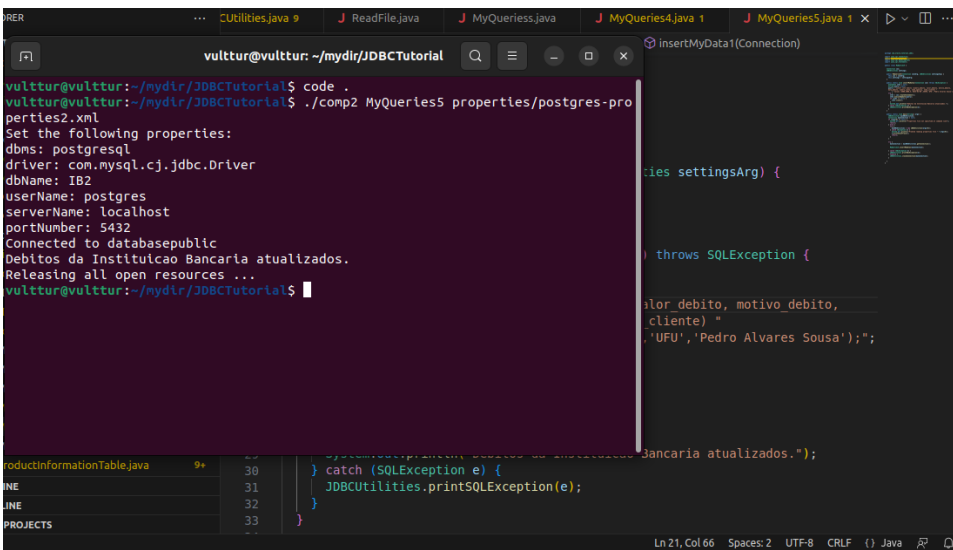


# TRABALHO 09 – COMANDOS PRE-COMPILADOS

Nome: Joao Otavio Rodrigues de Castro Manieri

Matricula: 12021BSI263

1.



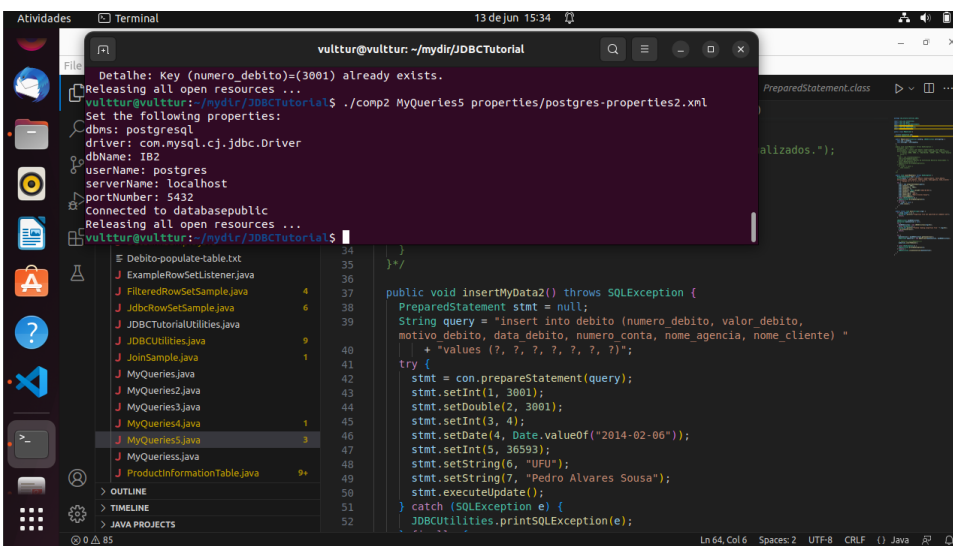
```
vultur@vultur: ~/mydir/JDBCTutorial
vultur@vultur:~/mydir/JDBCTutorial$ code .
vultur@vultur:~/mydir/JDBCTutorial$ ./comp2 MyQueries5 properties/postgres-properties2.xml
Set the following properties:
dbms: postgresql
driver: com.mysql.cj.jdbc.Driver
dbName: IB2
userName: postgres
serverName: localhost
portNumber: 5432
Connected to databasepublic
Debitos da Instituicao Bancaria atualizados.
Releasing all open resources ...
vultur@vultur:~/mydir/JDBCTutorial$
```

```
productInformationTable.java 9+
LINE
LINE
PROJECTS
```

```
30
31
32
33
```

```
Ln 21, Col 66 Spaces: 2 UTF-8 CRLF ( ) Java
```

2.



```
Atividades Terminal 13 de jun 15:34
vultur@vultur: ~/mydir/JDBCTutorial
Detalhe: Key (numero_debito)=(3001) already exists.
Releasing all open resources ...
vultur@vultur:~/mydir/JDBCTutorial$ ./comp2 MyQueries5 properties/postgres-properties2.xml
Set the following properties:
dbms: postgresql
driver: com.mysql.cj.jdbc.Driver
dbName: IB2
userName: postgres
serverName: localhost
portNumber: 5432
Connected to databasepublic
Releasing all open resources ...
vultur@vultur:~/mydir/JDBCTutorial$
```

```
Debito-populate-table.txt
J ExampleRowSetListener.java
J FilteredRowSetSample.java 4
J JdbcRowSetSample.java 6
J JDBCTutorialUtilities.java
J JDBCUtilities.java 9
J JoinSample.java 1
J MyQueries.java
J MyQueries2.java
J MyQueries3.java
J MyQueries4.java 1
J MyQueries5.java 3
J MyQueries6.java
J ProductInformationTable.java 9+
> OUTLINE
> TIMELINE
> JAVA PROJECTS
```

```
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
```

```
Ln 64, Col 6 Spaces: 2 UTF-8 CRLF ( ) Java
```

Dashboard Properties SQL Statistics Dependencies Dependents Processes IB2/postgres@localhost

IB2/postgres@localhost

Query Query History Scratch Pad

```

1 TRUNCATE TABLE debito;
2
3 select * from debito where numero_conta = 36593
4 and nome_agencia = 'UFU' and nome_cliente = 'Pedro Alvares Sousa';

```

Data Output Messages Notifications

	numero_debito [PK] integer	valor_debito double precision	motivo_debito smallint	data_debito date	numero_conta integer	nome_agencia character varying	nome_cliente character varying
1	3001	3001	4	2014-02-06	36593	UFU	Pedro Alvares Sousa

Successfully run. Total query runtime: 161 msec. 1 rows affected.

Total rows: 1 of 1 Query complete 00:00:00.161 Ln 3, Col 1

3.

```

vulture@vulture: ~/mydir/DBCTutorial$ ./comp2 MyQueries5 properties/postgres-properties2.xml
Set the following properties:
dbms: postgresql
driver: com.mysql.cj.jdbc.Driver
dbName: IB2
userName: postgres
serverName: localhost
portNumber: 5432
Connected to databasepublic
Um debito em IB2 inserido em 9 milissegundos
Releasing all open resources ...
vulture@vulture: ~/mydir/DBCTutorial$

```

MyQueries.java 84  
ProductInformationTable.java 9+ 85  
86  
87  
88  
89

```

myQueries.insertMyData1();
/* myQueries.insertMyData2();*/
} catch (SQLException e) {
    JDBCUtilities.printStackTrace(e);
}

```

Ln 85, Col 9 Spaces: 2 UTF-8 CRLF (1) Java

```

vulture@vulture: ~/mydir/DBCTutorial$ ./comp2 MyQueries5 properties/postgres-properties2.xml
Set the following properties:
dbms: postgresql
driver: com.mysql.cj.jdbc.Driver
dbName: IB2
userName: postgres
serverName: localhost
portNumber: 5432
Connected to databasepublic
Um debito em IB2 inserido em 6 milissegundos
Releasing all open resources ...
vulture@vulture: ~/mydir/DBCTutorial$

```

MyQueries.java 38  
ProductInformationTable.java 9+ 39  
40  
41  
42

```

public void insertMyData2() throws SQLException {
    PreparedStatement stmt = null;
    String query = "insert into debito (numero_debito, valor_debito,
    motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) "
    + "values (?, ?, ?, ?, ?, ?, ?)";
}

```

Ln 37, Col 5 Spaces: 2 UTF-8 CRLF (1) Java

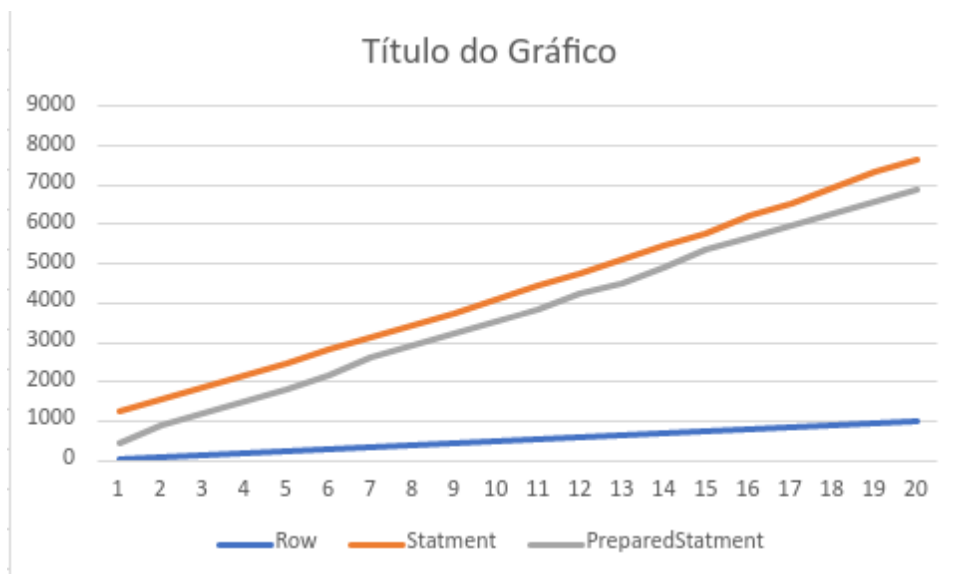
Com isso constatamos que o insertMyData2 é mais rapido

4.

```
Releasing all open resources ...
vulttur@vulttur: ~/mydir/JDBCTutorial$ ./comp2 MyQueries5 properties/postgres-properties2.xml
Set the following properties:
dbms: postgresql
driver: com.mysql.cj.jdbc.Driver
dbName: IB2
userName: postgres
serverName: localhost
portNumber: 5432
Connected to databasepublic
Um debito em IB2 inserido em 49 milisegundos
Um debito em IB2 inserido em 59 milisegundos
50      1239
100     1535
150     1851
200     2163
250     2493
300     2818
350     3137
400     3441
450     3733
500     4104
550     4433
600     4770
650     5093
700     5464
750     5784
800     6242
850     6549
900     6932
950     7338
1000    7675
Um debito da IB2 inserido em 7683 milisegundos
50      417
100     871
150     1174
```

```
vulttur@vulttur: ~/mydir/JDBCTutorial
500     4104
550     4433
600     4770
650     5093
700     5464
750     5784
800     6242
850     6549
900     6932
950     7338
1000    7675
Um debito da IB2 inserido em 7683 milisegundos
50      417
100     871
150     1174
200     1501
250     1819
300     2165
350     2639
400     2919
450     3226
500     3534
550     3836
600     4232
650     4504
700     4909
750     5371
800     5658
850     5987
900     6273
950     6566
1000    6896
Um debito da IB2 inserido em 6904 milisegundos
Releasing all open resources ...
vulttur@vulttur: ~/mydir/JDBCTutorial$
```

	A	B	C	D	E	F	G	H
1	Row	Statment	PreparedStatment					
2	50	1239	417					
3	100	1535	871					
4	150	1851	1174					
5	200	2163	1501					
6	250	2493	1819					
7	300	2818	2165					
8	350	3137	2639					
9	400	3441	2919					
10	450	3733	3226					
11	500	4104	3534					
12	550	4433	3836					
13	600	4770	4232					
14	650	5093	4504					
15	700	5464	4909					
16	750	5784	5371					
17	800	6242	5658					
18	850	6549	5987					
19	900	6932	6273					
20	950	7338	6566					
21	1000	7675	6896					
22								



Com esse resultado podemos constatar novamente que o PreparedStatment é mais rapido que o Statment

5.

```
6 delete from debito where numero_conta = 36593 and nome_agencia = 'U
7 nome_cliente = 'Pedro Alvares Sousa' and numero_debito >= 3000;
```

Data Output Messages Notifications

DELETE 2002

Query returned successfully in 101 msec.

✓ Query returned successfully in 101 msec. ✕

Total rows: 1 of 1 Query complete 00:00:00.101 Ln 6, Col 1

MyQueries5.java - JDBC Tutorial - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

- JDBCTUTORIAL
  - src/com/oracle/tutorial/jdbc
    - CachedRowSetSample.java
    - CityFilter.java
    - ClobSample.java
    - CoffeesFrame.java
    - CoffeesTable.java
    - CoffeesTableModel.java
    - DataLinkSample.java
    - Debito-populate-table.txt
    - ExampleRowSetListener.java
    - FilteredRowSetSample.java
    - JdbcRowSetSample.java
    - JDBCTutorialUtilities.java
    - JDBCUtilities.java
    - JoinSample.java
    - MyQueries.java
    - MyQueries2.java
    - MyQueries3.java
    - MyQueries4.java
    - MyQueries5.java
    - MyQueries.java
    - ProductInformationTable.java

OUTLINE

TIMELINE

JAVA PROJECTS

```
src > com > oracle > tutorial > jdbc > J MyQueries5.java > MyQueries5 > main(String[])
142
143
144
145
146
147

public void insertMyData3000() throws SQLException {
    PreparedStatement stmt = null;
    try {
        con.setAutoCommit(false);

        String query = "insert into debito (numero_debito, valor_debito,
        motivo_debito, data_debito, numero_conta, nome_agencia, nome_cliente) "
        + "values (?, ?, ?, ?, ?, ?, ?)";
        stmt = con.prepareStatement(query);

        long startTime = System.currentTimeMillis();
        for (int numdeb = 5002; numdeb < 6002; numdeb++) {
            stmt.setInt(1, numdeb);
            stmt.setInt(2, numdeb);
            stmt.setInt(3, 5);
            stmt.setDate(4, Date.valueOf("2014-02-06"));
            stmt.setInt(5, 36593);
            stmt.setString(6, "UFU");
            stmt.setString(7, "Pedro Alvares Sousa");
            stmt.addBatch();

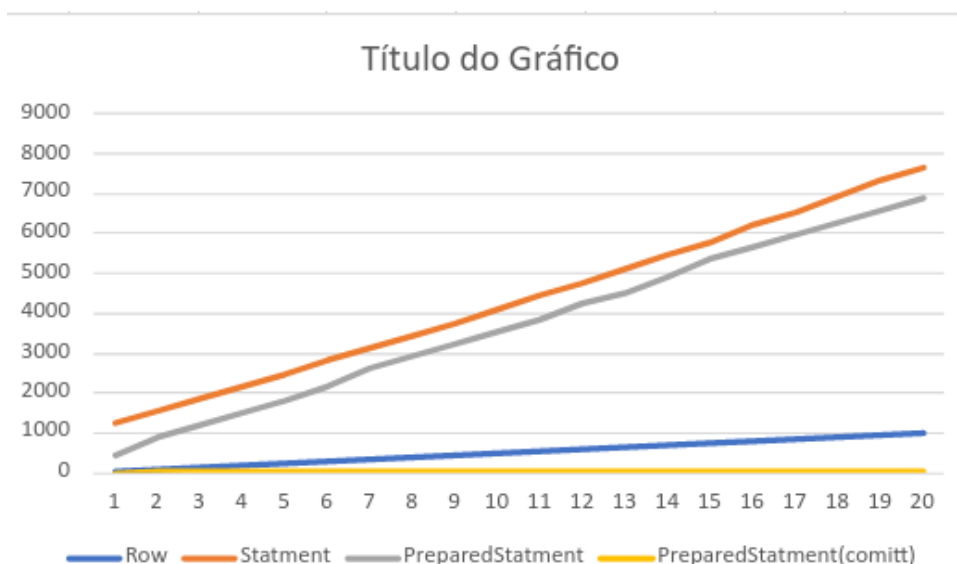
            if ((numdeb % 50) == 0) {
                long endTime = System.currentTimeMillis();
                System.out.println(numdeb - 5000 + "\t" + (endTime - startTime));
            }
        }
    }
}
```

Ln 190, Col 1 Spaces: 2 UTF-8 CRLF () Java

```

500      3206
550      3490
600      3788
650      4339
700      4593
750      4854
800      5120
850      5687
900      5958
950      6223
1000     6504
Um debito da IB2 inserido em 6513 milisegundos
50        3
100       4
150       5
200       5
250       5
300       7
350       7
400       8
450       9
500      13
550      13
600      14
650      16
700      16
750      16
800      17
850      17
900      17
950      17
1000     17
Um debito da IB2 inserido em 85 milisegundos
Releasing all open resources ...
vulttur@vulttur:~/mydir/JDBCTutorial$

```



Com isso constatmos que desabilitar o comitt tornou a funcao absurdamente mais rapida quando comparadas as outras duas

6.

Quando desabilitamos o auto-commit para a classe `Statement`, cada instrução SQL é tratada como uma transação separada. Isso significa que as alterações feitas em uma instrução não serão permanentes até que chamemos explicitamente o método `commit()`.

No caso do uso da classe `PreparedStatement`, podemos aproveitar o recurso do Batch Processing, que permite agrupar várias instruções SQL em uma única chamada ao banco de dados. Ao usar o Batch Processing, as alterações só serão confirmadas quando chamarmos o método `executeBatch()` seguido do método `commit()`.

Em resumo, ao desabilitar o auto-commit com a classe `Statement`, cada instrução requer uma chamada separada para confirmar as alterações. Com o `PreparedStatement` e o Batch Processing, podemos agrupar várias instruções e confirmar as alterações de uma vez, o que pode melhorar o desempenho.