



RELATÓRIO DISCENTE DE ACOMPANHAMENTO

Campus Polo Cavallhada – Porto Alegre

DESENVOLVIMENTO FULL STACK

Missão Prática | Nível 3 | Mundo 3

RPG0016 - BackEnd sem banco não tem

Turma: 9003

Semestre Letivo: 3º - 2024.1

GUILHERME BERNARDES BASTOS

Repositório: <https://github.com/guilhermebernardes96/Mundo3-Nivel3>


CRIANDO O BANCO DE DADOS

Objetivo da Prática

- Implementar persistência com base no JDBC.
- Utilizar o padrão DAO no manuseio de dados.
- Implementar o mapeamento objeto-relacional em sistemas Java.
- Criar sistemas cadastrais com persistência em banco.

1º Procedimento

CÓDIGOS



The screenshot shows a database IDE interface. At the top, the connection string is 'jdbc:mysql://localhost:3306/LOJA?zeroDateTimeBehavior=CONVERT_TO_NULL [loja on Default schema]'. Below this, a tree view shows the 'loja' database schema with the following structure:

- Tables
 - movimento
 - peessoa
 - peessoaJuridica
 - peessoafisica
 - produto
 - sequenciaPessoa
 - usuario
- Views
- Procedures

Below the tree view, a Java code file is open, showing the following code:

```
package cadastro.model;

import cadastro.model.util.ConectorBD;
import cadastro.model.util.SequenceManager;
import cadastrobd.model.PessoaFisica;
import java.sql.SQLException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

public class PessoaFisicaDAO {
    private ConectorBD conectorBD;
    private SequenceManager sequenceManager;

    public PessoaFisicaDAO() {
        this.conectorBD = new ConectorBD();
        this.sequenceManager = new SequenceManager();
    }

    public PessoaFisica getPessoa(int id) throws SQLException {
        PessoaFisica pessoa = null;
        Connection connection = null;
        PreparedStatement preparedStatement = null;
        ResultSet resultSet = null;

        try {
            connection = ConectorBD.getConnection();

            String sql = "SELECT p.idPessoa, p.nome, p.logradouro, p.cidade, p.estado, p.telefone, p.email, pf.cpf " +
                "FROM pessoa p INNER JOIN pessoaFisica pf ON p.idPessoa = pf.idPessoa " +
                "WHERE p.idPessoa = ?";

            preparedStatement = connection.prepareStatement(sql);
            preparedStatement.setInt(1, id);
            resultSet = preparedStatement.executeQuery();
        }
    }
}
```

```

        if (resultSet.next()) {
            pessoa = new PessoaFisica(
                resultSet.getInt("idPessoa"),
                resultSet.getString("nome"),
                resultSet.getString("logradouro"),
                resultSet.getString("cidade"),
                resultSet.getString("estado"),
                resultSet.getString("telefone"),
                resultSet.getString("email"),
                resultSet.getString("cpf")
            );
        }
    } finally {
        ConectorBD.close(resultSet);
        ConectorBD.close(preparedStatement);
        ConectorBD.close(connection);
    }

    return pessoa;
}

public List<PessoaFisica> getPessoas() throws SQLException {
    List<PessoaFisica> pessoas = new ArrayList<>();
    Connection connection = null;
    PreparedStatement preparedStatement = null;
    ResultSet resultSet = null;

    try {
        connection = ConectorBD.getConnection();

        String sql = "SELECT p.idPessoa, p.nome, p.logradouro, p.cidade, p.estado, p.telefone, p.email, pf.cpf " +
            "FROM pessoa p " +
            "JOIN pessoaFisica pf ON p.idPessoa = pf.idPessoa";

        preparedStatement = connection.prepareStatement(sql);
        resultSet = preparedStatement.executeQuery();
    }

```

```

        while (resultSet.next()) {
            PessoaFisica pessoa = new PessoaFisica(
                resultSet.getInt("idPessoa"),
                resultSet.getString("nome"),
                resultSet.getString("logradouro"),
                resultSet.getString("cidade"),
                resultSet.getString("estado"),
                resultSet.getString("telefone"),
                resultSet.getString("email"),
                resultSet.getString("cpf")
            );
            pessoas.add(pessoa);
        }
    } finally {
        ConectorBD.close(resultSet);
        ConectorBD.close(preparedStatement);
        ConectorBD.close(connection);
    }

    return pessoas;
}

public void incluir(PessoaFisica pessoa) throws SQLException {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    try {
        connection = ConectorBD.getConnection();
        connection.setAutoCommit(false);

        String sqlPessoa = "INSERT INTO pessoa (nome, logradouro, cidade, estado, telefone, email) VALUES (?, ?, ?, ?, ?, ?)";
        preparedStatement = connection.prepareStatement(sqlPessoa, Statement.RETURN_GENERATED_KEYS);
        preparedStatement.setString(1, pessoa.getNome());
        preparedStatement.setString(2, pessoa.getLogradouro());
        preparedStatement.setString(3, pessoa.getCidade());
        preparedStatement.setString(4, pessoa.getEstado());
        preparedStatement.setString(5, pessoa.getTelefone());
    }

```

```

        preparedStatement.setString(6, pessoa.getEmail());
        preparedStatement.executeUpdate();

        ResultSet generatedKeys = preparedStatement.getGeneratedKeys();
        int pessoaId;
        if (generatedKeys.next()) {
            pessoaId = generatedKeys.getInt(1);
        } else {
            throw new SQLException("Erro ao gerar ID.");
        }

        String sqlPessoaFisica = "INSERT INTO pessoaFisica (idPessoa, cpf) VALUES (?, ?)";
        preparedStatement = connection.prepareStatement(sqlPessoaFisica);
        preparedStatement.setInt(1, pessoaId);
        preparedStatement.setString(2, pessoa.getCpf());
        preparedStatement.executeUpdate();

        connection.commit();
    } catch (SQLException e) {
        if (connection != null) {
            connection.rollback();
        }
        throw e;
    } finally {
        if (connection != null) {
            connection.setAutoCommit(true);
        }
        ConectorBD.close(preparedStatement);
        ConectorBD.close(connection);
    }
}

public void alterar(PessoaFisica pessoa) throws SQLException {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

```

```

    try {
        connection = ConectorBD.getConnection();
        connection.setAutoCommit(false);

        String sqlPessoa = "UPDATE pessoa SET nome = ?, logradouro = ?, cidade = ?, estado = ?, telefone = ?, email = ? WHERE idPessoa = ?";
        preparedStatement = connection.prepareStatement(sqlPessoa);
        preparedStatement.setString(1, pessoa.getNome());
        preparedStatement.setString(2, pessoa.getLogradouro());
        preparedStatement.setString(3, pessoa.getCidade());
        preparedStatement.setString(4, pessoa.getEstado());
        preparedStatement.setString(5, pessoa.getTelefone());
        preparedStatement.setString(6, pessoa.getEmail());
        preparedStatement.setInt(7, pessoa.getId());
        preparedStatement.executeUpdate();

        String sqlPessoaFisica = "UPDATE pessoaFisica SET cpf = ? WHERE idPessoa = ?";
        preparedStatement = connection.prepareStatement(sqlPessoaFisica);
        preparedStatement.setString(1, pessoa.getCpf());
        preparedStatement.setInt(2, pessoa.getId());
        preparedStatement.executeUpdate();

        connection.commit();
    } catch (SQLException e) {
        if (connection != null) {
            connection.rollback();
        }
        throw e;
    } finally {
        if (connection != null) {
            connection.setAutoCommit(true);
        }
        ConectorBD.close(preparedStatement);
        ConectorBD.close(connection);
    }
}

```

```

public void excluir(int id) throws SQLException {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    try {
        connection = ConectorBD.getConnection();

        String sqlPessoa = "DELETE FROM pessoa WHERE idPessoa = ?";

        String sqlPessoaFisica = "DELETE FROM pessoaFisica WHERE idPessoa = ?";

        connection.setAutoCommit(false);

        preparedStatement = connection.prepareStatement(sqlPessoa);
        preparedStatement.setInt(1, id);
        preparedStatement.executeUpdate();

        preparedStatement = connection.prepareStatement(sqlPessoaFisica);
        preparedStatement.setInt(1, id);
        preparedStatement.executeUpdate();

        connection.commit();
    } catch (SQLException e) {
        if (connection != null) {
            connection.rollback();
        }
        throw e;
    } finally {
        if (connection != null) {
            connection.setAutoCommit(true);
        }
        ConectorBD.close(preparedStatement);
        ConectorBD.close(connection);
    }
}

```

```

package cadastro.model;

import cadastrobd.model.PessoaJuridica;
import cadastro.model.util.ConectorBD;
import cadastro.model.util.SequenceManager;
import java.sql.SQLException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.List;
import java.sql.Statement;

public class PessoaJuridicaDAO {
    private ConectorBD conectorBD;
    private SequenceManager sequenceManager;

    public PessoaJuridicaDAO() {
        this.conectorBD = new ConectorBD();
        this.sequenceManager = new SequenceManager();
    }

    public PessoaJuridica getPessoa(int id) throws SQLException {
        PessoaJuridica pessoa = null;
        Connection connection = null;
        PreparedStatement preparedStatement = null;
        ResultSet resultSet = null;

        try {
            connection = ConectorBD.getConnection();

            String sql = "SELECT p.idPessoa, p.nome, p.logradouro, p.cidade, p.estado, p.telefone, p.email, pj.cnpj " +
                "FROM Pessoa p INNER JOIN pessoaJuridica pj ON p.idPessoa = pj.idPessoa " +
                "WHERE p.idPessoa = ?";

            preparedStatement = connection.prepareStatement(sql);
            preparedStatement.setInt(1, id);
            resultSet = preparedStatement.executeQuery();

```

```

        if (resultSet.next()) {
            pessoa = new PessoaJuridica(
                resultSet.getInt("idPessoa"),
                resultSet.getString("nome"),
                resultSet.getString("logradouro"),
                resultSet.getString("cidade"),
                resultSet.getString("estado"),
                resultSet.getString("telefone"),
                resultSet.getString("email"),
                resultSet.getString("cnpj")
            );
        }
    } finally {
        ConectorBD.close(resultSet);
        ConectorBD.close(preparedStatement);
        ConectorBD.close(connection);
    }

    return pessoa;
}

public List<PessoaJuridica> getPessoas() throws SQLException {
    List<PessoaJuridica> pessoas = new ArrayList<>();
    Connection connection = null;
    PreparedStatement preparedStatement = null;
    ResultSet resultSet = null;

    try {
        connection = ConectorBD.getConnection();

        String sql = "SELECT p.idPessoa, p.nome, p.logradouro, p.cidade, p.estado, p.telefone, p.email, pj.cnpj " +
            "FROM pessoa p " +
            "JOIN pessoaJuridica pj ON p.idPessoa = pj.idPessoa";

        preparedStatement = connection.prepareStatement(sql);
        resultSet = preparedStatement.executeQuery();
    }

```

```

        while (resultSet.next()) {
            PessoaJuridica pessoa = new PessoaJuridica(
                resultSet.getInt("idPessoa"),
                resultSet.getString("nome"),
                resultSet.getString("logradouro"),
                resultSet.getString("cidade"),
                resultSet.getString("estado"),
                resultSet.getString("telefone"),
                resultSet.getString("email"),
                resultSet.getString("cnpj")
            );
            pessoas.add(pessoa);
        }
    } finally {
        ConectorBD.close(resultSet);
        ConectorBD.close(preparedStatement);
        ConectorBD.close(connection);
    }

    return pessoas;
}

public void incluir(PessoaJuridica pessoa) throws SQLException {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    try {
        connection = ConectorBD.getConnection();
        connection.setAutoCommit(false);

        String sqlPessoa = "INSERT INTO Pessoa (nome, logradouro, cidade, estado, telefone, email) VALUES (?, ?, ?, ?, ?, ?)";
        preparedStatement = connection.prepareStatement(sqlPessoa, Statement.RETURN_GENERATED_KEYS);
        preparedStatement.setString(1, pessoa.getNome());
        preparedStatement.setString(2, pessoa.getLogradouro());
        preparedStatement.setString(3, pessoa.getCidade());
        preparedStatement.setString(4, pessoa.getEstado());
        preparedStatement.setString(5, pessoa.getTelefone());
        preparedStatement.setString(6, pessoa.getEmail());
    }

```

```

        preparedStatement.executeUpdate();

        ResultSet generatedKeys = preparedStatement.getGeneratedKeys();
        int pessoaId;
        if (generatedKeys.next()) {
            pessoaId = generatedKeys.getInt(1);
        } else {
            throw new SQLException("Erro ao gerar ID.");
        }

        String sqlPessoaJuridica = "INSERT INTO PessoaJuridica (idPessoa, cnpj) VALUES (?, ?)";
        preparedStatement = connection.prepareStatement(sqlPessoaJuridica);
        preparedStatement.setInt(1, pessoaId);
        preparedStatement.setString(2, pessoa.getCnpj());
        preparedStatement.executeUpdate();

        connection.commit();

    } catch (SQLException e) {
        if (connection != null) {
            connection.rollback();
        }
        throw e;
    } finally {
        if (connection != null) {
            connection.setAutoCommit(true);
        }
        ConectorBD.close(preparedStatement);
        ConectorBD.close(connection);
    }
}

public void alterar(PessoaJuridica pessoa) throws SQLException {
    Connection connection = null;
    PreparedStatement preparedStatement = null;

    try {
        connection = ConectorBD.getConnection();

```

```

        connection.setAutoCommit(false);

        String sqlPessoa = "UPDATE pessoa SET nome = ?, logradouro = ?, cidade = ?, estado = ?, telefone = ?, email = ? WHERE idPessoa = ?";
        preparedStatement = connection.prepareStatement(sqlPessoa);
        preparedStatement.setString(1, pessoa.getNome());
        preparedStatement.setString(2, pessoa.getLogradouro());
        preparedStatement.setString(3, pessoa.getCidade());
        preparedStatement.setString(4, pessoa.getEstado());
        preparedStatement.setString(5, pessoa.getTelefone());
        preparedStatement.setString(6, pessoa.getEmail());
        preparedStatement.setInt(7, pessoa.getId());
        preparedStatement.executeUpdate();

        String sqlPessoaJuridica = "UPDATE pessoaJuridica SET cnpj = ? WHERE idPessoa = ?";
        preparedStatement = connection.prepareStatement(sqlPessoaJuridica);
        preparedStatement.setString(1, pessoa.getCnpj());
        preparedStatement.setInt(2, pessoa.getId());
        preparedStatement.executeUpdate();

        connection.commit();
    } catch (SQLException e) {
        if (connection != null) {
            connection.rollback();
        }
        throw e;
    } finally {
        if (connection != null) {
            connection.setAutoCommit(true);
        }
        ConectorBD.close(preparedStatement);
        ConectorBD.close(connection);
    }
}

public void excluir(int id) throws SQLException {
    Connection connection = null;

```

```

PreparedStatement preparedStatement = null;

try {
    connection = ConectorBD.getConnection();

    String sqlPessoa = "DELETE FROM pessoa WHERE idPessoa = ?";

    String sqlPessoaJuridica = "DELETE FROM pessoaJuridica WHERE idPessoa = ?";

    connection.setAutoCommit(false);

    preparedStatement = connection.prepareStatement(sqlPessoa);
    preparedStatement.setInt(1, id);
    preparedStatement.executeUpdate();

    preparedStatement = connection.prepareStatement(sqlPessoaJuridica);
    preparedStatement.setInt(1, id);
    preparedStatement.executeUpdate();

    connection.commit();
} catch (SQLException e) {
    if (connection != null) {
        connection.rollback();
    }
    throw e;
} finally {
    if (connection != null) {
        connection.setAutoCommit(true);
    }
    ConectorBD.close(preparedStatement);
    ConectorBD.close(connection);
}
}
}

```

```

package cadastrdbd.model;

public class PessoaJuridica extends Pessoa {
    private String cnpj;

    public PessoaJuridica() {
        super();
    }

    public PessoaJuridica (int id, String nome, String logradouro, String cidade, String estado, String telefone, String email, String cnpj) {
        super(id, nome, logradouro, cidade, estado, telefone, email); // Chama o construtor completo da classe pai Pessoa
        this.cnpj = cnpj;
    }

    @Override
    public void exibir() {
        super.exibir();
        System.out.println("CNPJ: " + cnpj);
    }

    public String getCnpj() {
        return cnpj;
    }

    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }
}

```



```

package cadastrobd.model;

public class PessoaFisica extends Pessoa {
    private String cpf;

    public PessoaFisica() {
        super();
    }

    public PessoaFisica(int id, String nome, String logradouro, String cidade, String estado, String telefone, String email, String cpf) {
        super(id, nome, logradouro, cidade, estado, telefone, email);
        this.cpf = cpf;
    }

    @Override
    public void exibir() {
        super.exibir();
        System.out.println("CPF: " + cpf);
    }

    public String getCpf() {
        return cpf;
    }

    public void setCpf(String cpf) {
        this.cpf = cpf;
    }
}

```

```

package cadastrobd.model;

public class Pessoa {
    private int id;
    private String nome;
    private String logradouro;
    private String cidade;
    private String estado;
    private String telefone;
    private String email;

    public Pessoa() {
    }

    public Pessoa(int id, String nome, String logradouro, String cidade, String estado, String telefone, String email) {
        this.id = id;
        this.nome = nome;
        this.logradouro = logradouro;
        this.cidade = cidade;
        this.estado = estado;
        this.telefone = telefone;
        this.email = email;
    }

    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 String getLogradouro() {
    return logradouro;
}

public void setLogradouro(String logradouro) {
    this.logradouro = logradouro;
}

public String getCidade() {
    return cidade;
}

public void setCidade(String cidade) {
    this.cidade = cidade;
}

public String getEstado() {
    return estado;
}

public void setEstado(String estado) {
    this.estado = estado;
}

public String getTelefone() {
    return telefone;
}

public void setTelefone(String telefone) {
    this.telefone = telefone;
}

public String getEmail() {
    return email;
}

public void setEmail(String email) {
    this.email = email;
}

public void exibir() {
    System.out.println("ID: " + id);
    System.out.println("Nome: " + nome);
    System.out.println("Logradouro: " + logradouro);
    System.out.println("Cidade: " + cidade);
    System.out.println("Estado: " + estado);
    System.out.println("Telefone: " + telefone);
    System.out.println("Email: " + email);
}
}
```

```

package cadastro.model.util;

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

public class ConectorBD {
    private static final String URL = "jdbc:mysql://localhost:3306/LOJA?zeroDateTimeBehavior=CONVERT_TO_NULL";
    private static final String USER = "loja";
    private static final String PASSWORD = "loja";

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL, USER, PASSWORD);
    }

    public static PreparedStatement getPrepared(String sql) throws SQLException {
        return getConnection().prepareStatement(sql);
    }

    public static ResultSet getSelect(String sql) throws SQLException {
        return getPrepared(sql).executeQuery();
    }

    public static void close(Statement statement) {
        if (statement != null) {
            try {
                statement.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}

```

```

    public static void close(ResultSet resultSet) {
        if (resultSet != null) {
            try {
                resultSet.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }

    public static void close(Connection connection) {
        if (connection != null) {
            try {
                connection.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}

```

```
package cadastro.model.util;
```

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

public class SequenceManager {
    public static int getValue(String sequenceName) throws SQLException {
        int nextValue = 0;
        Connection connection = null;
        PreparedStatement preparedStatement = null;
        ResultSet resultSet = null;

        try {
            connection = ConectorBD.getConnection();

            String sql = "Selecione o proximo valor para " + sequenceName;

            preparedStatement = connection.prepareStatement(sql);
            resultSet = preparedStatement.executeQuery();

            if (resultSet.next()) {
                nextValue = resultSet.getInt("nextValue");
            }
        } finally {
            ConectorBD.close(resultSet);
            ConectorBD.close(preparedStatement);
            ConectorBD.close(connection);
        }

        return nextValue;
    }
}
```

RESULTADO

```
package cadastro;

import cadastrobd.model.PessoaFisica;
import cadastro.model.PessoaFisicaDAO;
import cadastro.model.PessoaJuridicaDAO;
import cadastrobd.model.PessoaJuridica;
import java.sql.SQLException;
import java.util.Scanner;
import java.util.List;

public class CadastroBDTeste {
    public static void main(String[] args) {
        // CADASTRO PF
        Scanner scanner = new Scanner(System.in);
        System.out.println("NOVA PESSOA FISICA: ");

        System.out.print("NOME: ");
        String nome = scanner.nextLine();
        System.out.print("LOGRADOURO: ");
        String logradouro = scanner.nextLine();
        System.out.print("CIDADE: ");
        String cidade = scanner.nextLine();
        System.out.print("ESTADO: ");
        String estado = scanner.nextLine();
        System.out.print("TELEFONE: ");
        String telefone = scanner.nextLine();
        System.out.print("E-MAIL: ");
        String email = scanner.nextLine();
        System.out.print("CPF: ");
        String cpf = scanner.nextLine();
```

```
PessoaFisica pessoa = new PessoaFisica();
pessoa.setNome(nome);
pessoa.setLogradouro(logradouro);
pessoa.setCidade(cidade);
pessoa.setEstado(estado);
pessoa.setTelefone(telefone);
pessoa.setEmail(email);
pessoa.setCpf(cpf);

PessoaFisicaDAO pessoaFisicaDAO = new PessoaFisicaDAO();

try {
    pessoaFisicaDAO.incluir(pessoa);
    System.out.println("Pessoa fisica incluída com sucesso!");
} catch (SQLException e) {
    System.err.println("Erro ao incluir pessoa fisica: " + e.getMessage());
} finally {
    scanner.close();
}
```

```
run:
NOVA PESSOA FISICA:
NOME: JULIANO
LOGRADOURO: RUA NOVE
CIDADE: PORTO ALEGRE
ESTADO: RS
TELEFONE: 999999999
E-MAIL: JULIANO@NOVE.COM
CPF: 23423423423
Pessoa fisica incluida com sucesso!
BUILD SUCCESSFUL (total time: 59 seconds)
|
```

```
System.out.println("INSIRA OS NOVOS DADOS: ");
System.out.print("NOME: ");
String nome = scanner.nextLine();
System.out.print("LOGRADOURO: ");
String logradouro = scanner.nextLine();
System.out.print("CIDADE: ");
String cidade = scanner.nextLine();
System.out.print("ESTADO: ");
String estado = scanner.nextLine();
System.out.print("TELEFONE: ");
String telefone = scanner.nextLine();
System.out.print("E-MAIL: ");
String email = scanner.nextLine();
System.out.print("CPF: ");
String cpf = scanner.nextLine();

pessoa.setNome(nome);
pessoa.setLogradouro(logradouro);
pessoa.setCidade(cidade);
pessoa.setEstado(estado);
pessoa.setTelefone(telefone);
pessoa.setEmail(email);
pessoa.setCpf(cpf);

pessoaFisicaDAO.alterar(pessoa);
System.out.println("Dados da pessoa fisica alterados com sucesso!");
} else {
    System.out.println("Nenhuma pessoa fisica encontrada com esse ID.");
}
} catch (SQLException e) {
    System.err.println("Erro ao alterar pessoa fisica: " + e.getMessage());
} finally {
    scanner.close();
}
```

```
// ALTERAR DADOS PF
Scanner scanner = new Scanner(System.in);
System.out.println("ALTERAR DADOS DE PESSOA FISICA: ");
System.out.println();
System.out.println("ID da pessoa fisica que será alterada: ");

int id = scanner.nextInt();
scanner.nextLine();

PessoaFisicaDAO pessoaFisicaDAO = new PessoaFisicaDAO();

try {
    PessoaFisica pessoa = pessoaFisicaDAO.getPessoa(id);

    if (pessoa != null) {
        System.out.println("DADOS ATUAIS: ");
        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Logradouro: " + pessoa.getLogradouro());
        System.out.println("Cidade: " + pessoa.getCidade());
        System.out.println("Estado: " + pessoa.getEstado());
        System.out.println("Telefone: " + pessoa.getTelefone());
        System.out.println("Email: " + pessoa.getEmail());
        System.out.println("CPF: " + pessoa.getCpf());

        System.out.println();
    }
}

```

run:

ALTERAR DADOS DE PESSOA FISICA:

ID da pessoa fisica que será alterada:

1

DADOS ATUAIS:

Nome: JOCA

Logradouro: RUA JOCA

Cidade: JOAQUIM

Estado: JO

Telefone: 333333333

Email: JOCA@JOCA.COM

CPF: 33333333333

INSIRA OS NOVOS DADOS:

NOME: JOAQUIM

LOGRADOURO: RUA VINTE E DOIS

CIDADE: VIAMAO

ESTADO: RS

TELEFONE: 33333333

E-MAIL: JOAQUIM@VINTEDOIS.COM

CPF: 33333333333

Dados da pessoa fisica alterados com sucesso!

BUILD SUCCESSFUL (total time: 58 seconds)

|

```
// CONSULTAR TODOS DADOS DAS PF
PessoaFisicaDAO pessoaFisicaDAO = new PessoaFisicaDAO();

try {
    List<PessoaFisica> pessoas = pessoaFisicaDAO.getPessoas();

    System.out.println("DADOS DE PESSOAS FISICAS:");
    System.out.println();
    for (PessoaFisica pessoa : pessoas) {
        System.out.println("ID: " + pessoa.getId());
        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Logradouro: " + pessoa.getLogradouro());
        System.out.println("Cidade: " + pessoa.getCidade());
        System.out.println("Estado: " + pessoa.getEstado());
        System.out.println("Telefone: " + pessoa.getTelefone());
        System.out.println("Email: " + pessoa.getEmail());
        System.out.println("CPF: " + pessoa.getCpf());
        System.out.println();
    }
} catch (SQLException e) {
    System.err.println("Erro ao consultar pessoas fisica: " + e.getMessage());
}
}
```

run:

DADOS DE PESSOAS FISICAS:

ID: 1
 Nome: JOAQUIM
 Logradouro: RUA VINTE E DOIS
 Cidade: VIAMA0
 Estado: RS
 Telefone: 33333333
 Email: JOAQUIM@VINTEDOIS.COM
 CPF: 33333333333

ID: 3
 Nome: Avram Banks
 Logradouro: 3066 Mi Rd.
 Cidade: Ciudad Santa Catarina
 Estado: PR
 Telefone: 1423-2371
 Email: nunc@protonmail.ca
 CPF: 99999999999

ID: 5
 Nome: India Spencer
 Logradouro: 4332 Eget St.
 Cidade: Governador Valadares
 Estado: RN
 Telefone: 5253-5530
 Email: nisi.mauris@yahoo.ca
 CPF: 77777777777

ID: 7
Nome: Joao
Logradouro: Rua 12, casa 3
Cidade: Riacho do Sul
Estado: PA
Telefone: 1111-1111
Email: joao@riacho.com
CPF: 11111111111

ID: 8
Nome: Deacon Pickett
Logradouro: 7634 Curabatur Ave
Cidade: Alto del Carmen
Estado: RS
Telefone: 7511-7288
Email: donec.egestas@protonmail.com
CPF: 66666666666

ID: 10
Nome: Damian Sweet
Logradouro: Ap #657-9669 Ut, St.
Cidade: Nizhny
Estado: PR
Telefone: 5981-5724
Email: ac@hotmail.net
CPF: 55555555555

ID: 12
Nome: Astra Serrano
Logradouro: Ap #210-4769 Vivamus Rd.
Cidade: Lambayeque
Estado: RN
Telefone: 9537-8153
Email: tincidunt.vehicula.risus@google.couk
CPF: 44444444444

ID: 101
Nome: Joao
Logradouro: Rua Dois
Cidade: Porto Alegre
Estado: RS
Telefone: 999999999
Email: joao@estacio.com
CPF: 2222244444

ID: 102
Nome: Guiga
Logradouro: Rua Um
Cidade: Porto Alegre
Estado: RS
Telefone: 111111111
Email: guilherme@estacio.com
CPF: 11111111111

ID: 107
Nome: JULIANO
Logradouro: RUA NOVE
Cidade: PORTO ALEGRE
Estado: RS
Telefone: 999999999
Email: JULIANO@NOVE.COM
CPF: 23423423423

BUILD SUCCESSFUL (total time: 5 seconds)

1

```
// EXCLUIR PF
Scanner scanner = new Scanner(System.in);
PessoaFisicaDAO pessoaFisicaDAO = new PessoaFisicaDAO();

try {
    List<PessoaFisica> pessoas = pessoaFisicaDAO.getPessoas();
    System.out.println("DADOS DE PESSOAS FISICAS:");
    System.out.println();
    for (PessoaFisica pessoa : pessoas) {
        System.out.println("ID: " + pessoa.getId());
        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Logradouro: " + pessoa.getLogradouro());
        System.out.println("Cidade: " + pessoa.getCidade());
        System.out.println("Estado: " + pessoa.getEstado());
        System.out.println("Telefone: " + pessoa.getTelefone());
        System.out.println("Email: " + pessoa.getEmail());
        System.out.println("CPF: " + pessoa.getCpf());
        System.out.println();
    }

    System.out.println("Digite o ID da pessoa física que será excluída:");
    int idPessoa = scanner.nextInt();
    pessoaFisicaDAO.excluir(idPessoa);
    System.out.println("Pessoa física excluída com sucesso!");
} catch (SQLException e) {
    System.err.println("Erro ao excluir pessoa física: " + e.getMessage());
} finally {
    scanner.close();
}
}
```

run:

DADOS DE PESSOAS FISICAS:

ID: 1
 Nome: JOAQUIM
 Logradouro: RUA VINTE E DOIS
 Cidade: VIAMAO
 Estado: RS
 Telefone: 33333333
 Email: JOAQUIM@VINTEDOIS.COM
 CPF: 3333333333

ID: 3
 Nome: Avram Banks
 Logradouro: 3066 Mi Rd.
 Cidade: Ciudad Santa Catarina
 Estado: PR
 Telefone: 1423-2371
 Email: nunc@protonmail.ca
 CPF: 9999999999

ID: 5
 Nome: India Spencer
 Logradouro: 4332 Eget St.
 Cidade: Governador Valadares
 Estado: RN
 Telefone: 5253-5530
 Email: nisi.mauris@yahoo.ca
 CPF: 7777777777

ID: 8
Nome: Deacon Pickett
Logradouro: 7634 Curabitur Ave
Cidade: Alto del Carmen
Estado: RS
Telefone: 7511-7288
Email: donec.egestas@protonmail.com
CPF: 66666666666

ID: 10
Nome: Damian Sweet
Logradouro: Ap #657-9669 Ut, St.
Cidade: Nizhny
Estado: PR
Telefone: 5981-5724
Email: ac@hotmail.net
CPF: 55555555555

ID: 12
Nome: Astra Serrano
Logradouro: Ap #210-4769 Vivamus Rd.
Cidade: Lambayeque
Estado: RN
Telefone: 9537-8153
Email: tincidunt.vehicula.risus@google.couk
CPF: 44444444444

ID: 101
Nome: Joao
Logradouro: Rua Dois
Cidade: Porto Alegre
Estado: RS
Telefone: 999999999
Email: joao@estacio.com
CPF: 2222244444

ID: 102
Nome: Guiga
Logradouro: Rua Um
Cidade: Porto Alegre
Estado: RS
Telefone: 111111111
Email: guilherme@estacio.com
CPF: 11111111111

ID: 107
Nome: JULIANO
Logradouro: RUA NOVE
Cidade: PORTO ALEGRE
Estado: RS
Telefone: 999999999
Email: JULIANO@NOVE.COM
CPF: 23423423423

Digite o ID da pessoa física que será excluída:
7
Pessoa física excluída com sucesso!
BUILD SUCCESSFUL (total time: 32 seconds)

```
// CADASTRO PJ
Scanner scanner = new Scanner(System.in);

System.out.println("NOVA PESSOA JURIDICA: ");
System.out.print("NOME: ");
String nome = scanner.nextLine();
System.out.print("LOGRADOURO: ");
String logradouro = scanner.nextLine();
System.out.print("CIDADE: ");
String cidade = scanner.nextLine();
System.out.print("ESTADO: ");
String estado = scanner.nextLine();
System.out.print("TELEFONE: ");
String telefone = scanner.nextLine();
System.out.print("E-MAIL: ");
String email = scanner.nextLine();
System.out.print("CNPJ: ");
String cnpj = scanner.nextLine();

PessoaJuridica pessoa = new PessoaJuridica();
pessoa.setNome(nome);
pessoa.setLogradouro(logradouro);
pessoa.setCidade(cidade);
pessoa.setEstado(estado);
pessoa.setTelefone(telefone);
pessoa.setEmail(email);
pessoa.setCnpj(cnpj);

PessoaJuridicaDAO pessoaJuridicaDAO = new PessoaJuridicaDAO();
```

```
try {
    pessoaJuridicaDAO.incluir(pessoa);
    System.out.println("Pessoa juridica incluida com sucesso!");
} catch (SQLException e) {
    System.err.println("Erro ao incluir pessoa juridica: " + e.getMessage());
} finally {
    scanner.close();
}
```

run:

```
NOVA PESSOA JURIDICA:
NOME: BERNARDES LTDA
LOGRADOURO: RUA TREZE
CIDADE: PORTO ALEGRE
ESTADO: RS
TELEFONE: 999779977
E-MAIL: BERNARDES@LTDA.COM
CNPJ: 89899898889989
Pessoa juridica incluida com sucesso!
BUILD SUCCESSFUL (total time: 1 minute 21 seconds)
```

```
// ALTERAR DADOS PJ
Scanner scanner = new Scanner(System.in);
System.out.println("ALTERAR DADOS DE PESSOA JURIDICA: ");
System.out.println();
System.out.println("ID da pessoa juridica que será alterada: ");

int id = scanner.nextInt();
scanner.nextLine();

PessoaJuridicaDAO pessoaJuridicaDAO = new PessoaJuridicaDAO();

try {
    PessoaJuridica pessoa = pessoaJuridicaDAO.getPessoa(id);

    if (pessoa != null) {
        System.out.println("DADOS ATUAIS: ");
        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Logradouro: " + pessoa.getLogradouro());
        System.out.println("Cidade: " + pessoa.getCidade());
        System.out.println("Estado: " + pessoa.getEstado());
        System.out.println("Telefone: " + pessoa.getTelefone());
        System.out.println("Email: " + pessoa.getEmail());
        System.out.println("CNPJ: " + pessoa.getCnpj());

        System.out.println();
    }
}
```

```
System.out.println("INSIRA OS NOVOS DADOS: ");
System.out.print("NOME: ");
String nome = scanner.nextLine();
System.out.print("LOGRADOURO: ");
String logradouro = scanner.nextLine();
System.out.print("CIDADE: ");
String cidade = scanner.nextLine();
System.out.print("ESTADO: ");
String estado = scanner.nextLine();
System.out.print("TELEFONE: ");
String telefone = scanner.nextLine();
System.out.print("E-MAIL: ");
String email = scanner.nextLine();
System.out.print("CNPJ: ");
String cnpj = scanner.nextLine();

pessoa.setNome(nome);
pessoa.setLogradouro(logradouro);
pessoa.setCidade(cidade);
pessoa.setEstado(estado);
pessoa.setTelefone(telefone);
pessoa.setEmail(email);
pessoa.setCnpj(cnpj);

pessoaJuridicaDAO.alterar(pessoa);
System.out.println("Dados da pessoa juridica alterado com sucesso!");
} else {
    System.out.println("Nenhuma pessoa juridica encontrada com esse ID.");
}
}
```

```

    } catch (SQLException e) {
        System.err.println("Erro ao alterar pessoa juridica: " + e.getMessage());
    } finally {
        scanner.close();
    }
}

```

```

run:
ALTERAR DADOS DE PESSOA JURIDICA:

ID da pessoa juridica que será alterada:
4
DADOS ATUAIS:
Nome: Allen Flowers
Logradouro: 5075 Malesuada Road
Cidade: Uman
Estado: GO
Telefone: 3771-8832
Email: fringilla.purus@yahoo.couk
CNPJ: 44444444444444

INSIRA OS NOVOS DADOS:
NOME: JULIS LTDA
LOGRADOURO: RUA NINE
CIDADE: VIAMAO
ESTADO: RS
TELEFONE: 55554444
E-MAIL: JULIS@LTDA
CNPJ: 9988777666655
Dados da pessoa juridica alterado com sucesso!
BUILD SUCCESSFUL (total time: 2 minutes 13 seconds)

```

```

// CONSULTAR TODOS DADOS DAS PF
PessoaJuridicaDAO pessoaJuridicaDAO = new PessoaJuridicaDAO();

try {
    List<PessoaJuridica> pessoas = pessoaJuridicaDAO.getPessoas();

    System.out.println("DADOS DE PESSOAS JURIDICA:");
    System.out.println();
    for (PessoaJuridica pessoa : pessoas) {
        System.out.println("ID: " + pessoa.getId());
        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Logradouro: " + pessoa.getLogradouro());
        System.out.println("Cidade: " + pessoa.getCidade());
        System.out.println("Estado: " + pessoa.getEstado());
        System.out.println("Telefone: " + pessoa.getTelefone());
        System.out.println("Email: " + pessoa.getEmail());
        System.out.println("CNPJ: " + pessoa.getCnpj());
        System.out.println();
    }
} catch (SQLException e) {
    System.err.println("Erro ao consultar pessoas juridicas: " + e.getMessage());
}

```

run:
DADOS DE PESSOAS JURIDICA:

ID: 2
Nome: JOCA LTDA
Logradouro: RUA JO
Cidade: POA
Estado: RS
Telefone: 323232322
Email: JOCA
CNPJ: 22333232232323

ID: 4
Nome: JULIS LTDA
Logradouro: RUA NINE
Cidade: VIAMAO
Estado: RS
Telefone: 55554444
Email: JULIS@LTDA
CNPJ: 99887776666655

ID: 6
Nome: Tanek Roy
Logradouro: 3862 Ut, Road
Cidade: Christchurch
Estado: RJ
Telefone: 3242-6301
Email: eget@hotmail.edu
CNPJ: 66666666666666

ID: 9
Nome: Meredith Gaines
Logradouro: P.O. Box 867, 5786ELIT. Ave
Cidade: Melton
Estado: SC
Telefone: 6904-7307
Email: orci@yahoo.org
CNPJ: 88888888888888

ID: 11
Nome: Latifah Conway
Logradouro: Ap #628-8354 Auctor Av.
Cidade: Sokoto
Estado: GO
Telefone: 1224-6424
Email: consectetuer.adipiscing@icloud.com
CNPJ: 99999999999999

ID: 108
Nome: BERNARDES LTDA
Logradouro: RUA TREZE
Cidade: PORTO ALEGRE
Estado: RS
Telefone: 999779977
Email: BERNARDES@LTDA.COM
CNPJ: 89899898889989

BUILD SUCCESSFUL (total time: 5 seconds)
|

```
// EXCLUIR PJ
Scanner scanner = new Scanner(System.in);
PessoaJuridicaDAO pessoaJuridicaDAO = new PessoaJuridicaDAO();

try {
    List<PessoaJuridica> pessoas = pessoaJuridicaDAO.getPessoas();
    System.out.println("DADOS DE PESSOAS JURIDICAS:");
    System.out.println();
    for (PessoaJuridica pessoa : pessoas) {
        System.out.println("ID: " + pessoa.getId());
        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Logradouro: " + pessoa.getLogradouro());
        System.out.println("Cidade: " + pessoa.getCidade());
        System.out.println("Estado: " + pessoa.getEstado());
        System.out.println("Telefone: " + pessoa.getTelefone());
        System.out.println("Email: " + pessoa.getEmail());
        System.out.println("CNPJ: " + pessoa.getCnpj());
        System.out.println();
    }

    System.out.println("Digite o ID da pessoa juridica que será excluído: ");
    int idPessoa = scanner.nextInt();
    pessoaJuridicaDAO.excluir(idPessoa);
    System.out.println("Pessoa juridica excluída com sucesso!");
} catch (SQLException e) {
    System.err.println("Erro ao excluir pessoa juridica: " + e.getMessage());
} finally {
    scanner.close();
}
}
```

run:

DADOS DE PESSOAS JURIDICAS:

ID: 2
Nome: JOCA LTDA
Logradouro: RUA JO
Cidade: POA
Estado: RS
Telefone: 323232322
Email: JOCA
CNPJ: 22333232232323

ID: 4
Nome: JULIS LTDA
Logradouro: RUA NINE
Cidade: VIAMA0
Estado: RS
Telefone: 55554444
Email: JULIS@LTDA
CNPJ: 99887776666655

ID: 6
Nome: Tanek Roy
Logradouro: 3862 Ut, Road
Cidade: Christchurch
Estado: RJ
Telefone: 3242-6301
Email: eget@hotmail.edu
CNPJ: 66666666666666

ID: 9
Nome: Meredith Gaines
Logradouro: P.O. Box 867, 5786 Elit. Ave
Cidade: Melton
Estado: SC
Telefone: 6904-7307
Email: orci@yahoo.org
CNPJ: 88888888888888

ID: 11
Nome: Latifah Conway
Logradouro: Ap #628-8354 Auctor Av.
Cidade: Sokoto
Estado: GO
Telefone: 1224-6424
Email: consectetuer.adipiscing@icloud.com
CNPJ: 99999999999999

ID: 108
Nome: BERNARDES LTDA
Logradouro: RUA TREZE
Cidade: PORTO ALEGRE
Estado: RS
Telefone: 999779977
Email: BERNARDES@LTDA.COM
CNPJ: 89899898889989

Digite o ID da pessoa juridica que será excluido:
11
Pessoa juridica excluída com sucesso!
BUILD SUCCESSFUL (total time: 15 seconds)

2º Procedimento

CÓDIGOS

```
package cadastrobd;

import cadastro.model.PessoaFisicaDAO;
import cadastro.model.PessoaJuridicaDAO;
import cadastrobd.model.Pessoa;
import cadastrobd.model.PessoaFisica;
import cadastrobd.model.PessoaJuridica;
import java.util.Scanner;
import java.sql.SQLException;
import java.util.List;

public class CadastroBD {
    public static void main(String[] args) throws SQLException {
        int opcao;
        PessoaFisicaDAO pessoaFisicaDAO = new PessoaFisicaDAO();
        PessoaJuridicaDAO pessoaJuridicaDAO = new PessoaJuridicaDAO();
        Scanner scanner = new Scanner(System.in);
        String tipoPessoa;

        do {
            System.out.println("=====");
            System.out.println("1 - Incluir Pessoa");
            System.out.println("2 - Alterar Pessoa");
            System.out.println("3 - Excluir Pessoa");
            System.out.println("4 - Buscar pelo Id");
            System.out.println("5 - Exibir Todos");
            System.out.println("0 - Finalizar Programa");
            System.out.println("=====");
            opcao = scanner.nextInt();
            scanner.nextLine();

            switch (opcao) {
                case 1:
                    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
                    tipoPessoa = scanner.nextLine().toUpperCase();

                    if (tipoPessoa.equals("F")) {
                        System.out.println("Insira os dados: ");
                        System.out.print("NOME: ");
                        String nome = scanner.nextLine();
                        System.out.print("CPF: ");
                        String cpf = scanner.nextLine();
                        System.out.print("LOGRADOURO: ");
                        String logradouro = scanner.nextLine();
                        System.out.print("CIDADE: ");
                        String cidade = scanner.nextLine();
                        System.out.print("ESTADO: ");
                        String estado = scanner.nextLine();
                        System.out.print("TELEFONE: ");
                        String telefone = scanner.nextLine();
                        System.out.print("E-MAIL: ");
                        String email = scanner.nextLine();

                        PessoaFisica pessoaFisica = new PessoaFisica(00, nome, logradouro, cidade, estado, telefone, email, cpf);

                        try {
                            pessoaFisicaDAO.incluir(pessoaFisica);
                        } catch (SQLException e) {
                            System.out.println("Erro ao incluir: " + e.getMessage());
                        }
                    }
                }
            }
        } while (opcao != 0);
    }
}
```

```

    } else if (tipoPessoa.equals("J")) {
        System.out.println("Insira os dados: ");
        System.out.print("NOME: ");
        String nome = scanner.nextLine();
        System.out.print("CNPJ: ");
        String cnpj = scanner.nextLine();
        System.out.print("LOGRADOURO: ");
        String logradouro = scanner.nextLine();
        System.out.print("CIDADE: ");
        String cidade = scanner.nextLine();
        System.out.print("ESTADO: ");
        String estado = scanner.nextLine();
        System.out.print("TELEFONE: ");
        String telefone = scanner.nextLine();
        System.out.print("E-MAIL: ");
        String email = scanner.nextLine();

        PessoaJuridica pessoaJuridica = new PessoaJuridica(00, nome, logradouro, cidade, estado, telefone, email, cnpj);

        try {
            pessoaJuridicaDAO.incluir(pessoaJuridica);
        } catch (SQLException e) {
            System.out.println("Erro ao incluir: " + e.getMessage());
        }
    }
    break;
case 2:
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    tipoPessoa = scanner.nextLine().toUpperCase();

    System.out.println("Insira o Id que será alterado: ");
    int id = scanner.nextInt();
    scanner.nextLine();

```

```

    if (tipoPessoa.equals("F")) {
        System.out.println("INSIRA OS NOVOS DADOS: ");
        System.out.print("NOME: ");
        String nome = scanner.nextLine();
        System.out.print("CPF: ");
        String cpf = scanner.nextLine();
        System.out.print("LOGRADOURO: ");
        String logradouro = scanner.nextLine();
        System.out.print("CIDADE: ");
        String cidade = scanner.nextLine();
        System.out.print("ESTADO: ");
        String estado = scanner.nextLine();
        System.out.print("TELEFONE: ");
        String telefone = scanner.nextLine();
        System.out.print("E-MAIL: ");
        String email = scanner.nextLine();
        PessoaFisica pessoaFisica = pessoaFisicaDAO.getPessoa(id);

        try {
            pessoaFisica.setNome(nome);
            pessoaFisica.setCpf(cpf);
            pessoaFisica.setLogradouro(logradouro);
            pessoaFisica.setCidade(cidade);
            pessoaFisica.setEstado(estado);
            pessoaFisica.setTelefone(telefone);
            pessoaFisica.setEmail(email);
            pessoaFisicaDAO.alterar(pessoaFisica);
        } catch (SQLException e) {
            System.out.println("Erro ao alterar: " + e.getMessage());
        }
    }

```

```

    } else if (tipoPessoa.equals("J")) {
        System.out.print("NOME: ");
        String nome = scanner.nextLine();
        System.out.print("CNPJ: ");
        String cnpj = scanner.nextLine();
        System.out.print("LOGRADOURO: ");
        String logradouro = scanner.nextLine();
        System.out.print("CIDADE: ");
        String cidade = scanner.nextLine();
        System.out.print("ESTADO: ");
        String estado = scanner.nextLine();
        System.out.print("TELEFONE: ");
        String telefone = scanner.nextLine();
        System.out.print("E-MAIL: ");
        String email = scanner.nextLine();
        PessoaJuridica pessoaJuridica = pessoaJuridicaDAO.getPessoa(id);

        try {
            pessoaJuridica.setNome(nome);
            pessoaJuridica.setCnpj(cnpj);
            pessoaJuridica.setLogradouro(logradouro);
            pessoaJuridica.setCidade(cidade);
            pessoaJuridica.setEstado(estado);
            pessoaJuridica.setTelefone(telefone);
            pessoaJuridica.setEmail(email);
            pessoaJuridicaDAO.alterar(pessoaJuridica);
        } catch (SQLException e) {
            System.out.println("Erro ao alterar: " + e.getMessage());
        }
    }
}
break;

```

```

        break;
    case 3:
        System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
        tipoPessoa = scanner.nextLine().toUpperCase();

        System.out.println("Insira o ID da pessoa que será excluído: ");
        int idExcluir = scanner.nextInt();
        scanner.nextLine();

        if (tipoPessoa.equals("F")) {
            try {
                pessoaFisicaDAO.excluir(idExcluir);
                System.out.println("Pessoa fisica excluida com sucesso.");
            } catch (SQLException e) {
                System.out.println("Erro ao excluir: " + e.getMessage());
            }
        } else if (tipoPessoa.equals("J")) {
            try {
                pessoaJuridicaDAO.excluir(idExcluir);
                System.out.println("Pessoa juridica excluida com sucesso.");
            } catch (SQLException e) {
                System.out.println("Erro ao excluir: " + e.getMessage());
            }
        }
        break;
    case 4:
        System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
        tipoPessoa = scanner.nextLine().toUpperCase();

        System.out.println("Insira o ID da pessoa que será buscada: ");
        int idBuscar = scanner.nextInt();
        scanner.nextLine();

```

```

        if (tipoPessoa.equals("F")) {
            try {
                PessoaFisica pessoaFisica = pessoaFisicaDAO.getPessoa(idBuscar);
                if (pessoaFisica != null) {
                    System.out.println("Dados da Pessoa Fisica: ");
                    System.out.println("ID: " + pessoaFisica.getId());
                    System.out.println("NOME: " + pessoaFisica.getNome());
                    System.out.println("CPF: " + pessoaFisica.getCpf());
                    System.out.println("LOGRADOURO: " + pessoaFisica.getLogradouro());
                    System.out.println("CIDADE: " + pessoaFisica.getCidade());
                    System.out.println("ESTADO: " + pessoaFisica.getEstado());
                    System.out.println("TELEFONE: " + pessoaFisica.getTelefone());
                    System.out.println("E-MAIL: " + pessoaFisica.getEmail());
                } else {
                    System.out.println("Pessoa fisica não encontrada.");
                }
            } catch (SQLException e) {
                System.out.println("Erro ao buscar pessoa fisica: " + e.getMessage());
            }
        } else if (tipoPessoa.equals("J")) {
            try {
                PessoaJuridica pessoaJuridica = pessoaJuridicaDAO.getPessoa(idBuscar);
                if (pessoaJuridica != null) {
                    System.out.println("Dados da Pessoa Jurídica:");
                    System.out.println("ID: " + pessoaJuridica.getId());
                    System.out.println("NOME: " + pessoaJuridica.getNome());
                    System.out.println("CNPJ: " + pessoaJuridica.getCnpj());
                    System.out.println("LOGRADOURO: " + pessoaJuridica.getLogradouro());
                    System.out.println("CIDADE: " + pessoaJuridica.getCidade());
                    System.out.println("ESTADO: " + pessoaJuridica.getEstado());
                    System.out.println("TELEFONE: " + pessoaJuridica.getTelefone());
                    System.out.println("E-MAIL: " + pessoaJuridica.getEmail());
                }
            }
        }
    }
}

```

```

        } else {
            System.out.println("Pessoa juridica não encontrada.");
        }
    } catch (SQLException e) {
        System.out.println("Erro ao buscar pessoa juridica: " + e.getMessage());
    }
}
break;
case 5:
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    tipoPessoa = scanner.nextLine().toUpperCase();

    if (tipoPessoa.equals("F")) {
        try {
            List<PessoaFisica> pessoasFisicas = pessoaFisicaDAO.getPessoas();
            if (!pessoasFisicas.isEmpty()) {
                System.out.println("Dados de todas as Pessoas Fisicas:");
                for (PessoaFisica pessoa : pessoasFisicas) {
                    System.out.println("ID: " + pessoa.getId());
                    System.out.println("Nome: " + pessoa.getNome());
                    System.out.println("CPF: " + pessoa.getCpf());
                    System.out.println("Logradouro: " + pessoa.getLogradouro());
                    System.out.println("Cidade: " + pessoa.getCidade());
                    System.out.println("Estado: " + pessoa.getEstado());
                    System.out.println("Telefone: " + pessoa.getTelefone());
                    System.out.println("Email: " + pessoa.getEmail());
                    System.out.println();
                }
            } else {
                System.out.println("Nenhuma pessoa fisica encontrada.");
            }
        } catch (SQLException e) {
            System.out.println("Erro ao buscar pessoas fisicas: " + e.getMessage());
        }
    }
}

```

```

    } else if (tipoPessoa.equals("J")) {
        try {
            List<PessoaJuridica> pessoasJuridicas = pessoaJuridicaDAO.getPessoas();
            if (!pessoasJuridicas.isEmpty()) {
                System.out.println("Dados de todas as Pessoas Jurídicas:");
                for (PessoaJuridica pessoa : pessoasJuridicas) {
                    System.out.println("ID: " + pessoa.getId());
                    System.out.println("Nome: " + pessoa.getNome());
                    System.out.println("CNPJ: " + pessoa.getCnpj());
                    System.out.println("Logradouro: " + pessoa.getLogradouro());
                    System.out.println("Cidade: " + pessoa.getCidade());
                    System.out.println("Estado: " + pessoa.getEstado());
                    System.out.println("Telefone: " + pessoa.getTelefone());
                    System.out.println("Email: " + pessoa.getEmail());
                    System.out.println();
                }
            } else {
                System.out.println("Nenhuma pessoa juridica encontrada.");
            }
        } catch (SQLException e) {
            System.out.println("Erro ao buscar pessoas juridicas: " + e.getMessage());
        }
        break;
    case 0:
        System.out.println("Programa encerrado!");
        break;
    default:
        System.out.println("Opção inválida. Tente novamente.");
    }
} while (opcao != 0);
}

```

RESULTADO

run:

```
=====
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa
=====
1
F - Pessoa Fisica | J - Pessoa Juridica
f
Insira os dados:
NOME: KATHIA
CPF: 12312345677
LOGRADOURO: RUA NOVE
CIDADE: PORTO ALEGRE
ESTADO: RS
TELEFONE: 21551252
E-MAIL: KATHIA@NOVE.COM
```

```
=====
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa
=====
2
F - Pessoa Fisica | J - Pessoa Juridica
F
Insira o Id que será alterado:
1
INSIRA OS NOVOS DADOS:
NOME: NICOLAS
CPF: 87654387654
LOGRADOURO: RUA TRINTA E DOIS
CIDADE: VIAMAO
ESTADO: RS
TELEFONE: 87655678
E-MAIL: NICOLAS@TRINTADOIS.COM
```

```
=====
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa
=====
1
F - Pessoa Fisica | J - Pessoa Juridica
J
Insira os dados:
NOME: KATHIA LTDA
CNPJ: 32132132132132
LOGRADOURO: RUA DEZ
CIDADE: PORTO ALEGRE
ESTADO: RS
TELEFONE: 12345678
E-MAIL: KATHIA
```

```
=====
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa
=====
2
F - Pessoa Fisica | J - Pessoa Juridica
J
Insira o Id que será alterado:
2
NOME: NICOLAS LTDA
CNPJ: 76543876543876
LOGRADOURO: RUA TWO
CIDADE: GRAVATAI
ESTADO: RS
TELEFONE: 87658765
E-MAIL: NICOLAS@LTDA.COM
```

=====

1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa

=====

3
F - Pessoa Fisica | J - Pessoa Juridica
F
Insira o ID da pessoa que será excluido:
12
Pessoa fisica excluida com sucesso.

=====

1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa

=====

3
F - Pessoa Fisica | J - Pessoa Juridica
j
Insira o ID da pessoa que será excluido:
9
Pessoa juridica excluida com sucesso.

=====

1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa

=====

4
F - Pessoa Fisica | J - Pessoa Juridica
j
Insira o ID da pessoa que será buscada:
110
Dados da Pessoa Jurídica:
ID: 110
NOME: KATHIA LTDA
CNPJ: 32132132132132
LOGRADOURO: RUA DEZ
CIDADE: PORTO ALEGRE
ESTADO: RS
TELEFONE: 12345678
E-MAIL: KATHIA

=====

1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa

=====

4
F - Pessoa Fisica | J - Pessoa Juridica
f
Insira o ID da pessoa que será buscada:
109
Dados da Pessoa Fisica:
ID: 109
NOME: KATHIA
CPF: 12312345677
LOGRADOURO: RUA NOVE
CIDADE: PORTO ALEGRE
ESTADO: RS
TELEFONE: 21551252
E-MAIL: KATHIA@NOVE.COM

=====

1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa

=====

5
F - Pessoa Fisica | J - Pessoa Juridica
f
Dados de todas as Pessoas Fisicas:
ID: 1
Nome: NICOLAS
CPF: 87654387654
Logradouro: RUA TRINTA E DOIS
Cidade: VIAMAO
Estado: RS
Telefone: 87655678
Email: NICOLAS@TRINTADOIS.COM

ID: 3
Nome: Avram Banks
CPF: 99999999999
Logradouro: 3066 Mi Rd.
Cidade: Ciudad Santa Catarina
Estado: PR
Telefone: 1423-2371
Email: nunc@protonmail.ca

ID: 5
Nome: India Spencer
CPF: 77777777777
Logradouro: 4332 Eget St.
Cidade: Governador Valadares
Estado: RN
Telefone: 5253-5530
Email: nisi.mauris@yahoo.ca

ID: 8
Nome: Deacon Pickett
CPF: 66666666666
Logradouro: 7634 Curabatur Ave
Cidade: Alto del Carmen
Estado: RS
Telefone: 7511-7288
Email: donec.egestas@protonmail.com

ID: 10
Nome: Damian Sweet
CPF: 55555555555
Logradouro: Ap #657-9669 Ut, St.
Cidade: Nizhny
Estado: PR
Telefone: 5981-5724
Email: ac@hotmail.net

ID: 109
Nome: KATHIA
CPF: 12312345677
Logradouro: RUA NOVE
Cidade: PORTO ALEGRE
Estado: RS
Telefone: 21551252
Email: KATHIA@NOVE.COM

=====

1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa

=====

5
F - Pessoa Fisica | J - Pessoa Juridica
j
Dados de todas as Pessoas Jurídicas:
ID: 2
Nome: NICOLAS LTDA
CNPJ: 76543876543876
Logradouro: RUA TWO
Cidade: GRAVATAI
Estado: RS
Telefone: 87658765
Email: NICOLAS@LTDA.COM

ID: 110
Nome: KATHIA LTDA
CNPJ: 32132132132132
Logradouro: RUA DEZ
Cidade: PORTO ALEGRE
Estado: RS
Telefone: 12345678
Email: KATHIA

=====

1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Buscar pelo Id
5 - Exibir Todos
0 - Finalizar Programa

=====

0
Programa encerrado!
BUILD SUCCESSFUL (total time: 7 minutes 21 seconds)

ID: 101
Nome: Joao
CPF: 2222244444
Logradouro: Rua Dois
Cidade: Porto Alegre
Estado: RS
Telefone: 999999999
Email: joao@estacio.com

ID: 102
Nome: Guiga
CPF: 11111111111
Logradouro: Rua Um
Cidade: Porto Alegre
Estado: RS
Telefone: 111111111
Email: guilherme@estacio.com

ID: 107
Nome: JULIANO
CPF: 23423423423
Logradouro: RUA NOVE
Cidade: PORTO ALEGRE
Estado: RS
Telefone: 999999999
Email: JULIANO@NOVE.COM

ID: 4
Nome: JULIS LTDA
CNPJ: 9988777666655
Logradouro: RUA NINE
Cidade: VIAMAO
Estado: RS
Telefone: 5554444
Email: JULIS@LTDA

ID: 6
Nome: Tanek Roy
CNPJ: 66666666666666
Logradouro: 3862 Ut, Road
Cidade: Christchurch
Estado: RJ
Telefone: 3242-6301
Email: eget@hotmail.edu

ID: 108
Nome: BERNARDES LTDA
CNPJ: 8989898889989
Logradouro: RUA TREZE
Cidade: PORTO ALEGRE
Estado: RS
Telefone: 999779977
Email: BERNARDES@LTDA.COM

ANÁLISE E CONCLUSÃO

Procedimento 1

- a. Qual a importância dos componentes de middleware, como o JDBC?

São importantes por simplificar o acesso e manipulação de dados, fornecendo camadas de abstração entre o código e o banco. Facilitam a conectividade com o banco de dados, otimiza desempenho de consultas, garante segurança dos dados, dentre outras...

- b. Qual a diferença no uso de Statement ou PreparedStatement para a manipulação de dados?

Statement se usa para executar consultas estáticas sem parâmetros. Elas são enviadas ao banco como são escritas no código. Já o PreparedStatement é mais usado em consultas dinâmicas com parâmetros pois permite a inserção de valores nos comando SQL.

- c. Como o padrão DAO melhora a manutenibilidade do Software?

Ajuda a tornar mais fácil cuidar de um programa ao organizar a forma como ele acessa os dados, separando o código que lida com o banco de dados ou outras fontes do resto do programa. Além disso ele facilita a realização dos testes, garantindo que tudo funcione direitinho.

- d. Como a herança é refletida no banco de dados, quando lidamos com um modelo estritamente relacional?

Ela reflete de forma que herda características principais de um todo de forma eficiente, mostrando a relação entre elas. Sendo assim, cada classe herda o principal e é adicionado apenas características próprias daqueles elementos, sem ter que reescrever todo código novamente.

Procedimento 2

- a. Quais as diferenças entre a persistência em arquivo e a persistência em banco de dados?

As duas são formas de guardar informações. Quando se usa persistência em arquivo, os dados são guardados em documentos, como arquivos de textos simples, dentro de um computador. Já quando é usado a persistência em um banco de dados, eles são armazenados dentro de um sistema organizado e eficiente.

- b. Como o uso de operador lambda simplificou a impressão dos valores nas entidades, nas versões mais recente do Java?

Simplificou quando podemos definir funções simples em tempo real, que economizaram em um tempo considerável e permitiu ir além de funções normais.

- c. Por que métodos acionados diretamente pelo método main, sem o uso de um objeto, precisam ser marcados como static?

Porque o método main é estático e não possui acesso a instancias de objetos. A marcação dele como static permite que esses métodos sejam chamados independente de objetos e simplificam sua chamada direta pelo main.