUNT_DEBIT FROM TRANSACTIONS
WHERE TYPE = 1
Total number of debits is 816
Query of task 2 completed successfully!
```

Task 3:

### Problem Statement:

List the transactions that occurred in the year 2020.

### Analysis of the problem:

This requires selecting those transactions whose date of transaction was in 2020.

### Any problems faced and how it was solved:

There was confusion regarding the syntax for getting the year from a date but this was solved using internet resources.

### Code:

```
sql="SELECT T_ID, DTM, A_ID, AMOUNT, TYPE FROM TRANSACTIONS " +
      "WHERE EXTRACT(YEAR FROM DTM) = 2020";
System.out.println("Executing the query: " + sql);
ResultSet rs = stmt.executeQuery(sql);
while(rs.next())
{
   //get date
   int t_id =rs.getInt("T_ID");
   Date dtm =rs.getDate("DTM");
   int a_id = rs.getInt("A_ID");
   int amount = rs.getInt("AMOUNT");
   String type = rs.getString("TYPE");

   //printing
   System.out.print("Transaction ID " + t_id +
         " occurred at " + dtm + " where " + amount
         + " taka has been");
   if(type.charAt(0)=='0')
      System.out.print(" deposited to");
   else
      System.out.print(" taken out from");
   System.out.println(" account " + a_id);
}
```

Results:

```
Connecting to database
Creating statement
Executing the query: SELECT T_ID, DTM, A_ID, AMOUNT, TYPE FROM
TRANSACTIONS WHERE EXTRACT(YEAR FROM DTM) = 2020
Transaction ID 7 occurred at 2020-03-26 where 1011500 taka has been taken
out from account 2
Transaction ID 16 occurred at 2020-04-24 where 1603700 taka has been
deposited to account 4
Transaction ID 25 occurred at 2020-12-05 where 1487200 taka has been
deposited to account 5
Transaction ID 27 occurred at 2020-06-06 where 1119050 taka has been
deposited to account 6
Transaction ID 44 occurred at 2020-12-04 where 1638200 taka has been
deposited to account 8
Transaction ID 76 occurred at 2020-10-04 where 519050 taka has been taken
out from account 15
Transaction ID 85 occurred at 2020-09-05 where 1299800 taka has been
taken out from account 17
Transaction ID 138 occurred at 2020-05-06 where 592450 taka has been
deposited to account 25
Transaction ID 154 occurred at 2020-05-16 where 1478450 taka has been
taken out from account 29
Transaction ID 189 occurred at 2020-06-28 where 404650 taka has been
deposited to account 36
Transaction ID 210 occurred at 2020-05-25 where 449350 taka has been
deposited to account 40
Transaction ID 231 occurred at 2020-01-27 where 1484400 taka has been
taken out from account 43
Transaction ID 235 occurred at 2020-11-14 where 987450 taka has been
taken out from account 44
Transaction ID 267 occurred at 2020-09-27 where 493200 taka has been
deposited to account 51
Transaction ID 268 occurred at 2020-11-12 where 833300 taka has been
deposited to account 51
Transaction ID 270 occurred at 2020-11-26 where 1285800 taka has been
taken out from account 51
Transaction ID 284 occurred at 2020-03-22 where 1058500 taka has been
deposited to account 53
Transaction ID 326 occurred at 2020-06-23 where 1462700 taka has been
deposited to account 60
```

```
Transaction ID 342 occurred at 2020-01-27 where 592700 taka has been
taken out from account 64
Transaction ID 377 occurred at 2020-10-22 where 1557250 taka has been
taken out from account 68
Transaction ID 395 occurred at 2020-01-19 where 73500 taka has been
deposited to account 70
Transaction ID 419 occurred at 2020-01-23 where 933500 taka has been
deposited to account 75
Transaction ID 480 occurred at 2020-11-11 where 587850 taka has been
deposited to account 89
Transaction ID 482 occurred at 2020-02-12 where 1062200 taka has been
taken out from account 89
Transaction ID 520 occurred at 2020-09-28 where 307600 taka has been
taken out from account 95
Transaction ID 541 occurred at 2020-08-16 where 208200 taka has been
taken out from account 98
Transaction ID 604 occurred at 2020-02-05 where 1345150 taka has been
taken out from account 109
Transaction ID 613 occurred at 2020-05-26 where 270700 taka has been
taken out from account 110
Transaction ID 618 occurred at 2020-10-17 where 1151300 taka has been
taken out from account 111
Transaction ID 623 occurred at 2020-06-06 where 1262350 taka has been
deposited to account 112
Transaction ID 625 occurred at 2020-08-22 where 724200 taka has been
deposited to account 112
Transaction ID 637 occurred at 2020-02-10 where 866800 taka has been
taken out from account 114
Transaction ID 652 occurred at 2020-04-12 where 1075200 taka has been
taken out from account 117
Transaction ID 710 occurred at 2020-06-28 where 1104700 taka has been
taken out from account 126
Transaction ID 736 occurred at 2020-06-15 where 1552300 taka has been
taken out from account 130
Transaction ID 767 occurred at 2020-10-21 where 362200 taka has been
deposited to account 138
Transaction ID 793 occurred at 2020-01-10 where 822100 taka has been
deposited to account 143
Transaction ID 804 occurred at 2020-02-21 where 160150 taka has been
deposited to account 145
Transaction ID 857 occurred at 2020-09-05 where 1199250 taka has been
```

```
deposited to account 153
Transaction ID 894 occurred at 2020-03-12 where 328150 taka has been
taken out from account 160
Transaction ID 935 occurred at 2020-11-12 where 263400 taka has been
deposited to account 169
Transaction ID 953 occurred at 2020-06-27 where 165850 taka has been
taken out from account 171
Transaction ID 957 occurred at 2020-10-01 where 94150 taka has been taken
out from account 172
Transaction ID 1038 occurred at 2020-09-06 where 853000 taka has been
taken out from account 188
Transaction ID 1105 occurred at 2020-02-24 where 1331600 taka has been
deposited to account 199
Transaction ID 1118 occurred at 2020-11-24 where 1335250 taka has been
deposited to account 200
Transaction ID 1122 occurred at 2020-06-28 where 1177550 taka has been
deposited to account 202
Transaction ID 1124 occurred at 2020-05-18 where 1103850 taka has been
deposited to account 202
Transaction ID 1144 occurred at 2020-10-15 where 1220500 taka has been
taken out from account 205
Transaction ID 1216 occurred at 2020-03-06 where 1266250 taka has been
taken out from account 217
Transaction ID 1217 occurred at 2020-05-20 where 122050 taka has been
taken out from account 217
Transaction ID 1246 occurred at 2020-03-15 where 252900 taka has been
taken out from account 222
Transaction ID 1272 occurred at 2020-07-25 where 515900 taka has been
taken out from account 226
Transaction ID 1277 occurred at 2020-05-22 where 1498100 taka has been
deposited to account 226
Transaction ID 1353 occurred at 2020-06-28 where 1003350 taka has been
deposited to account 237
Transaction ID 1383 occurred at 2020-08-22 where 353400 taka has been
deposited to account 241
Transaction ID 1396 occurred at 2020-08-20 where 1442750 taka has been
taken out from account 244
Transaction ID 1398 occurred at 2020-12-19 where 1132250 taka has been
deposited to account 244
Transaction ID 1418 occurred at 2020-01-20 where 1125400 taka has been
deposited to account 247
```

```
Transaction ID 1457 occurred at 2020-11-03 where 66350 taka has been
taken out from account 253
Transaction ID 1496 occurred at 2020-08-13 where 1055850 taka has been
taken out from account 259
Transaction ID 1507 occurred at 2020-10-31 where 336800 taka has been
deposited to account 260
Transaction ID 1549 occurred at 2020-01-11 where 355850 taka has been
deposited to account 266
Transaction ID 1550 occurred at 2020-06-22 where 1052100 taka has been
taken out from account 266
Transaction ID 1566 occurred at 2020-09-12 where 1199200 taka has been
deposited to account 269
Transaction ID 1569 occurred at 2020-06-05 where 1545650 taka has been
deposited to account 270
Transaction ID 1596 occurred at 2020-07-08 where 642550 taka has been
deposited to account 274
Transaction ID 1603 occurred at 2020-09-12 where 831800 taka has been
taken out from account 275
Transaction ID 1623 occurred at 2020-02-24 where 638900 taka has been
taken out from account 279
Transaction ID 1632 occurred at 2020-08-26 where 1169350 taka has been
deposited to account 280
Transaction ID 1673 occurred at 2020-05-26 where 570300 taka has been
deposited to account 288
Query of task 3 completed successfully!
```

## Task 4:

### Problem Statement:

Count the number of CIP, VIP, and OPs. Also show the number of people that do not fall in any of the categories.

### Analysis of the problem:

This has to be solved in steps. Since balance is not provided in the table details, extra calculations have to be performed.

Information required is the current balance of each account and the account number associated with the balance. A hashmap would be perfect for this since it can store account number and also its balance.

First a query is used to get each row of transaction and a loop is used to iterate through the rows. If the map already has the account ID, the existing entry will be updated. Similarly, if the map did not have the account ID, a new entry will be created. For the balance, the type is checked and the corresponding balance is deducted or increased accordingly. In this way, calculation of balance for each account ID is completed.

For checking the conditions, an arraylist is initiated where the IDs that will be already included in some other category is stored. This is so the IDs that are not of any other categories can be included in the last category.

A general procedure can be followed for each category. A count variable is taken for each category. For all categories except the first category, first the ID is checked against the ararylist to see if it has already been included in any previous category.

Then, a SQL statement is used to check the condition for the sum of transactions, since this information is stored in database. Next, the ID is searched for in the map and the condition for account balance is checked. If both these conditions are fulfilled, the count is incremented and the ID is added to the arraylist keeping track of traversed IDs.

For the last category, the IDs that are not in the traversed ID arraylist is counted and added to the last category. After getting count of each category, the count is printed.

### Any problems faced and how it was solved:

The solution required storing the balance somewhere for checking condition. Its implementation was a bit confusing. This was solved by checking which data type in Java can store pair values.

Code:

```
//calculate balance
HashMap<Integer, Integer> account_balance = new HashMap();
sql = "SELECT A_ID, AMOUNT, TYPE FROM TRANSACTIONS";

System.out.println("Executing the query: " + sql);
ResultSet rs = stmt.executeQuery(sql);

while(rs.next()) {
    int a_id = rs.getInt("A_ID");
    int amount = rs.getInt("AMOUNT");
    String type = rs.getString("TYPE");

    //if map already has a_id
    if(account_balance.containsKey(a_id)){
        if(type.charAt(0)=='0'){
            account_balance.put(a_id, account_balance.get(a_id) - amount);
        }
        else{
            account_balance.put(a_id, account_balance.get(a_id) + amount);
        }
    }

    //if it does not exist
    else{
        if(type.charAt(0)=='0'){
            account_balance.put(a_id, - amount);
        }
        else{
            account_balance.put(a_id, amount);
        }
    }
}


ArrayList<Integer> id_traversed = new ArrayList<Integer>();

// count_cip
sql = "SELECT A_ID FROM TRANSACTIONS GROUP BY A_ID HAVING SUM(AMOUNT)>5000000" ;
System.out.println("Executing the query: " + sql);
```

```
rs = stmt.executeQuery(sql);

int count_cip = 0;
while(rs.next())
{
    int a_id = rs.getInt("A_ID");
    if(account_balance.get(a_id) > 1000000){
        count_cip++;
        id_traversed.add(a_id);
    }
}

//printing
System.out.print("Number of CIP accounts: " + count_cip + "\n");



//count_vip
sql = "SELECT A_ID FROM TRANSACTIONS GROUP BY A_ID HAVING SUM(AMOUNT)> 2500000 AND
SUM(AMOUNT)<4500000" ;
System.out.println("Executing the query: " + sql);
rs = stmt.executeQuery(sql);
int count_vip = 0;
while(rs.next())
{
    //count_vip
    int a_id =rs.getInt("a_id");

    if(account_balance.get(a_id) > 500000 && account_balance.get(a_id) <900000
&& !id_traversed.contains(a_id)){
        count_vip++;
        id_traversed.add(a_id);
    }
}

//printing
System.out.print("Number of VIP accounts: " + count_vip + "\n");
```

```
//count_op
sql = "SELECT A_ID FROM TRANSACTIONS GROUP BY A_ID HAVING SUM(AMOUNT)<1000000";
System.out.println("Executing the query: " + sql);
rs = stmt.executeQuery(sql);

int count_op = 0;
while(rs.next())
{
    //count_op
    int a_id = rs.getInt("A_ID");

    if(account_balance.get(a_id) < 100000 && !id_traversed.contains(a_id)) {
        id_traversed.add(a_id);
        count_op++;
    }
}

//printing
System.out.print("Number of OP accounts: " + count_op + "\n");



//count_others
int count_other = 0;
for (HashMap.Entry<Integer,Integer> entry : account_balance.entrySet()){
    if(!id_traversed.contains(entry.getKey())){
        count_other++;
    }
}

//printing
System.out.print("Number of other accounts: " + count_other + "\n");
```

Results:

```
Connecting to database
Creating statement
Executing the query: SELECT A_ID, AMOUNT, TYPE FROM TRANSACTIONS
Executing the query: SELECT A_ID FROM TRANSACTIONS GROUP BY A_ID HAVING
SUM(AMOUNT)>5000000
Number of CIP accounts: 50
Executing the query: SELECT A_ID FROM TRANSACTIONS GROUP BY A_ID HAVING
SUM(AMOUNT)> 2500000 AND SUM(AMOUNT)<4500000
Number of VIP accounts: 7
Executing the query: SELECT A_ID FROM TRANSACTIONS GROUP BY A_ID HAVING
SUM(AMOUNT)<1000000
Number of OP accounts: 12
Number of other accounts: 222
Query of task 4 completed successfully!
```