Instituto Federal Goiano - Campus Ceres
Bacharelado em Sistemas de Informação
Prof. Me. Ronneesley Moura Teles

Daniel Moreira Cardoso
Gusttavo Nunes Gomes
Jeferson Rossini Ferreira Lourenço
Paulo Henrique Rodrigues Araujo
Warley Rodrigues de Andrade

JDBC com transações

Outubro
2017

Sumário

JDBC com transações

1 Introdução

A linguagem de programação Java, diferente de linguagens como PHP, não suporta o acesso a banco de dados diretamente, para isso ele usa uma API (conjunto de classes e interfaces) para fazer o serviço. A JDBC (Java Database Connectivity), faz o envio de instruções para qualquer banco de dados relacional, desde que haja um driver que corresponda ao mesmo presente. Muitos podem encontrar uma certa semelhança entre JDBC e ODBC; estão absolutamente corretos, podemos dizer a "grosso modo"que as duas seguem a mesma ideia. Ambas funcionam como meio de comunicação Banco X Aplicação, porém, ODBC é uma aplicação Windows restrito apenas a ele, enquanto a JDBC, por ser escrita em java, é multiplataforma.[1]

2 O que é JDBC?

Java Database Connectivy (JDBC) é um API (Conjunto de classes e interfaces) que permite a comunicação da linguagem JAVA com qualquer banco de dados relacional. Porém é necessário um driver específico para cada SGBD. [2]

3 Tipos de JDBC

Existem quatro tipos de drivers JDBC:

1 - Tipo restrito a plataforma Windows. Esse tipo utiliza uma ODBC para se conectar com um banco de dados. Todas as chamadas ao JDBC s"ao traduzidas para chamadas IDBC (Open DataBase Connectivity);
2 - O driver JDBC faz chamadas a algum outro código normalmente escrito em C. Tanto o primeiro quando o segundo necessita de um software extra junto a sua aplicação Java;
3 - Converte a chamada JDBC para uma chamada de rede. Nesta tecnologia, temos um servidor provendo a conexão com o banco de dados. Todo o código é escrito em Java;
4 - Natividade 100%, ou seja, todas as chamadas JDBC são convertidas diretamente para o protocolo de comunicação do banco de dados, ou seja, acontece um acesso direto ao DBMS.[3]

4 O que é transação?

Transação é o nome que é dado à unidade de trabalho (dentro de um procedimento ou função) única, lógica e indivisível dentro de um SGBD. O código da transação é criado e disponibilizado dentro de um procedimento ou uma função: se o processamento não for realizado de forma completa ele será totalmente cancelado. Isto é feito para manter a consistência dos dados e das operações dentro do banco. [4] Um bom exemplo de transação é a transferência bancária entre contas. Onde o valor é debitado da conta de um cliente e depois adicionado na conta de outro. Para fazer esta transação é muito simples, conforme o código abaixo:

id	nome	saldo
1	Daniel	900
2	Paulo	200

```
1 update clientes set saldo = saldo-200 where id = 1;
2 update clientes set saldo = saldo+200 where id = 2;
```

recursos/codigo/01.sql

Para a transação ocorrer os dois comandos devem ser válidos, se a transação for válida os dados são enviados ao banco de dados (*commit*). Caso não seja válida, ou se algo der errado o procedimento deve ser abortado (rollback). Quando se fala em *commit* e *rollback* é importante falar de autocommit, que permite que as operações sejam commitadas automaticamente. A maioria dos SGBDS usam por modo padrão o *Autocommit*, ou seja, quando são executados comandos de UPDATE, INSERT ou DELETE sem iniciar uma transação explícita, essas operações são commitadas de forma automática.

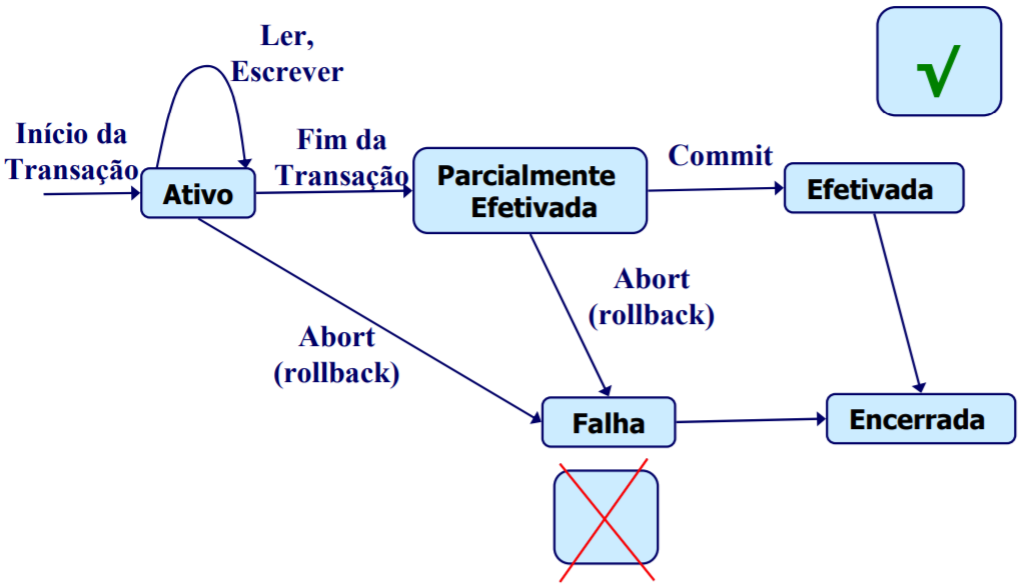

Figura 1: Fonte: https://www.ime.usp.br/ jef/TransacoesControleConcorrencia.pdf

Porém as transações normalmente são feitas a nível de programação e não a nível de banco de dados (*procedures*) para maior facilidade caso seja necessário migrar a base de dados para um outro SGBD. Agora que já vimos o que é o JDBC e transações, vamos entender como fazer transações com JDBC. Para fazer commit com java basta chamar o método commit() da interface connection.

```
connection.commit();
```

O método rollback também está na interface connection

```
connection.rollback();
```

O auto commit também pode ser feito em java e permite gerenciar automaticamente se será feito o commit, basta no início do método chamar o método passando como argumento um valor lógico (true ou false);

```
connection.SetAutoCommit(true);
```

Então vejamos um exemplo bastante genérico de uma transação em JAVA:

```
connection.setAutoCommit(false);
// Digite o sql aqui

connection.commit();
// se der errado

connection.rollback();
```

recursos/codigo/02.java

Implementando um exemplo com uma estrutura de controle.

```java
//Considere que ja tenha sido declarado e instanciados todos os objetos necessarios
try {
  connection.setAutoCommit(false);
  preparedStatement = connection.prepareStatement( "Um SQL qualquer aqui" );

  preparedStatement.executeUpdate();

  connection.commit();
}catch( Exception e ) {
  connection.rollback();
}
```

recursos/codigo/03.java

5 Referências Bibliográfica

[1] http://www.linhadecodigo.com.br/artigo/1711/java-acesso-a-dados-usando-jdbc.aspx

[2] https://pt.wikipedia.org/wiki/JDBC

[3] https://jmmwrite.wordpress.com/2009/08/12/diferencas-entre-as-versoes-do-jdbc/

[4] REVISTABW. Transações em Bancos de Dados.Revista Brasileira de Web: Tecnologia. Disponível em: <http://www.revistabw.com.br/revistabw/transacoes-em-bancos-de Criado em: 03/01/2013. Última atualização: 24/07/2015. Visitado em: 31/10/2017

[5] https://www.devmedia.com.br/aprendendo-java-com-jdbc/29116

[6] http://www.guj.com.br/t/transacoes-em-java/29674