

**INSTITUTO FEDERAL DE MINAS GERAIS
CAMPUS SÃO JOÃO EVANGELISTA
SISTEMAS DE INFORMAÇÃO**

VALDIR DE SOUZA CARVALHO NETO

Linguagem de Programação I - SI 241
Aula 10 - Exercícios de Aprendizagem

SUMÁRIO

1. Exercício 1.....	4
2. Exercício 2.....	6

LISTA DE FIGURAS

Figura 1. Captura de tela Atividade 01.....	6
Figura 2. Captura de tela Atividade 02.....	11

1. Exercício 1

Desenvolva uma aplicação em Java que implemente uma classe responsável pela criação e fornecimento de conexões com um banco de dados MySQL, utilizando o driver JDBC do MySQL conforme visto em sala de aula. Essa classe deverá possuir atributos estáticos e privados para armazenar os parâmetros de conexão com o banco de dados, incluindo o driver JDBC, a URL de conexão, o usuário e a senha, garantindo o encapsulamento adequado dessas informações.

A classe deverá disponibilizar um método estático e público responsável por instanciar e retornar um objeto do tipo `Connection`, realizando o carregamento do driver, o estabelecimento da conexão com o banco de dados e o tratamento das exceções que possam ocorrer durante esse processo. O código desenvolvido deve seguir boas práticas de programação em Java e demonstrar claramente o uso do método estático para obtenção da conexão a partir de outra classe.

Código:

Arquivo Main:

```
package com.mycompany.ex01;

import java.sql.*;
import java.util.Scanner;

public class Ex01 {

    public static void main(String[] args) {
        Connection conn = ConnectionFactory.getConexao();

        if (conn != null) {
            try {
                String nome;
                Scanner scan = new Scanner(System.in);
                System.out.println("Informe um nome: ");
                nome = scan.next();

                String sql = "insert into clientes (nome) values ( ?
);";

                // Preparar o comando sql
                PreparedStatement stmt = conn.prepareStatement(sql);
                stmt.setString(1, nome);
```

```

        int linhas = stmt.executeUpdate();

        if (linhas > 0) {
            System.out.println("Cliente inserido com
sucesso!");
        } else {
            System.out.println("Nenhum cliente foi
inserido!");
        }

        sql = "select * from clientes;";

        Statement stmt2 = conn.createStatement();
        ResultSet rs = stmt2.executeQuery(sql);

        System.out.println("\n----- CLIENTES
-----");

        while (rs.next()) {
            System.out.println("Codigo: " +
rs.getString("id"));
            System.out.println("Nome..: " +
rs.getString("nome"));
        }
        System.out.println("-----");

        conn.close();
    } catch (SQLException ex) {
        System.out.println("Erro do BD: " + ex.getMessage());
    }

    } else {
        System.out.println("Erro na conexao com BD!");
    }
}
}

```

Arquivo da Factory:

```

package com.mycompany.ex01;

import java.sql.*;

```

```

public class ConnectionFactory {

    private static final String user = "root";
    private static final String pwd = "teste1";
    private static final String url = "jdbc:mysql://localhost/dbtest";
    private static final String driver = "com.mysql.cj.jdbc.Driver";

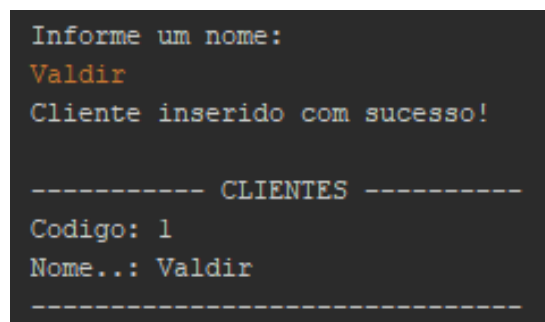
    public static Connection getConexao() {
        try {
            // Selecionar o driver JDBC para a aplicação
            Class.forName(driver);

            // Estabelecer conexão com DB
            return DriverManager.getConnection(url, user, pwd);
        } catch (ClassNotFoundException ex) {
            System.out.println("Erro: " + ex.getMessage());
            return null;
        } catch (SQLException ex) {
            System.out.println("Erro do BD: " + ex.getMessage());
            return null;
        }
    }
}

```

Imagens:

Figura 1. Captura de tela Atividade 01



```

Informe um nome:
Valdir
Cliente inserido com sucesso!

----- CLIENTES -----
Codigo: 1
Nome.: Valdir
-----

```

Fonte: Captura de tela retirada pelo autor em 18 de dezembro de 2025.

2. Exercício 2

Desenvolva uma classe denominada **Cliente**, responsável por representar e manipular os dados de um cliente no sistema. A classe deverá conter um construtor, métodos setters e getters para seus atributos, além de métodos que implementem as

operações básicas de persistência de dados no banco de dados, conforme os conceitos trabalhados em sala de aula.

A classe `Cliente` deverá possuir os seguintes métodos: `inserir()`, que não recebe parâmetros e utiliza os dados do próprio objeto para realizar a inserção no banco de dados; `alterar(Cliente novoCliente)`, que recebe como parâmetro um objeto contendo os novos dados a serem atualizados; `carregar(int id)`, responsável por receber o identificador do cliente e carregar suas informações a partir do banco de dados; e `excluir()`, que não recebe parâmetros e remove do banco de dados o cliente representado pelo objeto atual. O código deve seguir boas práticas de programação orientada a objetos e utilizar a classe de conexão desenvolvida anteriormente.

Código:

Arquivo Main:

```
package com.mycompany.ex02;

public class Ex02 {

    public static void main(String[] args) {
        System.out.println("--- Teste de Inserção ---");
        Cliente c1 = new Cliente();
        c1.setNome("Teste Valdir");
        c1.inserir();

        System.out.println("\n--- Teste de Carregamento ---");
        Cliente c2 = new Cliente();
        int idTeste = 1;
        c2.carregar(idTeste);
        System.out.println("Nome recuperado: " + c2.getNome());

        System.out.println("\n--- Teste de Alteração ---");
        Cliente novosDados = new Cliente();
        novosDados.setNome("Valdir Alterado");
        c2.alterar(novosDados);
        c2.carregar(idTeste);
        System.out.println("Nome recuperado: " + c2.getNome());

        System.out.println("\n--- Teste de Exclusão ---");
        c2.excluir();
    }
}
```

Arquivo da Factory:

```
package com.mycompany.ex01;

import java.sql.*;

public class ConnectionFactory {

    private static final String user = "root";
    private static final String pwd = "teste1";
    private static final String url = "jdbc:mysql://localhost/dbtest";
    private static final String driver = "com.mysql.cj.jdbc.Driver";

    public static Connection getConexao() {
        try {
            // Selecionar o driver JDBC para a aplicação
            Class.forName(driver);

            // Estabelecer conexão com DB
            return DriverManager.getConnection(url, user, pwd);
        } catch (ClassNotFoundException ex) {
            System.out.println("Erro: " + ex.getMessage());
            return null;
        } catch (SQLException ex) {
            System.out.println("Erro do BD: " + ex.getMessage());
            return null;
        }
    }
}
```

Arquivo da classe Cliente:

```
package com.mycompany.ex02;

import java.sql.*;

public class Cliente {
    private int id;
    private String nome;

    public Cliente() {}
}
```



```
public int getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getNome() {
    return nome;
}

public void setNome(String nome) {
    this.nome = nome;
}

public void inserir() {
    String sql = "insert into clientes (nome) values ( ? );";
    Connection conn = ConnectionFactory.getConexao();

    try {
        // Preparar o comadndo sql
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setString(1, this.getNome());

        int linhas = stmt.executeUpdate();

        if (linhas > 0) {
            System.out.println("Cliente inserido com sucesso!");
        } else {
            System.out.println("Nenhum cliente foi inserido!");
        }

        conn.close();
    } catch (SQLException ex) {
        System.out.println("Erro do BD: " + ex.getMessage());
    }
}

public void alterar(Cliente novoCliente) {
    String sql = "update clientes set nome = ? where id = ?";
    Connection conn = ConnectionFactory.getConexao();
}
```

```

try {
    // Preparar o comando sql
    PreparedStatement stmt = conn.prepareStatement(sql);
    stmt.setString(1, novoCliente.getNome());
    stmt.setInt(2, this.getId());

    int linhas = stmt.executeUpdate();

    if (linhas > 0) {
        System.out.println("Cliente alterado com sucesso!");
    } else {
        System.out.println("Nenhuma mudança executada!");
    }

    conn.close();
} catch (SQLException ex) {
    System.out.println("Erro do BD: " + ex.getMessage());
}

}

public void carregar(int id) {
    String sql = "select * from clientes where id = ?;";
    Connection conn = ConnectionFactory.getConexao();

    try {
        // Preparar o comando sql
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setInt(1, id);

        ResultSet rs = stmt.executeQuery();

        if (rs.next()) {
            this.setId(rs.getInt("id"));
            this.setNome(rs.getString("nome"));
        } else {
            System.out.println("Cliente não encontrado!");
        }

        conn.close();
    } catch (SQLException ex) {
        System.out.println("Erro do BD: " + ex.getMessage());
    }
}

```

```

    }

}

public void excluir() {
    String sql = "delete from clientes where id = ?;";
    Connection conn = ConnectionFactory.getConexao();

    try {
        // Preparar o comando sql
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setInt(1, this.getId());

        int linhas = stmt.executeUpdate();

        if (linhas > 0) {
            System.out.println("Cliente excluido com sucesso!");
        } else {
            System.out.println("Cliente nao encontrado!");
        }

        conn.close();
    } catch (SQLException ex) {
        System.out.println("Erro do BD: " + ex.getMessage());
    }
}
}

```

Imagens:

Figura 2. Captura de tela Atividade 02

```

--- Teste de Insercao ---
Cliente inserido com sucesso!

--- Teste de Carregamento ---
Nome recuperado: Teste Valdir

--- Teste de Alteracao ---
Cliente alterado com sucesso!
Nome recuperado: Valdir Alterado

--- Teste de Exclusao ---
Cliente excluido com sucesso!

```

Fonte: Captura de tela retirada pelo autor em 18 de dezembro de 2025.