Sistema de Cadastro de Alunos*

Sergio Henrique Floriano Silva[1], João Felipe dos Santos Silva[2]

Universidade Estácio de Sá (UNESA)

202001566732@alunos.estacio.br, 202003190731@alunos.estacio.br

Resumo. *O Sistema de Cadastro de Alunos é uma solução prática e funcional que permite armazenar informações de alunos em um banco de dados PostgreSQL. Com esse sistema, é possível criar uma tabela para os alunos, adicionar novos registros, listar todos os alunos cadastrados, atualizar seus dados e também remover alunos do sistema. Embora seja uma implementação básica, o sistema atende aos requisitos essenciais para gerenciar informações de alunos.*

1. Código

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class SistemaCadastroAlunos {
    private static final String URL = "jdbc:postgresql://localhost:5432/postgres";
    private static final String USUARIO = "postgres";
    private static final String SENHA = "1234";

    public static Connection conectar() throws SQLException {
        try {
            Class.forName("org.postgresql.Driver");
        } catch (ClassNotFoundException e) {
            System.out.println("Erro ao carregar o driver JDBC do PostgreSQL: " +
e.getMessage());
        }
```

```
    return DriverManager.getConnection(URL, USUARIO, SENHA);
}

public static void criarTabela() {
    String sql = "CREATE TABLE IF NOT EXISTS alunos ("
            + "id SERIAL PRIMARY KEY,"
            + "nome VARCHAR(100) NOT NULL,"
            + "nota FLOAT NOT NULL"
            + ")";

    try (Connection conn = conectar();
         Statement stmt = conn.createStatement()) {

        stmt.executeUpdate(sql);
        System.out.println("Tabela 'alunos' criada com sucesso!");

    } catch (SQLException e) {
        System.out.println("Erro ao criar tabela 'alunos': " + e.getMessage());
    }
}

public static void adicionarAluno(String nome, float nota) {
    String sql = "INSERT INTO alunos (nome, nota) VALUES (?, ?)";

    try (Connection conn = conectar();
         PreparedStatement pstmt = conn.prepareStatement(sql)) {

        pstmt.setString(1, nome);
        pstmt.setFloat(2, nota);
        pstmt.executeUpdate();

        System.out.println("Aluno adicionado com sucesso!");

    } catch (SQLException e) {
        System.out.println("Erro ao adicionar aluno: " + e.getMessage());
```

```
    }
}

public static void listarAlunos() {
    String sql = "SELECT * FROM alunos";

    try (Connection conn = conectar();
         Statement stmt = conn.createStatement();
         ResultSet rs = stmt.executeQuery(sql)) {

        System.out.println("Lista de Alunos:");
        while (rs.next()) {
            int id = rs.getInt("id");
            String nome = rs.getString("nome");
            float nota = rs.getFloat("nota");

            System.out.println("ID: " + id);
            System.out.println("Nome: " + nome);
            System.out.println("Nota: " + nota);
            System.out.println("------------------------");
        }

    } catch (SQLException e) {
        System.out.println("Erro ao listar alunos: " + e.getMessage());
    }
}

public static void atualizarAluno(int id, String novoNome, float novaNota) {
    String sql = "UPDATE alunos SET nome = ?, nota = ? WHERE id = ?";

    try (Connection conn = conectar();
         PreparedStatement pstmt = conn.prepareStatement(sql)) {

        pstmt.setString(1, novoNome);
        pstmt.setFloat(2, novaNota);
```

```
            pstmt.setInt(3, id);
            pstmt.executeUpdate();

            System.out.println("Aluno atualizado com sucesso!");

        } catch (SQLException e) {
            System.out.println("Erro ao atualizar aluno: " + e.getMessage());
        }
    }

    public static void excluirAluno(int id) {
        String sql = "DELETE FROM alunos WHERE id = ?";

        try (Connection conn = conectar();
             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            pstmt.setInt(1, id);
            pstmt.executeUpdate();

            System.out.println("Aluno excluído com sucesso!");

        } catch (SQLException e) {
            System.out.println("Erro ao excluir aluno: " + e.getMessage());
        }
    }

    public static void listarAprovados() {
        String sql = "SELECT * FROM alunos WHERE nota >= 6";

        try (Connection conn = conectar();
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(sql)) {

            System.out.println("Alunos Aprovados:");
            while (rs.next()) {
```

```
            int id = rs.getInt("id");
            String nome = rs.getString("nome");
            float nota = rs.getFloat("nota");

            System.out.println("ID: " + id);
            System.out.println("Nome: " + nome);
            System.out.println("Nota: " + nota);
            System.out.println("------------------------");
        }

    } catch (SQLException e) {
        System.out.println("Erro ao listar alunos aprovados: " + e.getMessage());
    }
}

public static void listarReprovados() {
    String sql = "SELECT * FROM alunos WHERE nota < 6";

    try (Connection conn = conectar();
         Statement stmt = conn.createStatement();
         ResultSet rs = stmt.executeQuery(sql)) {

        System.out.println("Alunos Reprovados:");
        while (rs.next()) {
            int id = rs.getInt("id");
            String nome = rs.getString("nome");
            float nota = rs.getFloat("nota");

            System.out.println("ID: " + id);
            System.out.println("Nome: " + nome);
            System.out.println("Nota: " + nota);
            System.out.println("------------------------");
        }

    } catch (SQLException e) {
```

```
            System.out.println("Erro ao listar alunos reprovados: " + e.getMessage());
        }
    }

    public static void main(String[] args) {
        criarTabela();

        adicionarAluno("João", 7.8f);
        adicionarAluno("Maria", 6.5f);
        adicionarAluno("Pedro", 5.2f);
        adicionarAluno("Ana", 8.9f);

        System.out.println("Todos os Alunos:");
        listarAlunos();

        System.out.println("Alunos Aprovados:");
        listarAprovados();

        System.out.println("Alunos Reprovados:");
        listarReprovados();

        atualizarAluno(3, "Lucas", 7.0f);

        System.out.println("Todos os Alunos Atualizados:");
        listarAlunos();

        excluirAluno(2);

        System.out.println("Todos os Alunos Após Exclusão:");
        listarAlunos();
    }
}
```

2. Detalhamento da Implementação

a. **O nome do Sistema:**

Avenue - Sistema de Cadastro de Alunos

b. **O objetivo do Sistema. Qual a problemática que o sistema irá resolver:**

O objetivo do sistema é permitir o cadastro, consulta, atualização e exclusão de alunos em um banco de dados PostgreSQL. O sistema resolve a problemática de armazenar e gerenciar informações de alunos, como nome e nota, de forma persistente.

c. **Os requisitos do Sistema. O que o sistema irá fazer?**

Criar uma tabela "alunos" no banco de dados (se não existir) com os campos: id (chave primária), nome e nota.

Adicionar um aluno ao banco de dados, fornecendo o nome e a nota.

Listar todos os alunos cadastrados no banco de dados.

Atualizar os dados de um aluno existente no banco de dados, fornecendo o ID do aluno, novo nome e nova nota.

Excluir um aluno do banco de dados, fornecendo o ID do aluno.

Listar todos os alunos aprovados (nota >= 6).

Listar todos os alunos reprovados (nota < 6).

d. **Caso de uso do Sistema:**

O usuário inicia o programa.

O programa cria a tabela "alunos" no banco de dados (se não existir).

O programa adiciona alguns alunos ao banco de dados.

O programa lista todos os alunos cadastrados.

O programa lista os alunos aprovados.

O programa lista os alunos reprovados.

O programa atualiza os dados de um aluno.

O programa lista todos os alunos após a atualização.

O programa exclui um aluno.

O programa lista todos os alunos após a exclusão.

O programa termina.

e. Explicação do que foi implementado:

A classe SistemaCadastroAlunos é a classe principal do programa.

O método conectar estabelece a conexão com o banco de dados PostgreSQL usando as informações de URL, usuário e senha fornecidas.

O método criarTabela cria a tabela "alunos" no banco de dados, se ela ainda não existir.

O método adicionarAluno insere um novo aluno na tabela, fornecendo o nome e a nota do aluno.

O método listarAlunos recupera todos os alunos da tabela e os exibe no console.

O método atualizarAluno atualiza os dados de um aluno existente na tabela, fornecendo o ID do aluno, novo nome e nova nota.

O método excluirAluno exclui um aluno da tabela, fornecendo o ID do aluno.

O método listarAprovados recupera e exibe no console todos os alunos aprovados (nota >= 6).

O método listarReprovados recupera e exibe no console todos os alunos reprovados (nota < 6).

O método main é o ponto de entrada do programa. Ele chama os métodos acima em uma sequência lógica para demonstrar o funcionamento do sistema.

f. Conclusão

No entanto, é importante destacar que o código é uma implementação simplificada e não aborda todos os aspectos do desenvolvimento de um sistema completo de cadastro de alunos, temos dois pontos que poderiam ser melhorados numa implementação mais complexa para atender a uma demanda real que não fosse apenas um exercício avaliativo como:

Segurança: O código atual não lida com questões de segurança, como a prevenção de injeção de SQL. Recomenda-se o uso de consultas parametrizadas ou o uso de PreparedStatements para mitigar esse tipo de vulnerabilidade.

Validação de dados: O exemplo não inclui validação de dados de entrada. É importante garantir que os dados fornecidos pelos usuários sejam validados adequadamente para evitar erros ou inconsistências.