DR1

Disciplina Regular 1 Fundamentos do Desenvolvimento Java

Graduação em Engenharia de Software - 2020

DR1

Etapa 5 Aula 1

Acesso a bancos de dados com JDBC

DR1

Competências Trabalhadas Nesta Etapa

- Implementar o acesso a dados com Java
  - Compreender a arquitetura JDBC.
  - Criar bancos de dados, tabelas e relacionamentos com o MySQL Workbench.
  - Executar comandos SQL de inserção, atualização, exclusão e seleção.
  - Compreender a hierarquia de coleções.
  - Manipular coleções para tratar o resultado de uma consulta ao banco de dados.

No Moodle esse conteúdo se refere à etapa 9

Visão Geral

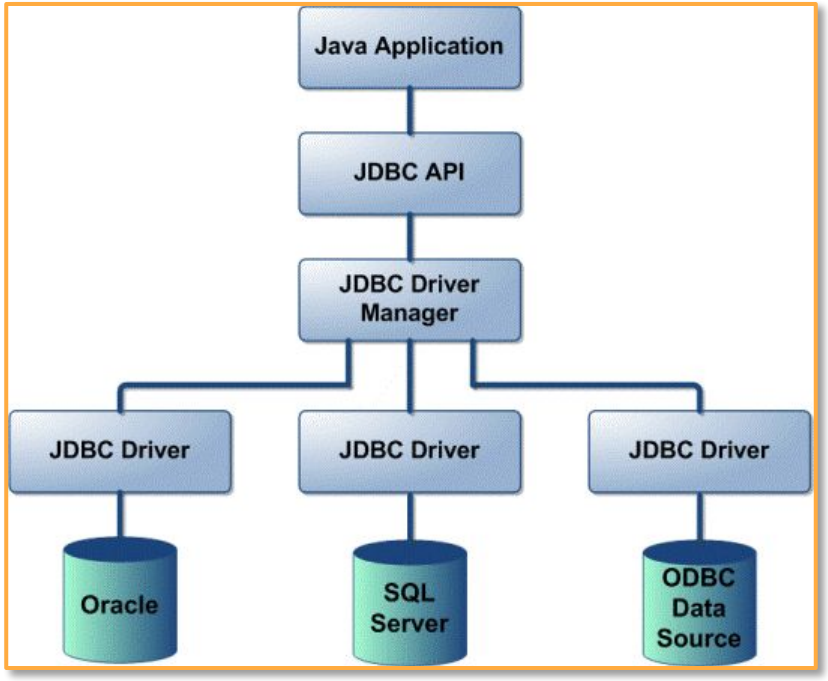
DR1

- JDBC é um conjunto de classes e *interfaces* independentes de fabricante para conexão a bancos de dados.

- É composto pelo pacote java.sql.

- Utiliza um *driver* para conectar a aplicação a um ou mais servidores de BD, executa consultas SQL e opera os resultados.

Visão Geral



JDBC é uma abstração do banco de dados

Visão Geral

DR1

- Roteiro de Programação:
  - Carga do *driver* na memória.
  - Criação da conexão usando o DriverManager.
  - Uso da conexão para acessar o banco de dados:
    - **Create** = criar registros no banco de dados.
    - **Retrieve** = obter registros do banco de dados.
    - **Update** = atualizar registros no banco de dados.
    - **Delete** = excluir registros do banco de dados.

Estabelecendo Conexões

DR1

- O “driver” é uma classe que aponta para uma biblioteca de classes “pares” das *interfaces* definidas no JDBC.

- Essas bibliotecas são distribuídas em arquivos JAR, o que permite a sua utilização nas nossas aplicações.

- O arquivo JAR precisa ser colocado no CLASSPATH da aplicação.

Estabelecendo Conexões

DR1

- Utilizaremos o *driver* JDBC-ODBC Bridge que já se encontra na instalação do JDK.

- Precisamos carregar o *driver* na memória para que o DriverManager possa reconhecer e criar a conexão.

```
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

ou

```
DriverManager.registerDriver (new JdbcOdbcDriver());
```

e então

```
Connection con =
     DriverManager.getConnection("url", "user", "pwd");
```

A primeira opção é preferível por causa da *string* que pode ser passada como parâmetro

Estabelecendo Conexões

DR1

- “url” = *Uniform Resource Locator* – *string* que identifica o protocolo de comunicação com o banco de dados.

```
jdbc:<subprotocolo>:<subnome>
String url = “jdbc:odbc:NorthwindDSN”;
```

- “user” = nome do usuário do banco de dados.
- “pwd” = senha do usuário do banco de dados.

Esses parâmetros devem ser obtidos via arquivo de properties

Estabelecendo Conexões

DR1

- A *interface* **Connection** representa uma conexão com um banco de dados.

- A informação sobre a conexão pode ser obtida usando o método getMetaData.

- É usada para criar comandos SQL a serem enviadas para o banco de dados.

Estabelecendo Conexões

DR1

- SQLException é um classe do pacote java.sql e do tipo “Checked Exception”.

- Métodos importantes:
  - getMessage = String descrevendo o erro
  - getErrorCode = Código específico do BD
  - getSQLState = Instrução SQL que deu erro

Statement

DR1

- Envia comandos SQL para o banco de dados. Necessita de uma conexão ativa.

```
Statement consulta = conexao.createStatement();
```

- O objeto **Statement** existe, mas ainda não possui nenhum comando SQL associado.

- Só deve utilizar comandos SQL simples, sem parâmetros.

Statement

DR1

- Método executeQuery é usado para consultas que retornam resultados.

```
String sql = “select * from livros”;
ResultSet rs = consulta.executeQuery(sql);
```

- O resultado da consulta vem em um objeto do tipo ResultSet.

Statement

DR1

- Método executeUpdate é usado para consultas DML, INSERT, UPDATE e DELETE e comandos DDL.

```
String sql = "delete from vendas";
int totalApagados = consulta.executeUpdate();
```

- O resultado da consulta retorna um **int** cujo valor é representado pelo número de linhas afetadas.

DML = Data Modeling Language

DDL = Data Definition Language

Statement

DR1

- Método execute é usado quando a consulta retorna um ResultSet, ou um indicador de linhas afetadas.

```
String SQL = ...;
boolean ehConsulta = consulta.execute(SQL);
```

- Retorna “true” se o resultado é um ResultSet, “false” se é um int.

PreparedStatement

DR1

- Trabalha com declarações SQL precompiladas.
- É subclasse de Statement.
- Deve ser usado sempre que a query tiver parâmetros.

```
PreparedStatement ps =
  conexao.prepareStatement("update livros set preco
  = ? where cod_livro = ?");
```

PreparedStatement

DR1

- Depois de preparada, devemos passar os valores para cada um dos parâmetros.

```
ps.setDouble(1, 35.00);
ps.setString(2, “08037-9383”);
```

- Em seguida executamos a consulta usando um dos métodos a seguir:

```
ps.executeQuery();
ps.executeUpdate();
```

ResultSet

DR1

- Armazena os resultados das consultas realizadas.
- Mantém um cursor apontando para o registro de dados atual.
- Com o método **next** podemos andar pelo conjunto resultado linha a linha.
- Utilizamos os métodos **get...** para recuperar os valores.

ResultSet

DR1

```
...
while (rs.next()) {
   String cod_livro = rs.getString("CODIGO");
   String nome_livro = rs.getString("NOME");
   double preco = rs.getDouble("PRECO");
   ...
}
```

Concluindo o Processamento

- Após o uso, devemos fechar todos os objetos **ResultSet** e **Statement** a fim de limpar a memória e liberar cursores do banco de dados.

```
...
rs.close();
consulta.close();
conexao.close()
...
```

É importante fechar as conexões para não saturar o banco de dados