



Estácio

Polo Tijuca

Aluno	Luiz Felipe Sarmiento Bonet
Matrícula	2023.09.34276-6
Curso	Desenvolvimento Full Stack
Disciplina	RPG0016 – Back-end Sem Banco Não Tem
Turma	9001
Período	2024 – 2º Semestre

Missão Prática: Mundo 3 - Nível 3

Título da Prática 1: Mapeamento Objeto-Relacional e DAO

Título da Prática 2: Alimentando a Base

Objetivos:

- Implementar persistência com base no middleware JDBC;
- Utilizar o padrão DAO (Data Access Object) no manuseio de dados;
- Implementar o mapeamento objeto-relacional em sistemas Java;
- Criar sistemas cadastrais com persistência em banco relacional;
- Criar um aplicativo cadastral com uso do SQL Server na persistência de dados.

Resultados de Execução – Prática 1

```
run:
A Pessoa Fisica foi incluída com sucesso
ID: 5
Nome: Vinicius Rocha Lima
Endereco: Travessa Sao Joao, 186
Cidade: Rio de Janeiro
Estado: Rio de Janeiro
E-mail: ViniciusRochaLima@dayrep.com
Telefone: 21 6391-4762
CPF: 127.460.858-95
Pessoa Fisica alterada com sucesso.
ID: 5
Nome: Vinicius Rocha Lima
Endereco: Rua Armandino Gonzaga, 386
Cidade: Florianopolis
Estado: Santa Catarina
E-mail: ViniciusRochaLima@dayrep.com
Telefone: 21 6391-4762
CPF: 127.460.858-95
Exibindo todas as Pessoas Fisicas:
ID: 1
Nome: Joao Fernandes Correia
Endereco: 365.410.330-80
Cidade: Rua Jose Aristides
Estado: Divinopolis
E-mail: Minas Gerais
Telefone: JoaoFernandesCorreia@rhyta.com
CPF: 37 4647-9323
ID: 2
Nome: Kaua Azevedo Silva
Endereco: 78349632011
Cidade: Rua Faustino Siqueira Franco
Estado: Guarulhos
E-mail: Sao Paulo
Telefone: KauaAzevedoSilva@armyspy.com
CPF: 11 4681-5575
ID: 5
Nome: Vinicius Rocha Lima
Endereco: 127.460.858-95
Cidade: Rua Armandino Gonzaga, 386
Estado: Florianopolis
E-mail: Santa Catarina
Telefone: ViniciusRochaLima@dayrep.com
CPF: 21 6391-4762
Pessoa Fisica excluída com sucesso
A Pessoa Juridica foi incluída com sucesso
ID: 6

Pessoa Fisica excluída com sucesso
A Pessoa Juridica foi incluída com sucesso
ID: 6
Nome: Patterson-Fletcher
Endereco: Travessa das Cravinas, 103
Cidade: Salvador
Estado: Bahia
E-mail: assistenciaPFjourrapide.com
Telefone: 71 8036-9442
CNPJ: 81.669.950/0001-05
Pessoa Juridica Alterada com sucesso.
ID: 6
Nome: Fletcher Enterprises
Endereco: Travessa das Cravinas, 103
Cidade: Salvador
Estado: Bahia
E-mail: Helpdesk.FletcherEnterprises@domain.com
Telefone: 71 8036-9442
CNPJ: 81.669.950/0001-05
Exibindo toas as Pessoas Juridicas:
ID: 3
Nome: Heilig-Meyers
Endereco: 95.823.031/0001-59
Cidade: Estrada do Recanto 161
Estado: Timoteo
E-mail: Minas Gerais
Telefone: Heilig-Meyers@teleworm.us
CNPJ: 31 5667-6663
ID: 4
Nome: Grupo Tecnico Jardins
Endereco: 67652009000158
Cidade: Praca Presidente Gamal Abdel Nasser 1403
Estado: Sao Paulo
E-mail: Sao Paulo
Telefone: Henceall@dayrep.com
CNPJ: 11 8783-6448
ID: 6
Nome: Fletcher Enterprises
Endereco: 81.669.950/0001-05
Cidade: Travessa das Cravinas, 103
Estado: Salvador
E-mail: Bahia
Telefone: Helpdesk.FletcherEnterprises@domain.com
CNPJ: 71 8036-9442
Pessoa Juridica excluída com sucesso.
BUILD SUCCESSFUL (total time: 6 seconds)
```

Figura 1. Visualização do output de CadastroDBTeste.java. A imagem esquerda mostra a parte do output referente à pessoa física e a imagem direita, à pessoa jurídica.

Resultados de Execução – Prática 2

The figure shows two side-by-side screenshots of a Java application running in NetBeans. Both windows show the 'Output - CadastroBD (run)' console.

Left Screenshot (Physical Person):

```
run:
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
1
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o proximo ID: 5
Digite o Nome: Vinicius Fernandes
Digite o Endereco: Rua do Oitizeiro 145
Digite a Cidade: Serra
Digite o Estado: Espirito Santo
Digite o E-mail: ViniciusFernandesSantos@dayrep.com
Digite o Telefone: 27 77537638
Digite o CPF: 304.441.749-07
Pessoa Fisica cadastrada com sucesso.
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
0
Encerrando o programa...
BUILD SUCCESSFUL (total time: 9 minutes 18 seconds)
```

Right Screenshot (Legal Person):

```
run:
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
1
F - Pessoa Fisica | J - Pessoa Juridica
J
Digite o proximo ID: 6
Digite o Nome: ConnectList
Digite o Endereco: Rua Adelino Torquato 179
Digite a Cidade: Mogi das Cruzes
Digite o Estado: Sao Paulo
Digite o E-mail: CtlT.Helpdesk@teleworm.br
Digite o Telefone: 11 99829944
Digite o CNPJ: 86.360.676/0001-01
Pessoa Juridica cadastrada com sucesso.
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
0
Encerrando o programa...
BUILD SUCCESSFUL (total time: 2 minutes 0 seconds)
```

Figura 2. Inclusão de novas pessoas física (esquerda) e jurídica (direita). Observe também como o programa é encerrado com sucesso, dado o comando adequado.

The figure shows two screenshots of a SQL query result in NetBeans. The query is 'SELECT TOP 100 * FROM dbo...'. The first screenshot shows a list of 6 rows, and the second screenshot shows a list of 3 rows.

Top Screenshot (6 rows):

#	idP	Nome	Endereco	Cidade	Estado	Email	Telefone
1	1	Joao Fernandes Correia	Rua Jose Aristides	Divinopolis	Minas Gerais	JoaoFernandesCorreia@rhyta.com	37 4647-9323
2	2	Kaua Azevedo Silva	Rua Faustino Siqueira Franco	Guarulhos	Sao Paulo	KauaAzevedoSilva@armyspy.com	11 4681-5575
3	3	Heilig-Meyers	Estrada do Recanto 161	Timoteo	Minas Gerais	Heilig-Meyers@teleworm.us	31 5667-6663
4	4	Grupo Técnico Jardins	Praça Presidente Gamal Abdel Nasser 1403	Sao Paulo	Sao Paulo	Henceall@dayrep.com	11 8783-6448
5	5	Vinicius Fernandes	Rua do Oitizeiro 145	Serra	Espirito Santo	ViniciusFernandesSantos@dayrep.com	27 77537638
6	6	ConnectList	Rua Adelino Torquato 179	Mogi das Cruzes	Sao Paulo	CtlT.Helpdesk@teleworm.br	11 99829944

Bottom Screenshot (3 rows):

#	FK_Pessoa_id	CPF
1	1	365.410.330-80
2	2	78349632011
3	5	304.441.749-07

#	FK_Pessoa_id	CNPJ
1	3	95.823.031/0001-59
2	4	67652009000158
3	6	86.360.676/0001-01

Figura 3. Imagens do query no NetBeans mostrando que as pessoas foram cadastradas no banco de dados.

```
run:
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
4
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o ID da Pessoa Fisica: 5
-- Exibindo cadastro de Pessoa Fisica --
ID: 5
Nome: Vinicius Fernandes
Endereco: Rua do Oitizeiro 145
Cidade: Serra
Estado: Espirito Santo
E-mail: ViniciusFernandesSantos@dayrep.com
Telefone: 27 77537638
CPF: 304.441.749-07
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
```

```
run:
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
4
F - Pessoa Fisica | J - Pessoa Juridica
J
Digite o ID da Pessoa Juridica: 6
-- Exibindo cadastro de Pessoa Juridica --
ID: 6
Nome: ConnectList
Endereco: Rua Adelino Torquato 179
Cidade: Mogi das Cruzes
Estado: Sao Paulo
E-mail: CtLt.Helpdesk@teleworm.br
Telefone: 11 99829944
CNPJ: 86.360.676/0001-01
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
```

Figura 4. Exibição das duas pessoas recém cadastradas.

```
run:
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
5
F - Pessoa Fisica | J - Pessoa Juridica
F
-- Exibindo todas as Pessoas Fisicas --
ID: 1
Nome: Joao Fernandes Correia
Endereco: 365.410.330-80
Cidade: Rua Jose Aristides
Estado: Divinopolis
E-mail: Minas Gerais
Telefone: JoaoFernandesCorreia@rhyta.com
CPF: 37 4647-9323

ID: 2
Nome: Kaua Azevedo Silva
Endereco: 78349632011
Cidade: Rua Faustino Siqueira Franco
Estado: Guarulhos
E-mail: Sao Paulo
Telefone: KauaAzevedoSilva@armyspy.com
CPF: 11 4681-5575

ID: 5
Nome: Vinicius Fernandes
Endereco: 304.441.749-07
Cidade: Rua do Oitizeiro 145
Estado: Serra
E-mail: Espirito Santo
Telefone: ViniciusFernandesSantos@dayrep.com
CPF: 27 77537638

-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
0
Encerrando o programa...
```

```
run:
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
5
F - Pessoa Fisica | J - Pessoa Juridica
J
-- Exibindo todas as Pessoas Juridicas --
ID: 3
Nome: Heilig-Meyers
Endereco: 95.823.031/0001-59
Cidade: Estrada do Recanto 161
Estado: Timoteo
E-mail: Minas Gerais
Telefone: Heilig-Meyers@teleworm.us
CNPJ: 31 5667-6663

ID: 4
Nome: Grupo Tecnico Jardins
Endereco: 67652009000158
Cidade: Praca Presidente Gamal Abdel Nasser 1403
Estado: Sao Paulo
E-mail: Sao Paulo
Telefone: Henceall@dayrep.com
CNPJ: 11 8783-6448

ID: 6
Nome: ConnectList
Endereco: 86.360.676/0001-01
Cidade: Rua Adelino Torquato 179
Estado: Mogi das Cruzes
E-mail: Sao Paulo
Telefone: CtLt.Helpdesk@teleworm.br
CNPJ: 11 99829944

-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
0
Encerrando o programa...
```

Figura 5. Exibição de todas as pessoas de cada tipo presentes no banco de dados.

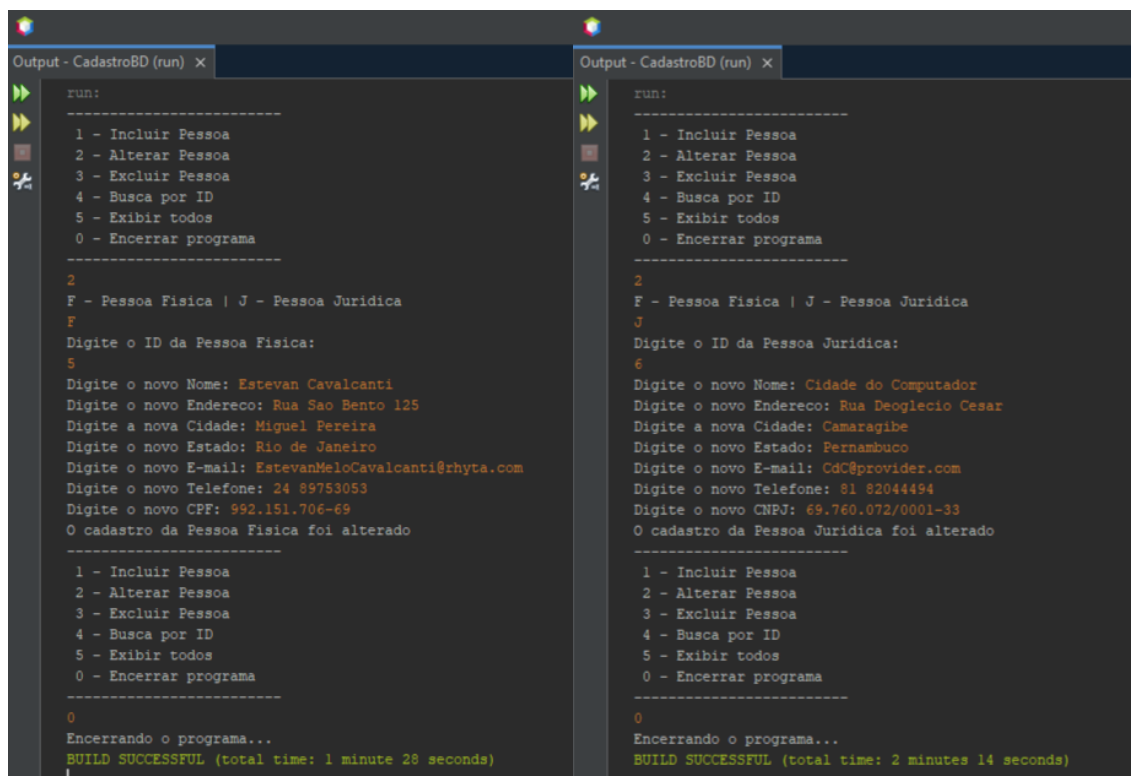


Figura 6. Alteração dos cadastros das pessoas recém adicionadas.

SELECT TOP 100 * FROM dbo... X							
		Max. rows: 100		Fetched Rows: 6		Matching Rows:	
#	idP	Nome	Endereco	Cidade	Estado	Email	Telefone
1	1	Joao Fernandes Correia	Rua Jose Aristides	Divinopolis	Minas Gerais	JoaoFernandesCorreia@rhyta.com	37 4647-9323
2	2	Kaua Azevedo Silva	Rua Faustino Siqueira Franco	Guarulhos	Sao Paulo	KauaAzevedoSilva@armyspy.com	11 4681-5575
3	3	Heilig-Meyers	Estrada do Recanto 161	Timoteo	Minas Gerais	Heilig-Meyers@teleworm.us	31 5667-6663
4	4	Grupo Tecnico Jardins	Praca Presidente Gamal Abdel Nasser 1403	Sao Paulo	Sao Paulo	Henceall@dayrep.com	11 8783-6448
5	5	Estevan Cavalcanti	Rua Sao Bento 125	Miguel Pereira	Rio de Janeiro	EstevanMeloCavalcanti@rhyta.com	24 89753053
6	6	Cidade do Computador	Rua Deoglecio Cesar	Camaragibe	Pernambuco	CdC@provider.com	81 82044494

SELECT TOP 100 * FROM dbo... X			SELECT TOP 100 * FROM dbo... X		
		Max. rows: 100			Max. rows: 100
		Fetched Rows: 3			Fetched Rows: 3
#	FK_Pessoa_id	CPF	#	FK_Pessoa_id	CNPJ
1	1	365.410.330-80	1	3	95.823.031/0001-59
2	2	78349632011	2	4	67652009000158
3	5	992.151.706-69	3	6	69.760.072/0001-33

Figura 7. Imagens do query no NetBeans mostrando que os cadastros das pessoas foram alterados no banco de dados. Ver imagem 4 para comparar com as informações originais.

```
run:
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
3
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o ID da Pessoa Fisica: 5
A Pessoa Fisica foi excluida
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
4
F - Pessoa Fisica | J - Pessoa Juridica
F
Digite o ID da Pessoa Fisica: 5
Pessoa Fisica nao encontrada.
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
0
Encerrando o programa...
BUILD SUCCESSFUL (total time: 25 seconds)
```

```
run:
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
3
F - Pessoa Fisica | J - Pessoa Juridica
J
Digite o ID da Pessoa Juridica: 6
A Pessoa Juridica foi excluida
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
4
F - Pessoa Fisica | J - Pessoa Juridica
J
Digite o ID da Pessoa Juridica: 6
Pessoa Juridica nao encontrada.
-----
1 - Incluir Pessoa
2 - Alterar Pessoa
3 - Excluir Pessoa
4 - Busca por ID
5 - Exibir todos
0 - Encerrar programa
-----
0
Encerrando o programa...
BUILD SUCCESSFUL (total time: 22 seconds)
```

Figura 8. Exclusão das pessoas cadastradas. Observe que uma busca pelos IDs agora retorna uma resposta que as pessoas não foram encontradas no banco de dados.

Análise e Conclusão – Prática 1

A) Qual a importância dos componentes de *middleware*, como o JDBC?

Middlewares como o *Java Database Connectivity* (JDBC) são *softwares* que, como o nome sugere, servem como intermediários entre a aplicação e o banco de dados. O JDBC funciona como interface para acessar e manipular o conteúdo de bancos de dados. Além disso, essa ferramenta é padronizada de modo a permitir que uma aplicação Java possa interagir com vários tipos de bancos de dados diferentes, sem necessidade de alteração na lógica de acesso de dados.

B) Qual a diferença no uso de *Statement* ou *PreparedStatement* para a manipulação de dados?

Ambos servem para executar comandos SQL em Java, mas o *Statement* executa apenas comandos explicitados no código, o que pode gerar vulnerabilidades no sistema. Em contrapartida, *PreparedStatement* executa comandos parametrizados, onde os comandos são preenchidos de maneira dinâmica, sendo uma opção mais segura do que o *Statement*. Em geral, o uso de *PreparedStatement* oferece mais vantagens para os programadores e para as aplicações.

C) Como o padrão DAO melhora a manutenibilidade do software?

O padrão *Data Access Object* (DAO) encapsula a lógica de acesso em classes específicas, separando essa parte do código do restante da aplicação. Algumas vantagens desse padrão são a melhor reutilização de código, maior legibilidade, entre outras. Tudo isso facilita a manutenção do código, já que, havendo necessidade de migrar de bancos de dados, por exemplo, basta editar as classes DAO, sem necessidade de mexer no código inteiro.

D) Como a herança é refletida no banco de dados, quando lidamos com um modelo estritamente relacional?

Herança não é representada de forma nativa em modelos relacionais, devendo ser mimetizada utilizando as ferramentas dos bancos de dados relacionais. Para fazer isso, deve-se criar uma tabela contendo os atributos comuns a todas as subclasses e outras tabelas cada uma contendo os atributos específicos de cada subclasse. Para criar a relação entre as tabelas, deve-se definir uma chave primária na primeira tabela, que representa a superclasse, e utilizar esta chave como chave estrangeira nas tabelas que representam as subclasses.

Análise e Conclusão – Prática 2

A) Quais as diferenças entre a persistência em arquivo e a persistência em banco de dados?

A persistência em arquivo armazena dados em arquivos binários ou de texto, como .bin, .xls, .csv, .xml, etc. Dessa forma, é um método rápido e simples de armazenar dados. Já a persistência em banco de dados faz uso de sistemas de gerenciamento de bancos de dados (SGBD), que armazenam dados de maneira mais estruturada, além de oferecer recursos avançados e vantagens como maior escalabilidade e possibilidade de consulta complexa de dados.

B) Como o uso de operador lambda simplificou a impressão dos valores contidos nas entidades, nas versões mais recentes do Java?

A utilização de operadores lambda, recurso que introduzido pela versão Java 8, simplifica a escrita de código ao permitir que funções sejam tratadas como objetos e eliminando a necessidade de se criar classes anônimas e grandes quantidades de códigos para realizar operações simples.

C) Por que métodos adicionados diretamente pelo método *main*, sem o uso de um objeto, precisam ser marcados como *static*?

O método *main* é o primeiro a ser chamado na execução de uma aplicação, portanto, marcar este método como *static* permite que o *Java Virtual Machine* (JVM) acesse-o diretamente, sem a necessidade de criar uma instância da classe que contém o método *main*. Entretanto, métodos estáticos só podem chamar outros métodos estáticos sem a necessidade de usar objetos, por isso que *main* só pode chamar outros métodos estáticos de maneira direta.

Anexo – Códigos do Projeto

Arquivo – Pessoa.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
 to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
 this template
 */
package model;

/**
 *
 * @author fel-f
 */
public class Pessoa {

    private Integer id;
    private String nome;
    private String endereco;
    private String cidade;
    private String estado;
    private String email;
    private String telefone;

    public Pessoa() {

    }

}
```

```
public Pessoa(Integer id, String nome, String endereco, String cidade, String
estado, String email, String telefone) {

    this.id = id;

    this.nome = nome;

    this.endereco = endereco;

    this.cidade = cidade;

    this.estado = estado;

    this.email = email;

    this.telefone = telefone;

}

public void exibir() {

    System.out.println("ID: " + id);

    System.out.println("Nome: " + nome);

    System.out.println("Endereco: " + endereco);

    System.out.println("Cidade: " + cidade);

    System.out.println("Estado: " + estado);

    System.out.println("E-mail: " + email);

    System.out.println("Telefone: " + telefone);

}

public void setID(int id) {

    this.id = id;

}

public int getID() {

    return id;

}
```

```
public void setNome(String nome) {  
    this.nome = nome;  
}  
  
public String getNome() {  
    return nome;  
}  
  
public void setEndereco(String endereco) {  
    this.endereco = endereco;  
}  
  
public String getEndereco() {  
    return endereco;  
}  
  
public void setCidade(String cidade) {  
    this.cidade = cidade;  
}  
  
public String getCidade() {  
    return cidade;  
}  
  
public void setEstado(String estado) {  
    this.estado = estado;  
}  
  
public String getEstado() {  
    return estado;  
}
```

```

    }

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

    public String getEmail() {
        return email;
    }

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

    public String getTelefone() {
        return telefone;
    }
}

```

Arquivo – PessoaFisica.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
this template
 */
package model;

/**
 *

```

```
* @author fel-f
```

```
*/
```

```
public class PessoaFisica extends Pessoa {
```

```
    private String cpf;
```

```
    public PessoaFisica() {
```

```
        super();
```

```
    }
```

```
    public PessoaFisica(int id, String nome, String endereco, String cidade,  
String estado, String email, String telefone, String cpf) {
```

```
        super(id, nome, endereco, cidade, estado, email, telefone);
```

```
        this.cpf = cpf;
```

```
    }
```

```
    public void setCPF(String cpf) {
```

```
        this.cpf = cpf;
```

```
    }
```

```
    public String getCPF() {
```

```
        return cpf;
```

```
    }
```

```
    @Override
```

```
    public void exibir() {
```

```
        super.exibir();
```

```
        System.out.println("CPF: " + cpf);
```

```
}  
}
```

Arquivo – PessoaJuridica.java

```
/*  
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt  
to change this license  
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit  
this template  
 */  
package model;  
  
/**  
 *  
 * @author fel-f  
 */  
  
public class PessoaJuridica extends Pessoa {  
  
    private String cnpj;  
  
    public PessoaJuridica() {  
        super();  
    }  
  
    public PessoaJuridica(int id, String nome, String endereco, String cidade,  
String estado, String email, String telefone, String cnpj) {  
        super(id, nome, endereco, cidade, estado, email, telefone);  
        this.cnpj = cnpj;  
    }  
}
```



```

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

public String getCNPJ() {
    return cnpj;
}

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

```

Arquivo – PessoaFisicaDAO.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
 to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
 this template
 */
package model;

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

```

```

import java.util.ArrayList;

import model.util.ConectorBD;

/**
 *
 * @author fel-f
 */

public class PessoaFisicaDAO {

    public ConectorBD connector;

    public PessoaFisicaDAO() {
        connector = new ConectorBD();
    }

    public PessoaFisica getPessoa(int id) throws SQLException {
        String sql = "SELECT pf.FK_Pessoa_id, pf.CPF, pe.Nome, pe.Endereco,
pe.Cidade, pe.Estado, pe.Email, pe.Telefone FROM PessoaFisica pf INNER
JOIN Pessoa pe ON pf.FK_Pessoa_id = pe.idPessoa WHERE
pf.FK_Pessoa_id = ?";

        try (Connection con = connector.getConnection(); PreparedStatement pst
= con.prepareStatement(sql)) {
            pst.setInt(1, id);
            try (ResultSet res = pst.executeQuery()) {
                if (res.next()) {
                    PessoaFisica peF = new PessoaFisica(
                        res.getInt("FK_Pessoa_id"),
                        res.getString("Nome"),
                        res.getString("Endereco"),

```

```

        res.getString("Cidade"),
        res.getString("Estado"),
        res.getString("Email"),
        res.getString("Telefone"),
        res.getString("CPF")
    );
    return peF;
}
}
}
return null;
}

```

```

public ArrayList<PessoaFisica> getPessoas() throws SQLException {
    ArrayList<PessoaFisica> list = new ArrayList<>();

    String sql = "SELECT pf.FK_Pessoa_id, pf.CPF, pe.Nome, pe.Endereco,
pe.Cidade, pe.Estado, pe.Email, pe.Telefone FROM PessoaFisica pf INNER
JOIN Pessoa pe ON pf.FK_Pessoa_id = pe.idPessoa";

    try (Connection con = connector.getConnection(); PreparedStatement pst
= con.prepareStatement(sql); ResultSet res = pst.executeQuery()) {
        while (res.next()) {
            list.add(new PessoaFisica(
                res.getInt("FK_Pessoa_id"),
                res.getString("Nome"),
                res.getString("CPF"),
                res.getString("Endereco"),
                res.getString("Cidade"),
                res.getString("Estado"),
                res.getString("Email"),
                res.getString("Telefone")
            ));
        }
    }
}

```

```

        ));
    }
}

return list;
}

public void incluir(PessoaFisica peF) throws SQLException {
    if (peF.getNome() == null || peF.getNome().trim().isEmpty()) {
        throw new IllegalArgumentException("O campo Nome não pode estar
vazio ou nulo.");
    }

    String sqlAddPessoa = "INSERT INTO Pessoa (idPessoa, Nome,
Endereco, Cidade, Estado, Email, Telefone) VALUES (?, ?, ?, ?, ?, ?, ?)";

    String sqlAddPessoaFisica = "INSERT INTO PessoaFisica
(FK_Pessoa_id, CPF) VALUES (?, ?)";

    try (Connection con = connector.getConnection(); PreparedStatement
pstPessoa = con.prepareStatement(sqlAddPessoa)) {
        pstPessoa.setInt(1, peF.getID());
        pstPessoa.setString(2, peF.getNome());
        pstPessoa.setString(3, peF.getEndereco());
        pstPessoa.setString(4, peF.getCidade());
        pstPessoa.setString(5, peF.getEstado());
        pstPessoa.setString(6, peF.getEmail());
        pstPessoa.setString(7, peF.getTelefone());
        int rows = pstPessoa.executeUpdate();
        if (rows == 0) {
            throw new SQLException("Ocorreu um erro ao criar a Pessoa.");
        }

        try (PreparedStatement pstPessoaFisica =
con.prepareStatement(sqlAddPessoaFisica)) {
            pstPessoaFisica.setInt(1, peF.getID());

```

```

        pstPessoaFisica.setString(2, peF.getCPF());
        pstPessoaFisica.executeUpdate();
    }

}

}

```

```

public void alterar(PessoaFisica peF) throws SQLException {
    String sqlUpdatePessoa = "UPDATE Pessoa SET Nome = ?, Endereco = ?, Cidade = ?, Estado = ?, Email = ?, Telefone = ? WHERE idPessoa = ?";
    String sqlUpdatePessoaFisica = "UPDATE PessoaFisica SET CPF = ? WHERE FK_Pessoa_id = ?";

    try (Connection con = connector.getConnection(); PreparedStatement
        pstPessoa = con.prepareStatement(sqlUpdatePessoa); PreparedStatement
        pstPessoaFisica = con.prepareStatement(sqlUpdatePessoaFisica)) {
        pstPessoa.setString(1, peF.getNome());
        pstPessoa.setString(2, peF.getEndereco());
        pstPessoa.setString(3, peF.getCidade());
        pstPessoa.setString(4, peF.getEstado());
        pstPessoa.setString(5, peF.getEmail());
        pstPessoa.setString(6, peF.getTelefone());
        pstPessoa.setInt(7, peF.getID());
        pstPessoa.executeUpdate();
        pstPessoaFisica.setString(1, peF.getCPF());
        pstPessoaFisica.setInt(2, peF.getID());
        pstPessoaFisica.executeUpdate();
    }
}

```

```

    }

    public void excluir(PessoaFisica peF) throws SQLException {

        String sqlDelPessoa = "DELETE FROM Pessoa WHERE idPessoa = ?;";

        String sqlDelPessoaFisica = "DELETE FROM PessoaFisica WHERE
FK_Pessoa_id = ?;";

        try (Connection con = connector.getConnection(); PreparedStatement
pstPessoa = con.prepareStatement(sqlDelPessoa); PreparedStatement
pstPessoaFisica = con.prepareStatement(sqlDelPessoaFisica)) {

            pstPessoa.setInt(1, peF.getID());

            pstPessoa.executeUpdate();

            pstPessoaFisica.setInt(1, peF.getID());

            pstPessoaFisica.executeUpdate();

        }

    }

    public void close() throws SQLException {

        connector.close();

    }

}

```

Arquivo – PessoaJuridicaDAO.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
this template
 */

package model;

import java.sql.Connection;

```

```

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import model.util.ConectorBD;

/**
 *
 * @author fel-f
 */
public class PessoaJuridicaDAO {

    public ConectorBD connector;

    public PessoaJuridicaDAO() {
        connector = new ConectorBD();
    }

    public PessoaJuridica getPessoa(int id) throws SQLException {
        String sql = "SELECT pj.FK_Pessoa_id, pj.CNPJ, pe.Nome, pe.Endereco,
pe.Cidade, pe.Estado, pe.Email, pe.Telefone FROM PessoaJuridica pj INNER
JOIN Pessoa pe ON pj.FK_Pessoa_id = pe.idPessoa WHERE pj.FK_Pessoa_id
= ?";

        try (Connection con = connector.getConnection(); PreparedStatement pst
= con.prepareStatement(sql)) {
            pst.setInt(1, id);
            try (ResultSet res = pst.executeQuery()) {
                if (res.next()) {
                    PessoaJuridica peJ = new PessoaJuridica(

```

```

        res.getInt("FK_Pessoa_id"),
        res.getString("Nome"),
        res.getString("Endereco"),
        res.getString("Cidade"),
        res.getString("Estado"),
        res.getString("Email"),
        res.getString("Telefone"),
        res.getString("CNPJ")
    );
    return peJ;
}
}
}
return null;
}

```

```

public ArrayList<PessoaJuridica> getPessoas() throws SQLException {
    ArrayList<PessoaJuridica> list = new ArrayList<>();

    String sql = "SELECT pj.FK_Pessoa_id, pj.CNPJ, pe.Nome, pe.Endereco,
pe.Cidade, pe.Estado, pe.Email, pe.Telefone FROM PessoaJuridica pj INNER
JOIN Pessoa pe ON pj.FK_Pessoa_id = pe.idPessoa";

    try (Connection con = connector.getConnection(); PreparedStatement pst
= con.prepareStatement(sql); ResultSet res = pst.executeQuery()) {
        while (res.next()) {
            list.add(new PessoaJuridica(
                res.getInt("FK_Pessoa_id"),
                res.getString("Nome"),
                res.getString("CNPJ"),
                res.getString("Endereco"),
                res.getString("Cidade"),

```



```

        res.getString("Estado"),
        res.getString("Email"),
        res.getString("Telefone")
    ));
    }
}
return list;
}

public void incluir(PessoaJuridica peJ) throws SQLException {
    if (peJ.getNome() == null || peJ.getNome().trim().isEmpty()) {
        throw new IllegalArgumentException("O campo Nome não pode estar
vazio ou nulo.");
    }

    String sqlAddPessoa = "INSERT INTO Pessoa (idPessoa, Nome,
Endereco, Cidade, Estado, Email, Telefone) VALUES (?, ?, ?, ?, ?, ?, ?)";

    String sqlAddPessoaJuridica = "INSERT INTO PessoaJuridica
(FK_Pessoa_id, CNPJ) VALUES (?, ?)";

    try (Connection con = connector.getConnection(); PreparedStatement
pstPessoa = con.prepareStatement(sqlAddPessoa)) {
        pstPessoa.setInt(1, peJ.getID());
        pstPessoa.setString(2, peJ.getNome());
        pstPessoa.setString(3, peJ.getEndereco());
        pstPessoa.setString(4, peJ.getCidade());
        pstPessoa.setString(5, peJ.getEstado());
        pstPessoa.setString(6, peJ.getEmail());
        pstPessoa.setString(7, peJ.getTelefone());
        int rows = pstPessoa.executeUpdate();
        if (rows == 0) {
            throw new SQLException("Ocorreu um erro ao criar a Pessoa.");
        }
    }
}

```

```

    }

    try (PreparedStatement pstPessoaFisica =
con.prepareStatement(sqlAddPessoaJuridica)) {

        pstPessoaFisica.setInt(1, peJ.getID());

        pstPessoaFisica.setString(2, peJ.getCNPJ());

        pstPessoaFisica.executeUpdate();

    }

}

}

```

```

public void alterar(PessoaJuridica peJ) throws SQLException {

    String sqlUpdatePessoa = "UPDATE Pessoa SET Nome = ?, Endereco =
?, Cidade = ?, Estado = ?, Email = ?, Telefone = ? WHERE idPessoa = ?";

    String sqlUpdatePessoaFisica = "UPDATE PessoaJuridica SET CNPJ = ?
WHERE FK_Pessoa_id = ?";

    try (Connection con = connector.getConnection(); PreparedStatement
pstPessoa = con.prepareStatement(sqlUpdatePessoa); PreparedStatement
pstPessoaFisica = con.prepareStatement(sqlUpdatePessoaFisica)) {

        pstPessoa.setString(1, peJ.getNome());

        pstPessoa.setString(2, peJ.getEndereco());

        pstPessoa.setString(3, peJ.getCidade());

        pstPessoa.setString(4, peJ.getEstado());

        pstPessoa.setString(5, peJ.getEmail());

        pstPessoa.setString(6, peJ.getTelefone());

        pstPessoa.setInt(7, peJ.getID());

        pstPessoa.executeUpdate();

        pstPessoaFisica.setString(1, peJ.getCNPJ());

        pstPessoaFisica.setInt(2, peJ.getID());
    }
}

```

```

        pstPessoaFisica.executeUpdate();
    }
}

public void excluir(PessoaJuridica peJ) throws SQLException {
    String sqlDelPessoa = "DELETE FROM Pessoa WHERE idPessoa = ?;";
    String sqlDelPessoaFisica = "DELETE FROM PessoaJuridica WHERE
FK_Pessoa_id = ?;";

    try (Connection con = connector.getConnection(); PreparedStatement
pstPessoa = con.prepareStatement(sqlDelPessoa); PreparedStatement
pstPessoaFisica = con.prepareStatement(sqlDelPessoaFisica)) {

        pstPessoa.setInt(1, peJ.getID());
        pstPessoa.executeUpdate();
        pstPessoaFisica.setInt(1, peJ.getID());
        pstPessoaFisica.executeUpdate();
    }
}

public void close() throws SQLException {
    connector.close();
}
}

```

Arquivo – ConectorBD.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
this template
 */
package model.util;

```

```

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

/**
 *
 * @author fel-f
 */

public class ConectorBD {

    public Connection con;
    public PreparedStatement pst;
    public ResultSet res;

    public Connection getConnection() throws SQLException {
        String url =
"jdbc:sqlserver://localhost:1433;databaseName=Loja;encrypt=true;trustServerC
ertificate=true";

        String user = "loja";
        String password = "loja";
        con = DriverManager.getConnection(url, user, password);
        return con;
    }
}

```

```

public ResultSet getSelect(String sql) throws SQLException {
    pst = getConnection().prepareStatement(sql);
    res = pst.executeQuery();
    return res;
}

public int insert(String sql) throws SQLException {
    pst = getConnection().prepareStatement(sql,
Statement.RETURN_GENERATED_KEYS);
    pst.executeUpdate();
    res = pst.getGeneratedKeys();
    if (res.next()) {
        return res.getInt(1);
    } else {
        throw new SQLException("Ocorreu um erro ao inserir os dados.");
    }
}

public boolean update(String sql) throws SQLException {
    pst = getConnection().prepareStatement(sql);
    int rows = pst.executeUpdate();
    return rows > 0;
}

public void close() throws SQLException {
    if (pst != null && !pst.isClosed()) {
        pst.close();
    }
    if (res != null && !res.isClosed()) {

```

```

        res.close();
    }
    if (con != null && !con.isClosed()) {
        con.close();
    }
}
}

```

Arquivo – SequenceManager.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
this template
 */
package model.util;

import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 * @author fel-f
 */
public class SequenceManager {

    public int getValue (String sequenceName) throws SQLException {

        ResultSet res = new ConectorBD().getSelect("SELECT NEXT VALUE FOR
" + sequenceName);
    }
}

```

```

        if (res.next()) {
            return res.getInt(1);
        } else {
            throw new SQLException ("Ocorreu um erro ao obter o proximo valor da
sequencia" + sequenceName);
        }
    }
}
}

```

Arquivo – CadastroBDTeste.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
this template
 */

package cadastrabd;

import java.sql.SQLException;
import java.util.ArrayList;
import model.PessoaFisicaDAO;
import model.PessoaJuridicaDAO;
import model.PessoaFisica;
import model.PessoaJuridica;

/**
 *
 * @author fel-f
 */

```

```

public class CadastroBDTeste {

    private final PessoaFisicaDAO PFD;
    private final PessoaJuridicaDAO PJD;

    public CadastroBDTeste() {
        PFD = new PessoaFisicaDAO();
        PJD = new PessoaJuridicaDAO();
    }

    private void run() {
        PessoaFisica peF = new PessoaFisica(5, "Vinicius Rocha Lima",
        "Travessa Sao Joao, 186", "Rio de Janeiro", "Rio de Janeiro",
        "ViniciusRochaLima@dayrep.com", "21 6391-4762", "127.460.858-95");

        if (peF.getNome() == null || peF.getNome().trim().isEmpty()) {
            System.out.println("O campo Nome não pode estar vazio.");
            return;
        }

        try {
            PFD.incluir(peF);
            System.out.println("A Pessoa Fisica foi incluída com sucesso");
            peF.exibir();
            peF.setEndereco("Rua Armandino Gonzaga, 386");
            peF.setCidade("Florianopolis");
            peF.setEstado("Santa Catarina");
            PFD.alterar(peF);
            System.out.println("Pessoa Fisica alterada com sucesso.");
        }
    }
}

```



```

        peF.exibir();

        ArrayList<PessoaFisica> listaPF = PFD.getPessoas();

        System.out.println("Exibindo todas as Pessoas Fisicas:");

        for (PessoaFisica pessoa : listaPF) {

            pessoa.exibir();

        }

        PFD.excluir(peF);

        System.out.println("Pessoa Fisica excluida com sucesso");

        PFD.close();

    }

    catch (SQLException e) {

        System.out.println("Ocorreu o seguinte erro: " + e.getMessage());

    }

```

```

        PessoaJuridica peJ = new PessoaJuridica(6, "Patterson-Fletcher",
        "Travessa das Cravinas, 103", "Salvador", "Bahia",
        "assistenciaPFjourrapide.com", "71 8036-9442", "81.669.950/0001-05");

```

```

        if (peJ.getNome() == null || peJ.getNome().trim().isEmpty()) {

            System.out.println("O campo Nome não pode estar vazio");

            return;

        }

```

```

        try {

            PJD.incluir(peJ);

            System.out.println("A Pessoa Juridica foi incluida com sucesso");

            peJ.exibir();

            peJ.setNome("Fletcher Enterprises");

            peJ.setEmail("Helpdesk.FletcherEnterprises@domain.com");

            PJD.alterar(peJ);

```

```

        System.out.println("Pessoa Juridica Alterada com sucesso.");
        peJ.exibir();
        ArrayList<PessoaJuridica> listaPJ = PJD.getPessoas();
        System.out.println("Exibindo toas as Pessoas Juridicas:");
        for (PessoaJuridica pessoa : listaPJ) {
            pessoa.exibir();
        }
        PJD.excluir(peJ);
        System.out.println("Pessoa Juridica excluida com sucesso.");
        PJD.close();
    }
    catch (SQLException e) {
        System.out.println("Ocorreu o seguinte erro: " + e.getMessage());
    }
}

public static void main(String[] args) {
    new CadastroBDTeste().run();
}
}

```

Arquivo – CadastroBD.java

```

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit
this template
 */

package cadastrabd;

```

```

import java.sql.SQLException;
import java.util.List;
import java.util.logging.Logger;
import java.util.logging.Level;
import java.util.Scanner;
import model.PessoaFisica;
import model.PessoaFisicaDAO;
import model.PessoaJuridica;
import model.PessoaJuridicaDAO;

/**
 *
 * @author fel-f
 */

public class CadastroBD {

    private Scanner scan;
    private PessoaFisicaDAO PFD;
    private PessoaJuridicaDAO PJD;
    private static final Logger lgr =
        Logger.getLogger(CadastroBD.class.getName());

    public CadastroBD() {
        scan = new Scanner(System.in);
        PFD = new PessoaFisicaDAO();
        PJD = new PessoaJuridicaDAO();
    }
}

```

```

private void menu() {
    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 - Busca por ID");
    System.out.println(" 5 - Exibir todos");
    System.out.println(" 0 - Encerrar programa");
    System.out.println("-----");
}

public void run() {
    int opcaoMenu = -1;
    while (opcaoMenu != '0') {
        menu();
        opcaoMenu = scan.next().charAt(0);
        scan.nextLine();
        try {
            switch (opcaoMenu) {
                case '1' -> {
                    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
                    char tipoPessoa = scan.next().charAt(0);
                    scan.nextLine();
                    switch (tipoPessoa) {
                        case 'F' -> {
                            cadastroPF(PFD, scan);
                        }
                        case 'J' -> {

```

```

        cadastroPJ(PJD, scan);
    }
    default -> {
        System.out.println("Tipo de cadastro invalido.");
    }
}
break;
}
case '2' -> {
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    char tipoPessoa = scan.next().charAt(0);
    scan.nextLine();
    switch (tipoPessoa) {
        case 'F' -> {
            alterarPF(PFD, scan);
        }
        case 'J' -> {
            alterarPJ(PJD, scan);
        }
        default -> {
            System.out.println("Tipo de cadastro invalido.");
        }
    }
    break;
}
case '3' -> {
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    char tipoPessoa = scan.next().charAt(0);
    scan.nextLine();

```

```

switch (tipoPessoa) {
    case 'F' -> {
        try {
            System.out.print("Digite o ID da Pessoa Fisica: ");
            int exclusao = Integer.parseInt(scan.nextLine());
            PessoaFisica peF = PFD.getPessoa(exclusao);
            if (peF != null){
                PFD.excluir(peF);
                System.out.println("A Pessoa Fisica foi excluida");
            } else {
                System.out.println("O ID nao foi encontrado.");
            }
        }
        catch (NullPointerException | SQLException e) {
            lgr.log(Level.SEVERE, e.toString(), e);
        }
    }
    case 'J' -> {
        try {
            System.out.print("Digite o ID da Pessoa Juridica: ");
            int exclusao = Integer.parseInt(scan.nextLine());
            PessoaJuridica peJ = PJD.getPessoa(exclusao);
            if (peJ != null){
                PJD.excluir(peJ);
                System.out.println("A Pessoa Juridica foi excluida");
            } else {
                System.out.println("O ID nao foi encontrado.");
            }
        }
    }
}

```

```

        catch (NullPointerException | SQLException e) {
            lgr.log(Level.SEVERE, e.toString(), e);
        }
    }
    default -> {
        System.out.println("Tipo de cadastro invalido.");
    }
}
break;
}
case '4' -> {
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");
    char tipoPessoa = scan.next().charAt(0);
    scan.nextLine();
    switch (tipoPessoa) {
        case 'F' -> {
            buscaPF(PFD, scan);
        }
        case 'J' -> {
            buscaPJ(PJD, scan);
        }
        default -> {
            System.out.println("Tipo de cadastro invalido.");
        }
    }
    break;
}
case '5' -> {
    System.out.println("F - Pessoa Fisica | J - Pessoa Juridica");

```

```

        char tipoPessoa = scan.next().charAt(0);
        scan.nextLine();
        switch (tipoPessoa) {
            case 'F' -> {
                exibirPF(PFD);
            }
            case 'J' -> {
                exibirPJ(PJD);
            }
            default -> {
                System.out.println("Tipo de cadastro invalido.");
            }
        }
        break;
    }
    case '0' -> {
        System.out.println("Encerrando o programa...");
    }
    default -> {
        System.out.println("Por favor digite uma opção válida.");
    }
}

catch (SQLException e) {
    System.out.println("Ocorreu o seguinte erro: " + e.getMessage());
}

}
}

```



```

private static void cadastroPF (PessoaFisicaDAO PFD, Scanner scan)
throws SQLException {

    System.out.print("Digite o proximo ID: ");
    int id = Integer.parseInt(scan.nextLine());

    System.out.print("Digite o Nome: ");
    String nome = scan.nextLine();

    System.out.print("Digite o Endereco: ");
    String endereco = scan.nextLine();

    System.out.print("Digite a Cidade: ");
    String cidade = scan.nextLine();

    System.out.print("Digite o Estado: ");
    String estado = scan.nextLine();

    System.out.print("Digite o E-mail: ");
    String email = scan.nextLine();

    System.out.print("Digite o Telefone: ");
    String telefone = scan.nextLine();

    System.out.print("Digite o CPF: ");
    String cpf = scan.nextLine();

    PessoaFisica novaPF = new PessoaFisica(id, nome, endereco, cidade,
estado, email, telefone, cpf);

    PFD.incluir(novaPF);

    System.out.println("Pessoa Fisica cadastrada com sucesso.");
}

```

```

private static void cadastroPJ (PessoaJuridicaDAO PJD, Scanner scan)
throws SQLException {

    System.out.print("Digite o proximo ID: ");
    int id = Integer.parseInt(scan.nextLine());

    System.out.print("Digite o Nome: ");
    String nome = scan.nextLine();

```

```

        System.out.print("Digite o Endereco: ");
        String endereco = scan.nextLine();
        System.out.print("Digite a Cidade: ");
        String cidade = scan.nextLine();
        System.out.print("Digite o Estado: ");
        String estado = scan.nextLine();
        System.out.print("Digite o E-mail: ");
        String email = scan.nextLine();
        System.out.print("Digite o Telefone: ");
        String telefone = scan.nextLine();
        System.out.print("Digite o CNPJ: ");
        String cnpj = scan.nextLine();

        PessoaJuridica novaPJ = new PessoaJuridica(id, nome, endereco, cidade,
estado, email, telefone, cnpj);
        PJD.incluir(novaPJ);
        System.out.println("Pessoa Juridica cadastrada com sucesso.");
    }

    private static void alterarPF (PessoaFisicaDAO PFD, Scanner scan) throws
SQLException {
        System.out.println("Digite o ID da Pessoa Fisica: ");
        int id = Integer.parseInt(scan.nextLine());
        PessoaFisica pessoaAlter = PFD.getPessoa(id);
        if (pessoaAlter != null) {
            System.out.print("Digite o novo Nome: ");
            String novoNome = scan.nextLine();
            System.out.print("Digite o novo Endereco: ");
            String novoEndereco = scan.nextLine();
            System.out.print("Digite a nova Cidade: ");
            String novaCidade = scan.nextLine();

```

```

        System.out.print("Digite o novo Estado: ");
        String novoEstado = scan.nextLine();
        System.out.print("Digite o novo E-mail: ");
        String novoEmail = scan.nextLine();
        System.out.print("Digite o novo Telefone: ");
        String novoTelefone = scan.nextLine();
        System.out.print("Digite o novo CPF: ");
        String novoCPF = scan.nextLine();

        PessoaFisica pessoaTemp = new PessoaFisica(id, novoNome,
novoEndereco, novaCidade, novoEstado, novoEmail, novoTelefone, novoCPF);
        PFD.alterar(pessoaTemp);

        System.out.println("O cadastro da Pessoa Fisica foi alterado");
    } else {
        System.out.println("Pessoa Fisica nao encontrada.");
    }
}

private static void alterarPJ (PessoaJuridicaDAO PJD, Scanner scan) throws
SQLException {
    System.out.println("Digite o ID da Pessoa Juridica: ");
    int id = Integer.parseInt(scan.nextLine());
    PessoaJuridica pessoaAlter = PJD.getPessoa(id);
    if (pessoaAlter != null) {
        System.out.print("Digite o novo Nome: ");
        String novoNome = scan.nextLine();
        System.out.print("Digite o novo Endereco: ");
        String novoEndereco = scan.nextLine();
        System.out.print("Digite a nova Cidade: ");
        String novaCidade = scan.nextLine();
        System.out.print("Digite o novo Estado: ");

```

```

        String novoEstado = scan.nextLine();
        System.out.print("Digite o novo E-mail: ");
        String novoEmail = scan.nextLine();
        System.out.print("Digite o novo Telefone: ");
        String novoTelefone = scan.nextLine();
        System.out.print("Digite o novo CNPJ: ");
        String novoCNPJ = scan.nextLine();

        PessoaJuridica pessoaTemp = new PessoaJuridica(id, novoNome,
novoEndereco, novaCidade, novoEstado, novoEmail, novoTelefone,
novoCNPJ);

        PJD.alterar(pessoaTemp);

        System.out.println("O cadastro da Pessoa Juridica foi alterado");
    } else {
        System.out.println("Pessoa Juridica nao encontrada.");
    }
}

private static void buscaPF(PessoaFisicaDAO PFD, Scanner scan) throws
SQLException {
    System.out.print("Digite o ID da Pessoa Fisica: ");
    int id = Integer.parseInt(scan.nextLine());
    PessoaFisica pessoa = PFD.getPessoa(id);
    if (pessoa != null) {
        System.out.println("-- Exibindo cadastro de Pessoa Fisica --");
        System.out.println("ID: " + pessoa.getID());
        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Endereco: " + pessoa.getEndereco());
        System.out.println("Cidade: " + pessoa.getCidade());
        System.out.println("Estado: " + pessoa.getEstado());
        System.out.println("E-mail: " + pessoa.getEmail());
    }
}

```

```

        System.out.println("Telefone: " + pessoa.getTelefone());
        System.out.println("CPF: " + pessoa.getCPF());
    } else {
        System.out.println("Pessoa Fisica nao encontrada.");
    }
}

```

```

private static void buscaPJ(PessoaJuridicaDAO PJD, Scanner scan) throws
SQLException {

```

```

    System.out.print("Digite o ID da Pessoa Juridica: ");
    int id = Integer.parseInt(scan.nextLine());
    PessoaJuridica pessoa = PJD.getPessoa(id);
    if (pessoa != null) {
        System.out.println("-- Exibindo cadastro de Pessoa Juridica --");
        System.out.println("ID: " + pessoa.getID());
        System.out.println("Nome: " + pessoa.getNome());
        System.out.println("Endereco: " + pessoa.getEndereco());
        System.out.println("Cidade: " + pessoa.getCidade());
        System.out.println("Estado: " + pessoa.getEstado());
        System.out.println("E-mail: " + pessoa.getEmail());
        System.out.println("Telefone: " + pessoa.getTelefone());
        System.out.println("CNPJ: " + pessoa.getCNPJ());
    } else {
        System.out.println("Pessoa Juridica nao encontrada.");
    }
}

```

```

private static void exibirPF(PessoaFisicaDAO PFD) throws SQLException {
    List<PessoaFisica> pessoas = PFD.getPessoas();

```

```

        if (pessoas.isEmpty()) {
            System.out.println("Não foram encontrados cadastros de Pessoas
Fisicas.");
        } else {
            System.out.println("-- Exibindo todas as Pessoas Fisicas --");
            for (PessoaFisica PF : pessoas) {
                System.out.println("ID: " + PF.getID());
                System.out.println("Nome: " + PF.getNome());
                System.out.println("Endereco: " + PF.getEndereco());
                System.out.println("Cidade: " + PF.getCidade());
                System.out.println("Estado: " + PF.getEstado());
                System.out.println("E-mail: " + PF.getEmail());
                System.out.println("Telefone: " + PF.getTelefone());
                System.out.println("CPF: " + PF.getCPF());
                System.out.println();
            }
        }
    }

    private static void exibirPJ(PessoaJuridicaDAO PJD) throws SQLException {
        List<PessoaJuridica> pessoas = PJD.getPessoas();
        if (pessoas.isEmpty()) {
            System.out.println("Não foram encontrados cadastros de Pessoas
Juridicas.");
        } else {
            System.out.println("-- Exibindo todas as Pessoas Juridicas --");
            for (PessoaJuridica PJ : pessoas) {
                System.out.println("ID: " + PJ.getID());
                System.out.println("Nome: " + PJ.getNome());
                System.out.println("Endereco: " + PJ.getEndereco());
            }
        }
    }
}

```

```

        System.out.println("Cidade: " + PJ.getCidade());
        System.out.println("Estado: " + PJ.getEstado());
        System.out.println("E-mail: " + PJ.getEmail());
        System.out.println("Telefone: " + PJ.getTelefone());
        System.out.println("CNPJ: " + PJ.getCNPJ());
        System.out.println();
    }
}

/**
 * @param args the command line arguments
 */
public static void main(String[] args) {
    new CadastroBD().run();
}
}

```