02. Executando comandos SQL no Java

Date	@12/12/2022
Categoria	Java e persistência
Curso	Java e JDBC: trabalhando com um banco de dados

Tópicos

- Projeto da aula anterior
- Listagem com Statement
- Connection, Statement e ResultSet
- Criando a ConnectionFactory
- Factory Pattern
- Inserindo com Statement
- Retorno do método execute()
- Removendo dados
- Faça que eu fiz
- O que aprendemos?

Listagem com Statement

```
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;

public class TestaListagem{

    public static void main(String[] args) throws SQLException{

        ConnectionFactory connectionFactory = new ConnectionFactory();
        Connection connection = connectionFactory.recuperarConexao();

        Statement stm = connection.createStatement();
        stm.execute("SELECT ID, NOME, DESCRICAO FROM PRODUTO");

        ResultSet rst = stm.getResultSet();

        while(rst.next()){
            Integer id = rst.getInt("ID");
            System.out.println(id);
            String nome = rst.getString("NOME");
            System.out.println(nome);
            String descricao = rst.getString("DESCRICAO")
            System.out.println(descricao);
        }

        con.close();
    }
}
```

Connection, Statement e ResultSet

O que o `java.sql.Connection`, `java.sql.Statement` e `java.sql.ResultSet` possuem em comum?

- Todas são interfaces: `Connection`, `Statement` e `ResultSet` são algumas das interface do pacote `java.sql`.

Criando a ConnectionFactory

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;


public class CriaConexao{

    public Connection recuperarConexao() throws SQLException{
        return DriverManager.getConnection("jdbc:mysql://localhost/loja_virtual?useTimezone=true&serverTimezone=UTC","root","root");
    }
}
```

Factory Pattern

Qual a vantagem de utilizar uma `ConnectionFactory` na nossa aplicação?

- Fornecer uma maneira mais simples para criar um objeto
    - Criamos objetos sem expor a lógica ou as configurações de criação ao cliente. Além disso, podemos nos referir ao objeto recém-criado usando uma interface (usando uma abstração), desacoplando a implementação.

Inserindo com Statement

```
import java.sql.Connection;
import java.sql.SQLException;
import java.sql.Statement;

public class TestaInsercao {
  public static void main(String[] args) throws SQLException{
    ConnectionFactory factory = new ConnectionFactory();
    Connection connection = factory.recuperarConexao();

    Statement stm = connection.createStatement();
    stm.execute("INSERT INTO PRODUTO (nome, descricao) VALUES ('Mouse', 'Mouse sem fio')"
    , Statement.RETURN_GENERATED_KEYS);

    ResultSet rst = stm.getGeneratedKeys();
    while(rst.next()){
      Integer id = rst.getInt(1);
      System.out.println("Id criado: " + id);
    }
  }
}
```

Retorno do método execute()

Como funciona o retorno do método `execute`, da interface `java.sql.Statement`?

- O método devolve `true` quando o seu resultado é um `java.sql.ResultSet` e `false` caso contrário (*update*, *delete*, etc).

Removendo dados

```
import java.sql.Connection;
import java.sql.SQLException;
```

```
import java.sql.Statement;


public class TestaRemocao{

    public static void main(String[] args) throws SQLException{

        ConnectionFactory factory = new ConnectionFactory();
        Connection connection = factory.recuperarConexao();

        Statement stm = connection.createStatement();
        stm.execute("DELETE FROM PRODUTO WHERE ID > 2");

        Integer linhasModificadas = stm.getUpdateCount();

        System.out.println("Quantidade de linhas que foram modificadas: " + linhasModificadas);

    }
}
```

O que aprendemos?

Nesta aula, aprendemos que:

- Para simplificar e encapsular a criação da conexão, devemos usar uma classe `ConnectionFactory`
  - A classe `ConnectionFactory` segue o padrão de criação *Factory Method*
  - O *Factory Method* encapsula a criação de um objeto
- Para executar um comando SQL, podemos usar a interface `java.sql.Statement`
  - O método `execute` envia o comando para o banco de dados
  - Dependendo do comando SQL, podemos recuperar a chave primária ou os registros selecionados.
