Trabalho 07 - BD2

Guilherme Rimoldi Kameoka
Júlio César de Jesus Batista

1. Retornando os nomes de todos os clientes do banco, com suas respectivas somas de depósitos e empréstimos, caso existam.

Properties SQL Statistics Dependencies Dependents Processes IB2/guilhermekameoka@IB2*

IB2/guilhermekameoka@IB2

No limit

Query Query History

```sql
SELECT
    nome_cliente,
    nome_agencia,
    numero_conta,
    COALESCE(SUM(saldo_deposito), 0) AS total_depositos,
    COALESCE(SUM(valor_emprestimo), 0) AS total_emprestimos
FROM (
    SELECT nome_cliente, nome_agencia, numero_conta, saldo_deposito, 0 AS valor_emprestimo
    FROM deposito
    UNION ALL
    SELECT nome_cliente, nome_agencia, numero_conta, 0 AS saldo_deposito, valor_emprestimo
    FROM emprestimo
) AS aux_query
GROUP BY nome_cliente, nome_agencia, numero_conta
ORDER BY nome_cliente;
```

Data Output Messages Notifications

	nome_cliente character varying (80)	nome_agencia character varying (50)	numero_conta integer	total_depositos double precision	total_emprestimos double precision
1	Adilson de Oliveira	Glória	61521	4832.31	0
2	Adilson de Oliveira	Glória	16910	1214.75	0
3	Alexandre Márcio de Souza	Gameleira	24067	4206.5599999999995	515.06
4	Andrade de Freitas	Central	72069	2114.8	0
5	Andrade de Freitas	Central	42436	4663.56	3147.63
6	André Cabral da Silva	Gameleira	93125	1752.23	771.98
7	André Cabral da Silva	Gameleira	42593	3036.67	0
8	André Cabral da Silva	Cidade Jardim	50073	6308.32	824.58
9	Bruno Miranda Pacheco de Castro	Pampulha	54505	3404.26	1469.77
10	Bruno Miranda Pacheco de Castro	UFU	76313	2944.82	684.12
11	Bruno Miranda Pacheco de Castro	Gameleira	77069	3653.51	761.44
12	Bruno Miranda Pacheco de Castro	UFU	54194	2316.61	1482.15
13	Bruno Tadeu Pita	Gameleira	39864	4046.54	0
14	Bruno Tadeu Pita	Glória	77563	614.83	1750.85
15	Bruno Tadeu Pita	Gameleira	21707	7062.639999999999	1955.46

Total rows: 86 of 86 Query complete 00:00:00.213

2. **Compilando e executando o MyQueries:**

```
guilhermekameoka@192 JDBCTutorial % ./comp MyQueries properties/postgres-propert
ies.xml
Set the following properties:
dbms: postgresql
driver: org.postgresql.Driver
dbName: IB2
userName: guilhermekameoka
serverName: localhost
portNumber: 5432
Connected to database
Contas da Instituição Bancária:
55282, Central
50073, Cidade Jardim
39864, Gameleira
17477, Glória
95617, Central
30273, Glória
66119, Pampulha
50906, Central
69118, Central
24035, Pampulha
70719, Glória
4190, Central
52607, Gameleira
```

3. **Modificando o programa para retornar os dados da consulta do item 1:**

```
guilhermekameoka@192 JDBCTutorial % ./comp MyQueries properties/postgres-properties.xml
Set the following properties:
dbms: postgresql
driver: org.postgresql.Driver
dbName: IB2
userName: guilhermekameoka
serverName: localhost
portNumber: 5432
Connected to database
Clientes e suas somas de depósitos e empréstimos:
Cliente: Adilson de Oliveira
Agência: Glória
Número da Conta: 61521
Total de Depósitos: 4832.31
Total de Empréstimos: 0.0

Cliente: Adilson de Oliveira
Agência: Glória
Número da Conta: 16910
Total de Depósitos: 1214.75
Total de Empréstimos: 0.0

Cliente: Alexandre Márcio de Souza
Agência: Gameleira
Número da Conta: 24067
Total de Depósitos: 4206.5599999999995
Total de Empréstimos: 515.06

Cliente: Andrade de Freitas
Agência: Central
Número da Conta: 72069
Total de Depósitos: 2114.8
Total de Empréstimos: 0.0

Cliente: Andrade de Freitas
Agência: Central
Número da Conta: 42436
Total de Depósitos: 4663.56
Total de Empréstimos: 3147.63

Cliente: André Cabral da Silva
Agência: Gameleira
Número da Conta: 93125
Total de Depósitos: 1752.23
Total de Empréstimos: 771.98

Cliente: André Cabral da Silva
Agência: Gameleira
Número da Conta: 42593
Total de Depósitos: 3036.67
Total de Empréstimos: 0.0

Cliente: André Cabral da Silva
Agência: Cidade Jardim
Número da Conta: 50073
Total de Depósitos: 6308.32
Total de Empréstimos: 824.58

Cliente: Bruno Miranda Pacheco de Castro
Agência: Pampulha
Número da Conta: 54505
Total de Depósitos: 3404.26
Total de Empréstimos: 1469.77

Cliente: Bruno Miranda Pacheco de Castro
Agência: UFU
Número da Conta: 76313
Total de Depósitos: 2944.82
Total de Empréstimos: 684.12

Cliente: Bruno Miranda Pacheco de Castro
Agência: Gameleira
Número da Conta: 77069
Total de Depósitos: 3653.51
Total de Empréstimos: 761.44
```

4. **Verificando a utilização dos metadados pelo SGBD através do MyQueries.java**

```
public static void cursorHoldabilitySupport(Connection conn)
        throws SQLException {
    DatabaseMetaData dbMetaData = conn.getMetaData();
    System.out.println("ResultSet.HOLD_CURSORS_OVER_COMMIT = " +
            ResultSet.HOLD_CURSORS_OVER_COMMIT);
    System.out.println("ResultSet.CLOSE_CURSORS_AT_COMMIT = " +
            ResultSet.CLOSE_CURSORS_AT_COMMIT);
    System.out.println("Default cursor holdability: " +
            dbMetaData.getResultSetHoldability());
    System.out.println("Supports HOLD_CURSORS_OVER_COMMIT? " +
            dbMetaData.supportsResultSetHoldability(
                    ResultSet.HOLD_CURSORS_OVER_COMMIT));
    System.out.println("Supports CLOSE_CURSORS_AT_COMMIT? " +
            dbMetaData.supportsResultSetHoldability(
                    ResultSet.CLOSE_CURSORS_AT_COMMIT));
}
```

```
guilhermekameoka@192 JDBCTutorial % ./comp MyQueries properties/postgres-propert
ies.xml
Set the following properties:
dbms: postgresql
driver: org.postgresql.Driver
dbName: IB2
userName: guilhermekameoka
serverName: localhost
portNumber: 5432
Connected to database
ResultSet.HOLD_CURSORS_OVER_COMMIT = 1
ResultSet.CLOSE_CURSORS_AT_COMMIT = 2
Default cursor holdability: 1
Supports HOLD_CURSORS_OVER_COMMIT? true
Supports CLOSE_CURSORS_AT_COMMIT? true
Releasing all open resources ...
guilhermekameoka@192 JDBCTutorial %
```

As constantes numeradas como 1 e 2 são usadas para definir as configurações de holdability de um ResultSet. Quando se utiliza o valor 1, conhecido como HOLD_CURSORS_OVER_COMMIT, o ResultSet permanece ativo mesmo após a transação ser concluída ou confirmada. Por outro lado, se o valor 2, denominado CLOSE_CURSORS_AT_COMMIT, for escolhido, o ResultSet será encerrado automaticamente ao término da transação. Normalmente, a configuração padrão é 1, que mantém o ResultSet ativo após a transação.

5. **Implementação do método supportsResultSetConcurrency e resultados da execução do programa**

```java
public static void supportsResultSetConcurrency(Connection conn) throws SQLException {
    DatabaseMetaData dbMetaData = conn.getMetaData();

    int[] resultSetTypes = { ResultSet.TYPE_FORWARD_ONLY, ResultSet.TYPE_SCROLL_INSENSITIVE,
            ResultSet.TYPE_SCROLL_SENSITIVE };
    int[] concurrencyTypes = { ResultSet.CONCUR_READ_ONLY, ResultSet.CONCUR_UPDATABLE };

    for (int resultSetType : resultSetTypes) {
        for (int concurrencyType : concurrencyTypes) {
            boolean supports = dbMetaData.supportsResultSetConcurrency(resultSetType, concurrencyType);
            System.out.println("ResultSet type: " + resultSetType + ", Concurrency type: " + concurrencyType
                    + ", Supports: " + supports);
        }
    }
}
```

```
guilhermekameoka@192 JDBCTutorial % ./comp MyQueries properties/postgres-propert
ies.xml
Set the following properties:
dbms: postgresql
driver: org.postgresql.Driver
dbName: IB2
userName: guilhermekameoka
serverName: localhost
portNumber: 5432
Connected to database
ResultSet type: 1003, Concurrency type: 1007, Supports: true
ResultSet type: 1003, Concurrency type: 1008, Supports: true
ResultSet type: 1004, Concurrency type: 1007, Supports: true
ResultSet type: 1004, Concurrency type: 1008, Supports: true
ResultSet type: 1005, Concurrency type: 1007, Supports: false
ResultSet type: 1005, Concurrency type: 1008, Supports: false
Releasing all open resources ...
guilhermekameoka@192 JDBCTutorial %
```

6. **Modificando e corrigindo o código de exemplo para atualizar a tabela de depósitos, adicionando juros de 0.5% em todas as linhas da tabela.**

```java
public static void modifyPricesJuros(Connection con) throws SQLException {
    Statement stmt = null;
    try {
        stmt = con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CONCUR_UPDATABLE);
        ResultSet uprs = stmt.executeQuery(sql:"SELECT * FROM deposito");
        while (uprs.next()) {
            double saldo_deposito = uprs.getDouble(columnLabel:"saldo_deposito");
            double updatedBalance = saldo_deposito * 1.005;
            uprs.updateDouble(columnLabel:"saldo_deposito", updatedBalance);
            uprs.updateRow();
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        if (stmt != null) {
            stmt.close();
        }
    }
}
```

Resultado após o acréscimo de juros: (executamos o método getMyData4 novamente)

```
Connected to database
Clientes e suas somas de depósitos e empréstimos:
Cliente: Adilson de Oliveira
Agência: Glória
Número da Conta: 61521
Total de Depósitos: 4856.471549999999
Total de Empréstimos: 0.0

Cliente: Adilson de Oliveira
Agência: Glória
Número da Conta: 16910
Total de Depósitos: 1220.8237499999998
Total de Empréstimos: 0.0

Cliente: Alexandre Márcio de Souza
Agência: Gameleira
Número da Conta: 24067
Total de Depósitos: 4227.592799999999
Total de Empréstimos: 515.06

Cliente: Andrade de Freitas
Agência: Central
Número da Conta: 72069
Total de Depósitos: 2125.374
```

Os valores foram acrescidos de 0.5%

7. **Modificando o código para receber um input do teclado referente à taxa de juros a ser aplicada nas tabelas**

```java
public static void modifyPricesJurosTeclado(Connection con) throws SQLException {
    Statement stmt = null;
    try {
        System.out.println(x:"Digite a porcentagem de juros a ser aplicada (Exemplo: 5% = 5.0):");
        Scanner in = new Scanner(System.in);
        double percentage = in.nextDouble();

        stmt = con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CONCUR_UPDATABLE);
        ResultSet uprs = stmt.executeQuery(sql:"SELECT * FROM deposito");
        while (uprs.next()) {
            double saldo_deposito = uprs.getDouble(columnLabel:"saldo_deposito");
            double updatedBalance = saldo_deposito * (1 + (percentage / 100.0));
            uprs.updateDouble(columnLabel:"saldo_deposito", updatedBalance);
            uprs.updateRow();
        }
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        if (stmt != null) {
            stmt.close();
        }
    }
}
```

Acrescentando mais 5% de juros

```
guilhermekameoka@192 JDBCTutorial % ./comp MyQueries properties/postgres-propert
ies.xml
Set the following properties:
dbms: postgresql
driver: org.postgresql.Driver
dbName: IB2
userName: guilhermekameoka
serverName: localhost
portNumber: 5432
Connected to database
Digite a porcentagem de juros a ser aplicada (Exemplo: 5% = 5.0):
5
Releasing all open resources ...
guilhermekameoka@192 JDBCTutorial %
```

Tabela atualizada com os 5% de juros

```
Clientes e suas somas de depósitos e empréstimos:
Cliente: Adilson de Oliveira
Agência: Glória
Número da Conta: 61521
Total de Depósitos: 5099.2951275
Total de Empréstimos: 0.0

Cliente: Adilson de Oliveira
Agência: Glória
Número da Conta: 16910
Total de Depósitos: 1281.8649374999998
Total de Empréstimos: 0.0

Cliente: Alexandre Márcio de Souza
Agência: Gameleira
Número da Conta: 24067
Total de Depósitos: 4438.97244
Total de Empréstimos: 515.06

Cliente: Andrade de Freitas
Agência: Central
Número da Conta: 72069
Total de Depósitos: 2231.6427
Total de Empréstimos: 0.0
```

8. **Modificando a aula 10 para fazer a inserção das linhas da tabela debito com batch:**

```java
public static void populateTableBatch(Connection con) throws SQLException {
    Statement stmt = null;
    String create = "";
    con.setAutoCommit(autoCommit:false);

    try {
        stmt = con.createStatement();
        stmt.executeUpdate(sql:"TRUNCATE TABLE debito");

        BufferedReader inputStream = new BufferedReader(new FileReader(
                fileName:"/Users/guilhermekameoka/Desktop/mydir/JDBCTutorial/BD2-15-debito-populate-table.txt"));
        String line;

        while ((line = inputStream.readLine()) != null) {
            create = "INSERT INTO debito (numero_debito, valor_debito, motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) ";
            create += "VALUES " + line;

            stmt.addBatch(create);
        }

        inputStream.close();

        //int[] updateCounts = stmt.executeBatch();
        con.commit();
    } catch (SQLException e) {
        e.printStackTrace();
        con.rollback();
    } catch (IOException e) {
        e.printStackTrace();
        System.out.println(x:"Falha na leitura do arquivo!");
    } finally {
        if (stmt != null) {
            stmt.close();
        }
        con.setAutoCommit(autoCommit:true);
    }
}
```

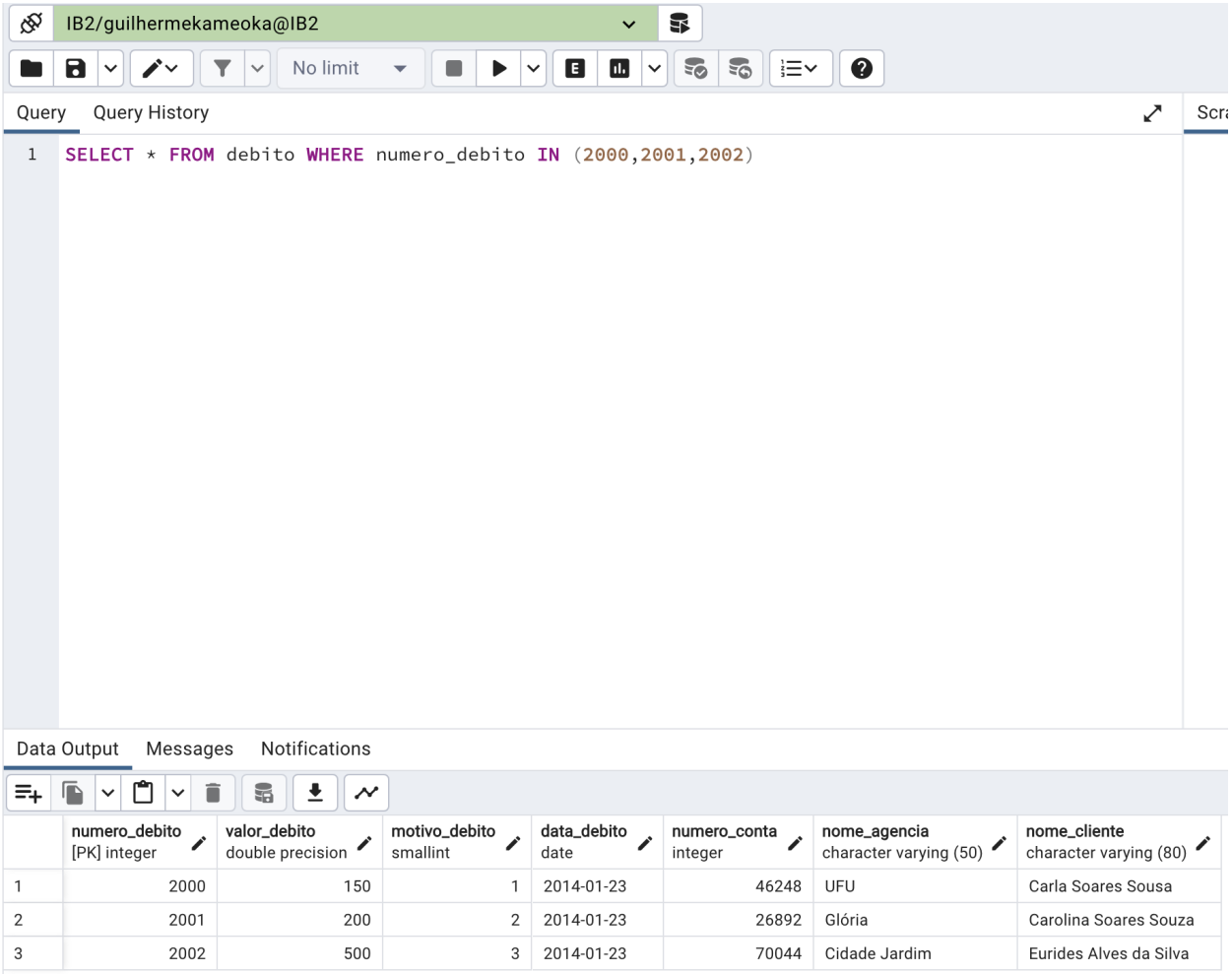
9. Fazendo a inserção das tuplas

```java
public static void insertRows(Connection con) throws SQLException {
    String sql = "INSERT INTO debito (numero_debito, valor_debito, motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) "
            + "VALUES (?, ?, ?, ?, ?, ?, ?)";

    try (PreparedStatement pstmt = con.prepareStatement(sql)) {
        // Primeira inserção
        pstmt.setInt(parameterIndex:1, x:2000);
        pstmt.setDouble(parameterIndex:2, x:150);
        pstmt.setInt(parameterIndex:3, x:1);
        pstmt.setDate(parameterIndex:4, Date.valueOf(s:"2014-01-23"));
        pstmt.setInt(parameterIndex:5, x:46248);
        pstmt.setString(parameterIndex:6, x:"UFU");
        pstmt.setString(parameterIndex:7, x:"Carla Soares Sousa");
        pstmt.executeUpdate();

        // Segunda inserção
        pstmt.setInt(parameterIndex:1, x:2001);
        pstmt.setDouble(parameterIndex:2, x:200);
        pstmt.setInt(parameterIndex:3, x:2);
        pstmt.setDate(parameterIndex:4, Date.valueOf(s:"2014-01-23"));
        pstmt.setInt(parameterIndex:5, x:26892);
        pstmt.setString(parameterIndex:6, x:"Glória");
        pstmt.setString(parameterIndex:7, x:"Carolina Soares Souza");
        pstmt.executeUpdate();

        // Terceira inserção
        pstmt.setInt(parameterIndex:1, x:2002);
        pstmt.setDouble(parameterIndex:2, x:500);
        pstmt.setInt(parameterIndex:3, x:3);
        pstmt.setDate(parameterIndex:4, Date.valueOf(s:"2014-01-23"));
        pstmt.setInt(parameterIndex:5, x:70044);
        pstmt.setString(parameterIndex:6, x:"Cidade Jardim");
        pstmt.setString(parameterIndex:7, x:"Eurides Alves da Silva");
        pstmt.executeUpdate();
    } catch (SQLException e) {
        JDBCUtilities.printSQLException(e);
    }
}
```



	numero_debito [PK] integer	valor_debito double precision	motivo_debito smallint	data_debito date	numero_conta integer	nome_agencia character varying (50)	nome_cliente character varying (80)
1	2000	150	1	2014-01-23	46248	UFU	Carla Soares Sousa
2	2001	200	2	2014-01-23	26892	Glória	Carolina Soares Souza
3	2002	500	3	2014-01-23	70044	Cidade Jardim	Eurides Alves da Silva