Instituto Federal de Educação, Ciência e Tecnologia- IFPB

Análise e Desenvolvimento de Sistemas

Iago Alves de Oliveira

Pedro Pereira de Morais Júnior

Daniel de Oliveira Sousa

Gestão comercial de uma livraria.

Cajazeiras – PB

2022

Iago Alves de Oliveira
Pedro Pereira de Morais Júnior
Daniel de Oliveira Sousa

Gestão comercial de uma livraria

Projeto apresentado ao Instituto Federal da Paraíba - Campus Cajazeiras, em cumprimento aos requisitos para disciplina banco de dados I

Cajazeiras – PB
2022

Sumário

1. Introdução

Considerando a competitividade do mercado junto com as novas tecnologias e o marketing, a livraria Nova Era, visa torna-se uma opção competitiva no mercado de vendas de livros em loja física e online. Desta maneira foi proposto para a equipe o desenvolvimento de uma aplicação de banco de dados visando a melhor experiência para os clientes e uma melhor organização e controle de estoque para loja.

2. Modelo Conceitual

2.1. Levantamento dos requisitos

Pensando no controle de todos os exemplares existentes e vendidos pela loja e no controle das finanças, a livraria vai ter necessariamente que armazenar as informações dos livros disponíveis, como o código dos livros, o título, o autor, o preço, a categoria e o ano de lançamento.

Cada livro desta livraria tem um número de exemplares. Sobre eles, dever-se-á armazenar o seu status atual - ou seja, se já foram vendidos ou se encontram disponíveis para venda - e o seu código.

Esta livraria também quer guardar as informações das editoras dos livros para que ela possa realizar pedidos de novos livros. As informações que importam ser guardadas sobre elas são CNPJ, números de telefone, e-mails, nome e código para identificação.

Para melhor experiência dos clientes a Nova Era tem o objetivo de implantar a opção de vendas on-line e ainda continuar com as vendas presenciais, dando assim ao cliente a oportunidade de fazer pedidos em qualquer uma das formas.

Independente do tipo da venda, é necessário armazenar seu número, status, valor, forma de pagamento e a data de realização. Entretanto, se a venda foi feita remotamente, além das características citadas anteriormente, é preciso guardar o endereço do cliente que fez a compra (estado, cidade, bairro, rua, número, complemento e CEP).

Quando a venda é presencial deve-se armazenar quem foi o funcionário responsável por registrá-la. Sobre esse funcionário é desejado saber tanto seu código, quanto seu nome, ano de admissão e salário.

Para controles fiscais os clientes da livraria têm que informar algumas informações. Informações essas que são de suma importância. Elas são: nome, e-mail e telefone. Além disso, a livraria precisa de um código para cada cliente.

A Nova Era aceita pedidos tanto de clientes físicos quanto de jurídicos. Quando um cliente físico faz uma compra, ele tem que informar seu RG e CPF, já quando um cliente jurídico o faz, este tem que informar seu CNPJ.

2.2. Modelo Entidade-Relacionamento

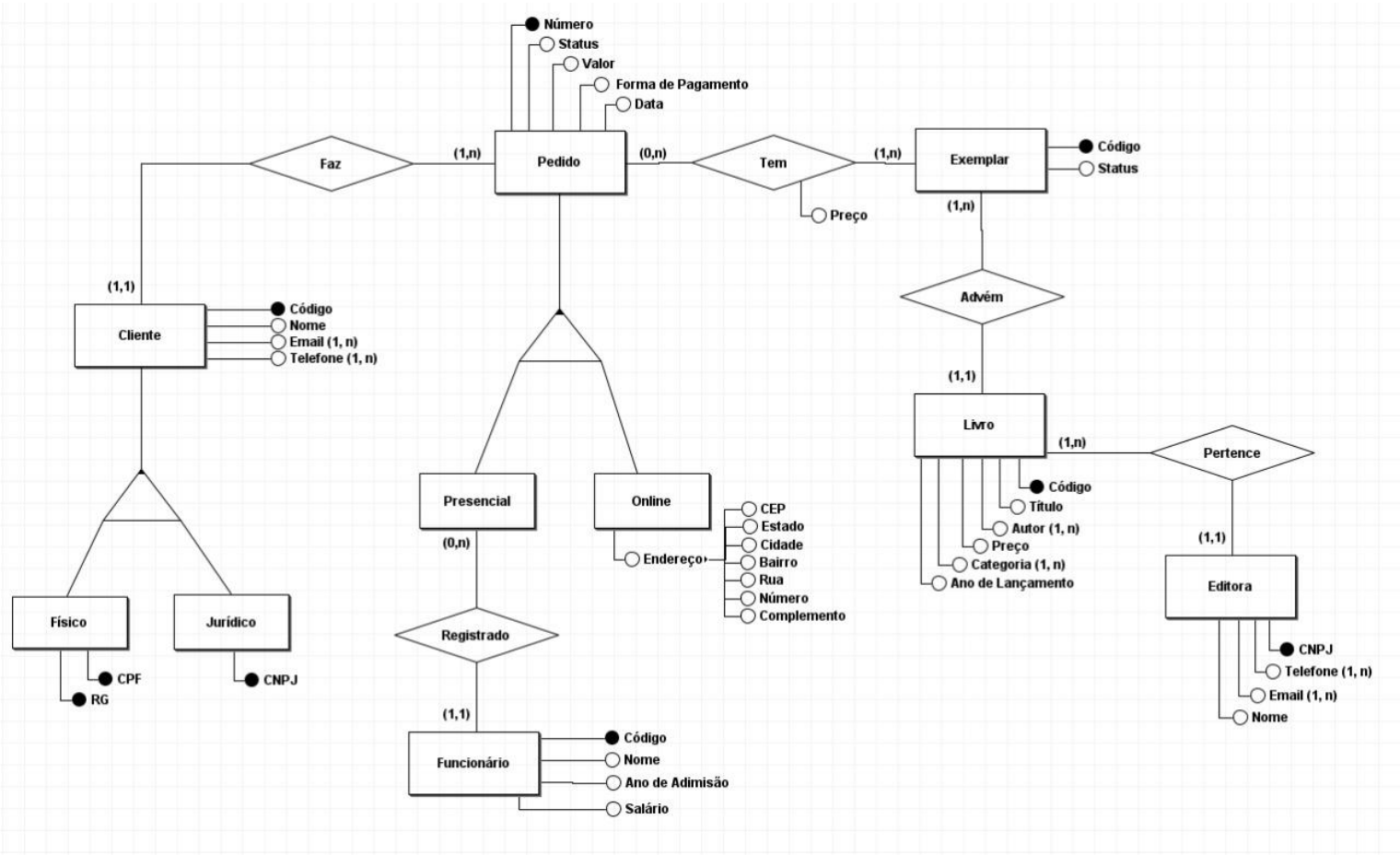

Figura 1: *Modelo Entidade-Relacionamento*

2.3. Dicionário de Dados

Entidade Cliente:

- Representa um cliente da livraria. Além disso é uma generalização da entidade Físico e Jurídico que representam respectivamente se o cliente é uma pessoa física ou jurídica.

Atributos do Cliente:

- Código: é um atributo chave, representa o código do cliente;
- Nome: armazena o nome do cliente;
- E-mail: atributo multivalorado que informa e-mails do cliente;
- Telefone: atributo multivalorado que diz respeito aos telefones do cliente.

Entidade Físico:

- Representa um cliente físico da livraria. Essa entidade é uma especialização da entidade Cliente, juntamente com a entidade Jurídico.

Atributos da Físico:

- CPF: é um atributo chave, armazena o CPF do cliente físico;
- RG: é um atributo chave que guarda o RG do cliente físico.

Entidade Jurídico:

- Representa um cliente jurídico da livraria. Essa entidade é, juntamente da entidade Físico, uma especialização da entidade Cliente.

Atributos da Jurídico:

- CNPJ: é um atributo chave, armazena o CNPJ do cliente jurídico.

Entidade Pedido:

- Representa um pedido feito nessa livraria. Essa entidade é uma generalização das entidades Presencial e Online que trazem informações se o pedido é presencial ou online.

Atributos do Pedido:

- Número: é um atributo chave, guarda informação a respeito do número do pedido;
- Status: diz o status atual do pedido;
- Valor: armazena quanto o pedido custou;
- Forma de pagamento: revela por qual meio o pedido foi pago;
- Data: armazena a data na qual foi realizado o pedido.

Entidade Presencial:

- Representa um pedido feito presencialmente na livraria. Essa entidade, junto a entidade Online, é uma especialização da entidade Pedido.

Atributos do Presencial: A entidade Presencial não possui atributos.

Entidade Online:

- Representa um pedido feito remotamente na livraria. Essa entidade é uma especialização da entidade Pedido, juntamente com a entidade Presencial.

Atributos do Online:

- Endereço: Atributo que armazena o endereço de um cliente que fez um pedido on-line. Esse atributo é composto por CEP, Estado, Cidade, Bairro, Rua, Número e Complemento.

Entidade Funcionário:

- É uma entidade criada com o propósito de armazenar informações sobre os funcionários da livraria.

Atributos do Funcionário:

- Código: é um atributo chave, representa o código do funcionário;
- Nome: armazena o nome do funcionário;
- Ano de Admissão: diz respeito ao ano no qual o funcionário foi admitido na empresa;
- Salário: armazena quanto um funcionário recebe de salário.

Entidade Exemplar:

- Essa entidade deve armazenar informações sobre exemplares de um livro.

Atributos do Exemplar:

- Código: é um atributo chave, armazena o código do exemplar;
- Status: Diz se esse exemplar está disponível para venda ou se já foi vendido.

Entidade Livro:

- É uma entidade criada com o propósito de armazenar informações sobre os livros presentes na livraria.

Atributos do Livro:

- Código: é um atributo chave, armazena o código do livro;
- Título: informa o título do livro;
- Autor: atributo multivalorado que guarda o(s) nome(s) do autor(es) do livro;
- Preço: revela quanto custa o livro atualmente;
- Categoria: atributo multivalorado que diz a(s) categoria(s) na qual se encaixa o livro;
- Ano de lançamento: representa o ano de lançamento do livro.

Entidade Editora:

- É uma entidade criada com o propósito de armazenar informações sobre as editoras dos livros.

Atributos da Editora:

- CNPJ: atributo chave que guarda o CPNJ da editora;
- Telefone: atributo multivalorado que armazena telefones da editora;
- E-mail: atributo multivalorado que armazena e-mails da editora;
- Nome: guarda a informação do nome da editora.

Relacionamentos:

- **Relacionamento Cliente Faz Pedido:** Esse relacionamento representa o ato do cliente de fazer um pedido de exemplares de livros para compra. Cada Cliente pode fazer de 1 a vários pedidos e cada Pedido pode ser realizado por um único Cliente.

- **Relacionamento Funcionário Registra Pedido Presencial:** Esse relacionamento representa o ato do funcionário registrar um pedido realizado presencialmente. Cada pedido presencial pode ser registrado por um único funcionário e cada funcionário registra de 0 a vários pedidos presenciais.

- **Relacionamento Pedido Tem Exemplar:** Esse relacionamento representa o pertencimento de exemplares a um pedido. Cada pedido pode conter de 1 a vários exemplares e cada exemplar pode estar em nenhum ou um pedido. Esse relacionamento possui um atributo Preço que armazena o preço pelo qual foi vendido um determinado exemplar.

- **Relacionamento Exemplar Advém Livro:** Esse relacionamento representa que cada exemplar é um exemplar de um livro específico. Cada Livro pode ter de um a vários exemplares e cada exemplar pode ser unicamente de 1 livro.

- **Relacionamento Livro Pertence Editora:** Esse relacionamento representa o pertencimento de um livro a uma determinada editora. Cada Livro pode pertencer a uma única Editora e cada Editora pode publicar de 1 a vários livros.

3. Modelo Lógico

3.1. Mapeamento de Entidades e Relacionamentos

1. **Cliente:** (Código, Nome, E-mail 1, E-mail 2, Telefone 1, Telefone 2)

2. **ClienteFísico:** (*CódigoCliente*, RG, CPF)

3. **ClienteJurídico:** (*CódigoCliente*, CNPJ)

4. **Pedido:** (Número, Status, Valor, Data, FormaDePagamento, *CódigoCliente*)

5. **PedidoPresencial:** (*NúmeroPedido*, *CódigoFuncionario*)

6. **PedidoOnline:** (*NúmeroPedido*, CEP, Estado, Cidade, Rua, Número, Complemento)

7. **Funcionario:** (Código, Nome, Salario, AnoDeAdmissão)

8. **Exemplar:** (Código, Status, *CódigoLivro*)

9. **PedidoTemExemplar:** (Id, Preço, *NúmeroPedido*, *CódigoExemplar*)

10. **Livro:** (Código, Título, Preço, AnoDeLançamento, *CNPJEditora*)

11. **LivroAutor:** (CódigoLivro, Autor)

12. **LivroCategoria:** (CódigoLivro, Categoria)

13. **Editora:** (CNPJ, Nome)

14. **TelefoneEditora:** (Telefone, *CNPJEditora*)

15. **E-mail Editora:** (E-mail, *CNPJEditora*)

3.2. Refinamento do Esquema Entidade Relacionamento

Houve um pequeno refinamento, em que os atributos Autor e Categoria, da entidade livro foram transformados em atributos multivalorados, criando assim no mapeamento uma relação para cada.

3.3. Dicionário Lógico de Dados

Cliente: Generalização das Entidades ClienteFísico e ClienteJurídico, guarda atributos comuns de um cliente				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Código	Armazena codigo como Indentificador do cliente	INT	Valores Positivos	* Chave Primária
Nome	Armazena nome do cliente	VARCHAR(255)	VARCHAR(255)	* Não nulo
E-mail 1	Armazena um dos e-mails do cliente	VARCHAR(255)	VARCHAR(255)	* Não nulo
E-mail 2	Armazena um dos e-mails do cliente	VARCHAR(255)	VARCHAR(255)	Nenhuma Restrição
Telefone 1	Armazena um dos telefones do cliente	CHARACTER(11)	Caracteres que representam Números	* Não nulo
Telefone 2	Armazena um dos telefones do cliente	CHARACTER(11)	Caracteres que representam Números	Nenhuma Restrição

Tabela 01 – Relação Cliente

ClienteFisico: Expercialização da Entidade Cliente, guarda atributos comuns de um cliente como pessoa física				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CodigoCliente	Armazena codigo de um cliente	INT	Valores Positivos	* Chave Primária
RG	Armazena o RG de um cliente	VARCHAR(50)	Caracteres que representam Números	* Não nulo * Unique
CPF	Armazena o CPF de um cliente	CHARACTER(11)	Caracteres que representam Números	* Não nulo * Unique

Tabela 02 – Relação ClienteFísico

ClienteJurídico: Expercialização da Entidade Cliente, armazena informações de um cliente como pessoa Júridica				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CodigoCliente	Armazena codigo de um cliente	INT	Valores Positivos	* Chave Primária
CNPJ	Armazena o CNPJ de um cliente	CHARACTER(14)	Caracteres que representam Números	* Não nulo * Unique

Tabela 03 – Relação ClienteJurídico

Pedido: Generalização das Entidades PedidoPresencial e PedidoOnline, guarda atributos comuns de um Pedido				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Número	Armazena número como Indentificador do pedido	INT	Valores Positivos	* Chave Primária
Status	Armazena o status do pedido	VARCHAR(50)	VARCHAR(50)	* Não nulo
Valor	Armazena o valor ao total do pedido	DOUBLE	Valores Reais Positivos	* Não nulo
Data	Armazena a data em que foi demandado o pedido	TIMESTAMP	TIMESTAMP	* Não nulo
Forma de Pagamento	Armazena a forma de pagamento do Pedido	VARCHAR(50)	VARCHAR(50)	* Não nulo
CodigoCliente	Armazena codigo de um cliente que demandou pedido	INT	Valores Positivos	* Chave estrangeira que referencia o código do Cliente.

Tabela 04 – Relação Pedido

PedidoOnline: Expecialização da entidade Pedido, armazena dados de um pedido feito de forma online				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
NúmeroPedido	Armazena número de um pedido	INT	Valores Positivos	* Chave Primária
CEP	Armazena o CEP do cliente	CHARACTER(9)	Caracteres que representam Números	* Não nulo
Estado	Armazena o Estado do cliente	VARCHAR(255)	VARCHAR(255)	* Não nulo
Cidade	Armazena a cidade do cliente	VARCHAR(255)	VARCHAR(255)	* Não nulo
Rua	Armazena a Rua do cliente	VARCHAR(255)	VARCHAR(255)	* Não nulo
Número	Armazena o número da casa do cliente	VARCHAR(10)	VARCHAR(10)	* Não nulo
Complemento	Armazena o Complemento Informado pelo cliente	VARCHAR(255)	VARCHAR(255)	Nenhuma Restrição

Tabela 05 – Relação PedidoOnline

PedidoPresencial: Expecialização da entidade Pedido, guarda dados de um pedido feito presencialmente				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
NúmeroPedido	Armazena número de um pedido	INT	Valores Positivos	* Chave Primária
CódigoFuncionario	Código do Funcionario que registrou o pedido	INT	Valores Positivos	* Chave estrangeira que referencia o código do Funcionario.

Tabela 06 – Relação PedidoPresencial

Funcionario: Tabela que armazena informações dos funcionarios contratados				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Código	Armazena código como Indentificador do funcionario	INT	Valores Positivos	* Chave Primária
Nome	Armazena nome do funcionario	VARCHAR(255)	VARCHAR(255)	* Não nulo
Salario	Armazena o salario do funcionario	DOUBLE	Valores Reais Positivos	* Não nulo
AnoDeAdmissão	Armazena o ano de admissão do funcionario	DATE	DATE	* Não nulo

Tabela 07 – Relação Funcionario

Exemplar: Guarda dados dos exemplares de todos livros				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Código	Armazena código como Indentificador do exemplar	INT	Valores Positivos	* Chave Primária
Status	Armazena status do exemplar armazenado	VARCHAR(50)	VARCHAR(50)	* Não nulo
CódigoLivro	O código do livro que o exemplar referencia	INT	Valores Positivos	* Chave estrangeira que referencia o código do Livro.

Tabela 08 – Relação Exemplar

PedidoTemExemplar: Relaçionamento que armazena os dados de um dados de um pedido de exemplar				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
ID	Armazena o ID da relação	INT	Valores Positivos	* Chave Primária
Peço	Armazena o preço do exemplar vendido	DOUBLE	Valores Reais Positivos	* Não nulo
NúmeroPedido	Armazena número do pedido em que o exemplar foi relacionado	INT	Valores Positivos	* Chave estrangeira que referencia o número do Pedido.
CódigoExemplar	Armazena código do exemplar em que foi relacionado ao pedido	INT	Valores Positivos	* Chave estrangeira que referencia o código do Exemplar.

Tabela 09 – Relação PedidoTemExemplar

Livro: Armazena informações dos livros registrados pela livraria				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Código	Armazena código como Indentificador do livro	INT	Valores Positivos	* Chave Primária
Titulo	Armazena titulo do livro	VARCHAR(255)	VARCHAR(255)	* Não nulo
Preço	Armazena o preço do livro	DOUBLE	Valores Reais Positivos	* Não nulo
AnoDeLaçamento	Armazena o ano do lançamento do livro	CHARACTER(4)	Caracteres que representam Números	* Não nulo
CNPJEditora	Armazena o CNPJ da editora que publicou o livro	CHARACTER(14)	Caracteres que representam Números	* Chave estrangeira que referencia o CNPJ da Editora

Tabela 10 – Relação Livro

LivroAutor: Armazena dados de todos autores de cada livro				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CódigoLivro	O código do livro que o exemplar referencia	INT	Valores Positivos	* Chave Primária * Chave estrangeira que referencia o código do Livro.
Autor	Armazena nome do autor do livro	VARCHAR(255)	VARCHAR(255)	* Chave Primária

Tabela 11 – Relação LivroAutor

LivroCategoria: Armazena informações das categorias dos livros registrados pela livraria				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CódigoLivro	Armazena código do livro	INT	Valores Positivos	* Chave Primária * Chave estrangeira que referencia o código do Livro.
Categoria	Armazena uma das categorias do livro	VARCHAR(50)	VARCHAR(50)	* Chave Primária

Tabela 12 – Relação LivroCategoria

Editora: Informações das editoras dos livros adquiridos pela livraria				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CNPJ	Armazena codigo de uma editora	CHARACTER(14)	Caracteres que representam Números	* Chave Primária
Nome	Armazena nome da editora	VARCHAR(255)	VARCHAR(255)	* Não nulo

Tabela 13 – Relação Livro

Editora: Informações das editoras dos livros adquiridos pela livraria				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
CNPJ	Armazena codigo de uma editora	CHARACTER(14)	Caracteres que representam Números	* Chave Primária
Nome	Armazena nome da editora	VARCHAR(255)	VARCHAR(255)	* Não nulo

Tabela 14 – Relação Editora

TelefoneEditora: Relação que armazena os números de telefone das Editoras.				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
Telefone	Armazena um dos telefones da editora	CHARACTER(11)	Caracteres que representam Números	* Chave Primária
CNPJEditora	Armazena codigo da editora	CHARACTER(14)	Caracteres que representam Números	* Chave Primária * Chave estrangeira que referencia o CNPJ da Editora

Tabela 15 – Relação TelefoneEditora

E-mailEditora: Relação que armazena os e-mails usados pelas editoras.				
Atributo	**Descrição**	**Tipo**	**Domínio**	**Restrição**
E-mail	Armazena um dos telefones da editora	VARCHAR(255)	VARCHAR(255)	* Chave Primária
CNPJEditora	Armazena codigo da editora	CHARACTER(14)	Caracteres que representam Números	* Chave Primária * Chave estrangeira que referencia o CNPJ da Editora

Tabela 16 – Relação E-mailEditora

4. Modelo Físico

4.1. Scripts SQL para criar a consulta

Criação da Relação Cliente

```
CREATE TABLE Cliente (
      Codigo INT,
      Nome VARCHAR(255) NOT NULL,
      Email_01 VARCHAR(255) NOT NULL,
      Email_02 VARCHAR(255),
      Telefone_01 CHARACTER(11) NOT NULL,
      Telefone_02 CHARACTER(11),
      CONSTRAINT PK_Cliente PRIMARY KEY (Codigo)
);
```

Criação da Relação ClienteFísico

```
CREATE TABLE ClienteFisico (
      CodigoCliente INT,
      RG VARCHAR(50) UNIQUE NOT NULL,
      CPF CHARACTER(11) UNIQUE NOT NULL,
      CONSTRAINT PK_ClienteFisico PRIMARY KEY (CodigoCliente),
      CONSTRAINT FK_ClienteFisico FOREIGN KEY (CodigoCliente)
      REFERENCES Cliente(Codigo)
);
```

Criação da Relação ClienteJúridico

```
CREATE TABLE ClienteJuridico (
      CodigoCliente INT,
      CNPJ CHARACTER(14) UNIQUE NOT NULL,
      CONSTRAINT PK_ClienteJuridico PRIMARY KEY (CodigoCliente),
      CONSTRAINT FK_ClienteJuridico FOREIGN KEY (CodigoCliente)
      REFERENCES Cliente(Codigo)
);
```

Criação da Relação Pedido

```
CREATE TABLE Pedido (
      Numero INT,
      Status VARCHAR(50) NOT NULL,
      Valor DOUBLE PRECISION NOT NULL,
      Data TIMESTAMP NOT NULL,
      Forma_de_Pagamento VARCHAR(50) NOT NULL,
      CodigoCliente INT,
      CONSTRAINT PK_Pedido PRIMARY KEY (Numero),
      CONSTRAINT FK_Pedido FOREIGN KEY (CodigoCliente) REFERENCES
      Cliente(Codigo)
);
```

Criação da Relação PedidoOnline

```
CREATE TABLE PedidoOnline (
      NumeroPedido INT,
      CEP VARCHAR(50) NOT NULL,
      Estado VARCHAR(255) NOT NULL,
      Cidade VARCHAR(255) NOT NULL,
      Rua VARCHAR(255) NOT NULL,
      Numero VARCHAR(15) NOT NULL,
      Compremento VARCHAR(255),
      CONSTRAINT PK_PedidoOnline PRIMARY KEY (NumeroPedido),
      CONSTRAINT FK_PedidoOnline FOREIGN KEY (NumeroPedido)
      REFERENCES Pedido(Numero)
);
```

Criação da Relação Funcionario

```
CREATE TABLE Funcionario (
      Codigo INT,
      Nome VARCHAR(255) NOT NULL,
      Salario DOUBLE PRECISION NOT NULL,
      Ano_de_Admissao DATE NOT NULL,
      CONSTRAINT PK_Funcionario PRIMARY KEY (Codigo),
      CONSTRAINT Salario_Positivo_Funcionario CHECK (0 < Salario)
);
```

Criação da Relação PedidoPresencial

```
CREATE TABLE PedidoPresencial (
      NumeroPedido INT,
      CodigoFuncionario INT,
      CONSTRAINT PK_PedidoPresencial PRIMARY KEY (NumeroPedido),
      CONSTRAINT FK_01_PedidoPresencial FOREIGN KEY (NumeroPedido)
      REFERENCES Pedido(Numero),
```

```
    CONSTRAINT FK_02_PedidoPresencial FOREIGN KEY
    (CodigoFuncionario) REFERENCES Funcionario(Codigo)
);
```

Criação da Relação Editora

```
CREATE TABLE Editora (
    CNPJ CHARACTER(14),
    Nome VARCHAR(255) NOT NULL,
    CONSTRAINT PK_Editora PRIMARY KEY (CNPJ)
);
```

Criação da Relação TelefoneEditora

```
CREATE TABLE TelefoneEditora (
    CNPJEditora CHARACTER(14),
    Telefone CHARACTER(11),
    CONSTRAINT PK_TelefoneEditora PRIMARY KEY (Telefone,
    CNPJEditora),
    CONSTRAINT FK_TelefoneEditora FOREIGN KEY (CNPJEditora)
    REFERENCES Editora(CNPJ)
);
```

Criação da Relação EmailEditora

```
CREATE TABLE EmailEditora (
    CNPJEditora CHARACTER(14),
    Email VARCHAR(255),
    CONSTRAINT PK_EmailEditora PRIMARY KEY (Email, CNPJEditora),
    CONSTRAINT FK_EmailEditora FOREIGN KEY (CNPJEditora) REFERENCES
    Editora(CNPJ)
);
```

Criação da Relação Livro

```
CREATE TABLE Livro (
      Codigo INT,
      Titulo VARCHAR(255) NOT NULL,
      Preco DOUBLE PRECISION NOT NULL,
      Ano_de_Lancamento CHARACTER(4) NOT NULL,
      CNPJEditora CHARACTER(14),
      CONSTRAINT PK_Livro PRIMARY KEY (Codigo),
      CONSTRAINT FK_Livro FOREIGN KEY (CNPJEditora) REFERENCES
      Editora(CNPJ),
      CONSTRAINT Preco_Positivo_Livro CHECK (0 < preco)
);
```

Criação da Relação LivroAutor

```
CREATE TABLE LivroAutor (
      CodigoLivro INT,
      Autor VARCHAR(255) NOT NULL,
      CONSTRAINT PK_LivroAutor PRIMARY KEY (Autor, CodigoLivro),
      CONSTRAINT FK_LivroAutor FOREIGN KEY (CodigoLivro) REFERENCES
      Livro(Codigo)
);
```

Criação da Relação LivroCategoria

```
CREATE TABLE LivroCategoria (
      CodigoLivro INT,
      Categoria VARCHAR(50),
      CONSTRAINT PK_LivroCategoria PRIMARY KEY (Categoria,
      CodigoLivro),
      CONSTRAINT FK_LivroCategoria FOREIGN KEY (CodigoLivro)
      REFERENCES Livro(Codigo)
);
```

Criação da Relação Exemplar

```
CREATE TABLE Exemplar (
	Codigo INT,
	Status VARCHAR(50) NOT NULL,
	CodigoLivro INT,
	CONSTRAINT PK_Exemplar PRIMARY KEY (Codigo),
	CONSTRAINT FK_Exemplar FOREIGN KEY (CodigoLivro) REFERENCES
	Livro(Codigo)
);
```

Criação da Relação PedidoTemExemplar

```
CREATE TABLE PedidoTemExemplar (
	Id INT,
	Preco DOUBLE PRECISION NOT NULL,
	NumeroPedido INT,
	CodigoExemplar INT,
	CONSTRAINT PK_PedidoTemExemplar PRIMARY KEY (Id),
	CONSTRAINT FK_01_PedidoTemExemplar FOREIGN KEY (NumeroPedido)
	REFERENCES Pedido(Numero),
	CONSTRAINT FK_02_PedidoTemExemplar FOREIGN KEY (CodigoExemplar)
	REFERENCES Exemplar(Codigo),
	CONSTRAINT Preco_Positivo_PedidoTemExemplar CHECK (0 < preco)
);
```

4.2. Scripts SQL para a povoar as tuplas das tabelas

Povoando as tuplas da tabela Cliente

```
INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01,
telefone_02) VALUES (1, 'Vanessa Barher Pereira Silvino',
'vanessa@gmail.com', null, '11975305249', '11995550524');
```

```
INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01,
telefone_02) VALUES (2, 'Carlos Marotti Rios Nazare',
'carlos909@gmail.com', null, '89925571883', null);
```

```
INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01, telefone_02) VALUES (3, 'Matheus Rangel Baesso Chaves', 'matheus55@gmail.com', 'matheus55@hotmail.com', '92921627417', '92939230862');

INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01, telefone_02) VALUES (4, 'Miguel Malaquias Dores Pinho', 'miguel77@gmail.com', 'miguel77@hotmail.com', '83938687338', null);

INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01, telefone_02) VALUES (5, 'Humberto Paiva Mendonça Carmo', 'humberto3@gmail.com', null, '83927388577', null);

INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01, telefone_02) VALUES (6, 'Spilman Macedo Contabilidade ME', 'compras@contabilidade.spilman.com.br', 'controle.compras@contabilidade.spilman.com.br', '81934220121', '81987857785');

INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01, telefone_02) VALUES (7, 'Bueno Freitas Advocacia ME', 'compras@dvocacia.buenofreitas.com.br', 'controle.compras@dvocacia.buenofreitas.com.br', '92929436740', '97985017500');

INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01, telefone_02) VALUES (8, 'Thomaz Navega Empreendimentos EPP', 'compras@empreendimentos.thomaznavega.com.br', 'controle.compras@empreendimentos.thomaznavega.com.br', '69934464125', '69980512885');

INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01, telefone_02) VALUES (9, 'Rosa Falcão Planejamento EPP', 'compras@planejamento.rosafalcão.com.br', 'controle.compras@planejamento.rosafalcão.com.br', '15924716609', '16995714825');

INSERT INTO Cliente (codigo, nome, email_01, email_02, telefone_01, telefone_02) VALUES (10, 'Cunha Theodoro Consultoria LTDA', 'compras@consultoria.cunhatheodoro.com.br',
```

'controle.compras@consultoria.cunhatheodoro.com.br', '82932942597', '8280284778');

Povoando as tuplas da tabela ClienteFisico

```
INSERT INTO ClienteFisico (CodigoCliente, RG, CPF) values (1, '414205200', '11475773820');

INSERT INTO ClienteFisico (CodigoCliente, RG, CPF) values (2, '318817433', '66070253385');

INSERT INTO ClienteFisico (CodigoCliente, RG, CPF) values (3, '216815647', '85636138202');

INSERT INTO ClienteFisico (CodigoCliente, RG, CPF) values (4, '327925681', '52871215472');

INSERT INTO ClienteFisico (CodigoCliente, RG, CPF) values (5, '159548287', '31838288473');
```

Povoando as tuplas da tabela ClienteJuridico

```
INSERT INTO ClienteJuridico (CodigoCliente, CNPJ) values (6, '81025286000161');

INSERT INTO ClienteJuridico (CodigoCliente, CNPJ) values (7, '53890177000177');

INSERT INTO ClienteJuridico (CodigoCliente, CNPJ) values (8, '74860373000179');

INSERT INTO ClienteJuridico (CodigoCliente, CNPJ) values (9, '11451105000183');

INSERT INTO ClienteJuridico (CodigoCliente, CNPJ) values (10, '07019562000126');
```

Povoando as tuplas da tabela Funcionarios

```
insert into funcionario (Codigo, Nome, Salario, Ano_de_Admissao) values (1, 'Isaías Feijó Azevedo', 1300, '05-09-2018');


insert into funcionario (Codigo, Nome, Salario, Ano_de_Admissao) values (2, 'Ranya Franqueira Raminhos', 1300, '12-08-2018');
```

```
insert into funcionario (Codigo, Nome, Salario, Ano_de_Admissao)
values (3, 'Ravi Aranha Frade', 1450, '25-11-2016');


insert into funcionario (Codigo, Nome, Salario, Ano_de_Admissao)
values (4, 'Rodrigo Covilhã Mourão', 1150, '16-02-2018');


insert into funcionario (Codigo, Nome, Salario, Ano_de_Admissao)
values (5, 'Pietro Póvoas Feijó', 1300, '07-06-2021');
```

Povoando as tuplas da tabela Editora

```
INSERT INTO Editora (CNPJ, Nome) values ('45365284015101', 'Saraiva');

INSERT INTO Editora (CNPJ, Nome) values ('52322826099407', 'Companhia
da Letras');

INSERT INTO Editora (CNPJ, Nome) values ('61372276809708', 'FTD');

INSERT INTO Editora (CNPJ, Nome) values ('11382676909013', 'Leya');

INSERT INTO Editora (CNPJ, Nome) values ('91686776006807', 'Editora
Arqueiro');
```

Povoando as tuplas da tabela EmailEditora

```
INSERT INTO EmailEditora (CNPJEditora, Email) values
('45365284015101', 'www.saraiva.com.br/ajuda');

INSERT INTO EmailEditora (CNPJEditora, Email) values
('45365284015101', 'sac@saraiva.com.br');

INSERT INTO EmailEditora (CNPJEditora, Email) values
('45365284015101', 'vendas@saraiva.com.br');

INSERT INTO EmailEditora (CNPJEditora, Email) values
('52322826099407', 'sac@companhiadasletras.com.br,');

INSERT INTO EmailEditora (CNPJEditora, Email) values
('52322826099407', 'vendas@companhiadasletras.com.br,');

INSERT INTO EmailEditora (CNPJEditora, Email) values
('61372276809708', 'central.atendimento@ftd.com.br');

INSERT INTO EmailEditora (CNPJEditora, Email) values
('11382676909013', 'sac@leyabrasil.com.br');

INSERT INTO EmailEditora (CNPJEditora, Email) values
('11382676909013', 'vendas@leyabrasil.com.br');
```

```
INSERT INTO EmailEditora (CNPJEditora, Email) values ('91686776006807', 'atendimento@editoraarqueiro.com.br');
```

Povoando as tuplas da tabela TelefoneEditora

```
INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('45365284015101', '89937450393');

INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('45365284015101', '98935655148');

INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('45365284015101', '85938263723');

INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('52322826099407', '73920601359');

INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('52322826099407', '64938913128');

INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('61372276809708', '83932528242');

INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('11382676909013', '61931228675');

INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('11382676909013', '79925674094');

INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('11382676909013', '67933477969');

INSERT INTO TelefoneEditora (CNPJEditora, Telefone) values ('91686776006807', '53921858244');
```

Povoando as tuplas da tabela Livro

```
INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('1', 'ASSIM FALOU ZARATUSTRA', 37.90, 2018,'52322826099407');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('2', 'Teoria Geral do Direito', 54.95, 2019,'45365284015101');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('3', 'Lula', 47.99, 2021,'52322826099407');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('4', 'Os Miseráveis', 37.90, 1970,'61372276809708');
```

```
INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('5', 'O guia do mochileiro das galáxias', 30.99, 2007,'91686776006807');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('6', 'A Guerra dos Tronos: As Crônicas de Gelo e Fogo', 57.90, 2019,'11382676909013');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('7', 'A fúria dos reis: As Crônicas de Gelo e Fogo', 58.89, 2019,'11382676909013');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('8', 'A tormenta de espadas: As Crônicas de Gelo e Fogo', 61.95, 2019,'11382676909013');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('9', 'Sapiens (Nova edição): Uma breve história da humanidade', 4553, 2020,'52322826099407');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('10', 'O MUNDO DE SOFIA', 64.99, 2019,'52322826099407');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('11', 'Genealogia da moral ', 37.85, 2012,'52322826099407');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('12', 'A gaia ciência ', 20.81, 2012,'52322826099407');

INSERT INTO Livro (Codigo, Titulo, Preco, Ano_de_Lancamento, CNPJEditora ) values ('13', 'Freud (1901) - Obras completas volume 5: Psicopatologia da vida cotidiana e Sobre os sonhos ', 57.87, 2021,'52322826099407');
```

Povoando as tuplas da tabela LivroAutor

```
INSERT INTO LivroAutor (CodigoLivro, Autor) values (1, 'Friedrich Wilhelm Nietzsche');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (2, 'Ricardo Mauricio Freire Soares');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (3, 'Fernando Morais');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (4, 'Victor Hugo');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (5, 'Douglas Adams');
```

```
INSERT INTO LivroAutor (CodigoLivro, Autor) values (6, 'George R. R. Martin');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (7, 'George R. R. Martin');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (8, 'George R. R. Martin');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (9, 'Yuval Harari');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (10, 'Jostein Gaarder');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (11, 'Friedrich Wilhelm Nietzsche');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (12, 'Friedrich Wilhelm Nietzsche');

INSERT INTO LivroAutor (CodigoLivro, Autor) values (13, 'Sigmund Freud');
```

Povoando as tuplas da tabela LivroCategoria

```
INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (1, 'Filosofia');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (2, 'Juridico');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (3, 'Biografia');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (4, 'Romance');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (5, 'Ficção Científica');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (6, 'Fantasia Histórica');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (6, 'Romance');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (7, 'Fantasia Histórica');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (8, 'Fantasia Histórica');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (8, 'Romance');
```

```
INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (9, 'Livros de História da Civilização e Cultura');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (9, 'Ciencias');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (9, 'Biologia');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (10, 'Ficção Literária Literatura e Ficção');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (11, 'Filosofia');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (12, 'Filosofia');

INSERT INTO LivroCategoria (CodigoLivro, Categoria) values (13, 'Psicanálise');
```

Povoando as tuplas da tabela Exemplar

```
INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 1, 'Disponivel', 1);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 2, 'Disponivel', 1);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 3, 'Disponivel', 1);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 4, 'Disponivel', 1);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 5, 'Indisponivel', 1);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 6, 'Indisponivel', 1);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 7, 'Disponivel', 2);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 8, 'Disponivel', 2);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 9, 'Disponivel', 2);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 10, 'Disponivel', 2);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 11, 'Indisponivel', 2);
```

```
INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 12, 'Disponivel', 2);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 13, 'Disponivel', 2);


INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 14, 'Disponivel', 3);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 15, 'Disponivel', 3);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 16, 'Disponivel', 3);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 17, 'Indisponivel', 3);


INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 18, 'Disponivel', 4);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 19, 'Disponivel', 4);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 20, 'Indisponivel',                                                  4);


INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 21, 'Disponivel', 5);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 22, 'Disponivel', 5);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 23, 'Indisponivel', 5);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 24, 'Disponivel',5);


INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 25, 'Disponivel', 6);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 26, 'Disponivel', 6);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 27, 'Disponivel', 6);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 28, 'Indisponivel', 6);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 29, 'Disponivel', 6);
```

```
INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 30, 'Disponivel', 7);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 31, 'Disponivel', 7);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 32, 'Disponivel', 7);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 33, 'Indisponivel', 7);


INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 35, 'Disponivel', 8);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 36, 'Disponivel', 8);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 37, 'Indisponivel', 8);


INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 38, 'Disponivel', 9);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 39, 'Disponivel', 9);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 40, 'Indisponivel',                                                 9);


INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 41, 'Disponivel', 10);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 42, 'Disponivel', 10);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 43, 'Indisponivel', 10);


INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 44, 'Disponivel', 11);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 45, 'Disponivel', 11);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 46, 'Indisponivel', 11);
```

```
INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 47, 'Disponivel', 12);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 48, 'Disponivel', 12);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 49, 'Disponivel',                                                    12);


INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 50, 'Disponivel', 13);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 51, 'Disponivel', 13);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 52, 'Indisponivel', 13);

INSERT INTO Exemplar (Codigo, Status, CodigoLivro ) values ( 53, 'Disponivel', 13);
```

Povoando as tuplas da tabela Pedido

```
INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (1, 'Em andamento', 75.8, '2019-07-02 05:58:06', 'Cartão', 1);

INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (2, 'Em andamento', 178.74, '2022-01-07 07:37:00', 'Cartão', 7);

INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (3, 'Entregue', 37.9, '2000-03-30 08:02:12', 'Cartão', 10);

INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (4, 'Entregue', 47.99, '2022-06-01 13:48:54', 'Cartão', 9);

INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (5, 'Em andamento', 64.99, '2003-09-14 15:15:55', 'Cartão', 3);

INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (6, 'Entregue', 37.85, '2015-08-14 16:28:30', 'Dinheiro', 2);

INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (7, 'Entregue', 57.87, '2022-08-31 17:03:10', 'Cartão', 5);
```

```
INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (8, 'Entregue', 30.99, '2007-03-30 17:07:01', 'Cartão', 6);

INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (9, 'Entregue', 45.53, '2021-12-31 21:39:47', 'Dinheiro', 4);

INSERT INTO Pedido (Numero, Status, Valor, Data, Forma_de_pagamento, CodigoCliente) VALUES (10, 'Entregue', 54.95, '2019-12-31 23:09:00', 'Dinheiro', 8);
```

Povoando as tuplas da tabela PedidoPresencial

```
INSERT INTO PedidoPresencial (NumeroPedido, CodigoFuncionario) VALUES (4, 1);

INSERT INTO PedidoPresencial (NumeroPedido, CodigoFuncionario) VALUES (6, 3);

INSERT INTO PedidoPresencial (NumeroPedido, CodigoFuncionario) VALUES (8, 5);

INSERT INTO PedidoPresencial (NumeroPedido, CodigoFuncionario) VALUES (9, 4);

INSERT INTO PedidoPresencial (NumeroPedido, CodigoFuncionario) VALUES (10, 1);
```

Povoando as tuplas da tabela PedidoOnline

```
INSERT INTO PedidoOnline (NumeroPedido, Cep, Estado, Cidade, Rua, Numero, Compremento) VALUES (1, '95146-924', 'Sergipe', 'Mossoró', 'Vitória', '849', 'Lote 04');

INSERT INTO PedidoOnline (NumeroPedido, Cep, Estado, Cidade, Rua, Numero, Compremento) VALUES (2, '34075-361', 'Pernambuco', 'Santos', 'Théo', '77326', 'Casa 5');

INSERT INTO PedidoOnline (NumeroPedido, Cep, Estado, Cidade, Rua, Numero, Compremento) VALUES (3, '03705-247', 'Bahia', 'Várzea Grande', 'Lavínia Travessa', '051', 'Quadra 29');

INSERT INTO PedidoOnline (NumeroPedido, Cep, Estado, Cidade, Rua, Numero, Compremento) VALUES (5, '19467-112', 'Minas Gerais', 'Cuiabá', 'Isis', '5714', 'Apto. 633');

INSERT INTO PedidoOnline (NumeroPedido, Cep, Estado, Cidade, Rua, Numero, Compremento) VALUES (7, '03658-508', 'Alagoas', 'Carapicuíba', 'Albuquerque Travessa', '979', 'Casa 5');
```

Povoando as tuplas da tabela PedidoTemExemplar

```
INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (1, 37.9, 1, 5);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (2, 37.9, 1, 6);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (3, 61.95, 2, 37);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (4, 58.89, 2, 33);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (5, 57.9, 2, 28);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (6, 37.9, 3, 20);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (7, 47.99, 4, 17);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (8, 64.99, 5, 43);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (9, 37.85, 6, 46);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (10, 57.87, 7, 52);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (11, 30.99, 8, 23);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (12, 45.53, 9, 40);

INSERT INTO PedidoTemExemplar (Id, Preco, NumeroPedido, CodigoExemplar) VALUES (13, 54.95, 10, 11);
```

4.3. Criando Índices

```
CREATE INDEX index_cliente ON Cliente (Codigo);


CREATE INDEX index_clienteFisico ON ClienteFisico (CodigoCliente);


CREATE INDEX index_clienteJuridico ON ClienteJuridico (CodigoCliente);
```

```
CREATE INDEX index_pedido ON Pedido (Numero);

CREATE INDEX index_pedidoPresencial ON PedidoPresencial
(NumeroPedido);

CREATE INDEX index_pedidoOnline ON PedidoOnline (NumeroPedido);

CREATE INDEX index_funcionario ON Funcionario (Codigo);

CREATE INDEX index_exemplar ON Exemplar (Codigo);

CREATE INDEX index_pedidoTemExemplar ON PedidoTemExemplar (Id);

CREATE INDEX index_livro ON Livro (Codigo);

CREATE INDEX index_livroAutor ON LivroAutor (CodigoLivro);

CREATE INDEX index_livroCategoria ON LivroCategoria (CodigoLivro);

CREATE INDEX index_editora ON Editora (Cnpj);

CREATE INDEX index_telefoneEditora ON TelefoneEditora (CnpjEditora);

CREATE INDEX index_emailEditora ON EmailEditora (CnpjEditora);
```

4.4. Criando Visões

1. Recupera todas as informações a respeito dos livros além de quantos exemplares disponíveis para venda de cada

```
CREATE VIEW Estoque AS (
    SELECT liv.Codigo, liv.Titulo, liv.Preco,
    liv.Ano_de_lancamento, liv.CnpjEditora, Count(*) AS
    QuantDisponiveis
```

```
    FROM Exemplar exe JOIN Livro liv ON exe.CodigoLivro =
    liv.Codigo

    WHERE exe.Status = 'Disponivel'

    GROUP BY liv.Codigo, liv.Titulo, liv.Preco,
    liv.Ano_de_lancamento, liv.CnpjEditora

    ORDER BY Codigo

);
```

2. Recupera todas as informações a respeito dos Pedidos Online

```
CREATE VIEW PedidoOnlineDetalhado AS (

    SELECT ped.Numero AS NumeroPedido, ped.Status, ped.Valor,
    ped.Data, ped.Forma_de_pagamento, ped.CodigoCliente, po.Cep,
    po.Estado, po.Cidade, po.Rua, po.Numero AS NumeroResidencia,
    po.Compremento

    FROM Pedido ped JOIN PedidoOnline po ON ped.Numero =
    po.NumeroPedido

);
```

3. Recupera o código nome e quantidade de pedidos realizados por cada funcionário

```
CREATE VIEW PedidosPorFuncionarios AS (

    SELECT fun.Codigo, fun.Nome, Count(*) AS QuantPedidos

    FROM pedidoPresencial pp JOIN Funcionario fun ON
    pp.CodigoFuncionario = fun.Codigo

    GROUP BY fun.Codigo, fun.Nome

    ORDER BY QuantPedidos DESC, Codigo ASC

);
```

4.5. Criando Consultas

2 consultas envolvendo junções

Deve retornar o título dos livros e o nome do seu autor

```
SELECT livAut.Autor, liv.Titulo as Livro
FROM LivroAutor livAut
INNER JOIN Livro liv ON livAut.CodigoLivro = liv.Codigo;
```

Deve retornar o título e a quantidade de exemplares de cada livro

```
SELECT liv.Titulo, Count(*) AS QuantExemplar
FROM Livro liv
INNER JOIN Exemplar exp ON liv.Codigo = exp.CodigoLivro
GROUP BY liv.Titulo
ORDER BY QuantExemplar DESC;
```

2 consultas envolvendo comparação com valores nulos

Deve retornar o nome e o único telefone de cada cliente que possui apenas um telefone

```
SELECT Nome, Telefone_01
FROM Cliente
WHERE Telefone_02 is null;
```

Deve retornar o nome e o único email de cada cliente que possui apenas um email

```
SELECT Nome, Email_01
FROM Cliente
WHERE Email_02 is null;
```

2 consultas envolvendo buscas por substring

Deve retornar o nome de todos os clientes que começam o nome com a letra ‘C’

```
SELECT Nome
FROM Cliente
WHERE Nome like 'C%';
```

Deve retornar o nome de todas editoras que começam o nome com a letra ‘A’

```
SELECT Titulo
FROM Livro
WHERE Titulo LIKE 'A%';
```

2 consultas envolvendo ordenação

Deve retornar todos os títulos dos livros ordenados por ordem crescente

```
SELECT Titulo
FROM Livro
ORDER BY Titulo;
```

Deve retornar todos os nomes das livrarias por ordem decrescente

```
SELECT Nome
FROM Editora
ORDER BY Nome DESC;
```

2 consultas aninhadas

Deve retornar os dados comuns de clientes de todos os clientes Jurídicos

```
SELECT *
FROM Cliente
WHERE Codigo in (
      SELECT CodigoCliente
      FROM ClienteJuridico
);
```

Deve retornar os dados comuns de Pedido de todos os Pedidos Online

```
SELECT *
FROM Pedido
WHERE Numero IN (
      SELECT NumeroPedido
      FROM PedidoOnline
);
```

2 consultas aninhadas correlacionadas

Deve retornar os dados de todos funcionário que não possuem nenhuma venda de um pedido presencial

```
SELECT *
FROM Funcionario fun
WHERE NOT EXISTS (
      SELECT *
      FROM PedidoPresencial pedPres
      WHERE fun.Codigo = pedPres.CodigoFuncionario
);
```

Deve retornar os dados de todos os clientes jurídicos

```
SELECT *
FROM Cliente cli
WHERE EXISTS (
      SELECT *
      FROM ClienteJuridico cliJur
      WHERE cli.Codigo = cliJur.CodigoCliente
);
```

2 consultas usando operações de conjunto

Deve retornar todos os dados sobre os livros que foram lançados em 2018, 2019, 2020, 2021

```
SELECT *
FROM Livro
WHERE Ano_de_lancamento in ('2018', '2019', '2020', '2021');
```

Deve retornar todos os dados sobre os livros que foram lançados em 2019 e 2021

```
(
SELECT *
FROM Livro
WHERE Ano_de_lancamento = '2021'
UNION
SELECT *
FROM Livro
WHERE Ano_de_lancamento = '2019'
);
```

2 consultas usando funções agregadas

Deve retornar o nome do autor e a quantidade de livros por cada autor, ordenado pela quantidade

```
SELECT Autor, Count(*) as QuantLivros
FROM LivroAutor
GROUP BY Autor
ORDER BY Count(*) DESC;
```

Deve retornar o título e o preço do livro mais caro

```
SELECT titulo, Preco
FROM Livro
WHERE Preco in (
      SELECT Max(Preco)
      FROM Livro
);
```

2 consultas usando agrupamentos dentre as quais pelo menos deve usar filtragem de grupo

Deve retornar o nome de todas as editoras que possuem mais de um e-mail de contato

```
SELECT edt.Nome
FROM Editora edt
INNER JOIN EmailEditora emEdt ON edt.Cnpj = emEdt.CnpjEditora
GROUP BY edt.Cnpj
HAVING count(emEdt.Email) > 1;
```

Deve retornar o Ano de lançamento e a quantidade de livros registrados lançados nesse ano

```
SELECT Ano_de_lancamento,
Count(Titulo) AS QuantLivros
FROM Livro
GROUP BY Ano_de_lancamento
ORDER BY QuantLivros DESC;
```

4.6. Criando Procedimentos Armazenados

1. Função criada para retornar a quantidade de pedidos registrados por um funcionário

```
CREATE OR REPLACE FUNCTION NumeroDePedidosPorFuncionario(INTEGER)
RETURNS INTEGER AS
$$
DECLARE
      codFuncionario ALIAS FOR $1;
      quantPedidos INTEGER;
BEGIN
```

```
      SELECT INTO quantPedidos Count(*)
      FROM PedidoPresencial pp
      WHERE pp.CodigoFuncionario = codFuncionario;
      RETURN quantPedidos;
END
$$ LANGUAGE PLPGSQL;
```

2. Recupera quantas vezes o livro determinado pelo código foi vendido

```
CREATE OR REPLACE FUNCTION NumeroDeVendas(INT)
RETURNS INT AS
$$
DECLARE
      codLivro ALIAS FOR $1;
      resultado INT;
BEGIN
      SELECT INTO resultado COUNT(*)
      FROM PedidoTemExemplar pte, Exemplar exe, Livro liv
      WHERE
      (pte.CodigoExemplar = E.Codigo AND exe.CodigoLivro =
      liv.Codigo) AND
      liv.Codigo = codLivro;
      RETURN resultado;
END
$$ LANGUAGE PLPGSQL;
```

2. Recuperar a quantidade de livros de uma editora, determinada pelo seu código

```
CREATE OR REPLACE FUNCTION NumeroDeLivrodeEditora(CHARACTER(14))
RETURNS INTEGER AS
$$
DECLARE
      CnpjEntrada ALIAS FOR $1;
      Resultado INTEGER;
```

```
BEGIN
      SELECT INTO Resultado COUNT(*) AS QuantLivros
      FROM Editora edt
      INNER JOIN livro liv ON edt.Cnpj = liv.CnpjEditora
      WHERE edt.Cnpj = CnpjEntrada
      GROUP BY edt.Cnpj;
      RETURN resultado;
END
$$ LANGUAGE PLPGSQL;
```

4.6. Criando Gatilhos

1. Atualiza status de exemplares adicionados a vendas indisponíveis

```
CREATE OR REPLACE FUNCTION AtualizarStatusExemplar()
RETURNS TRIGGER AS
$$
BEGIN
      UPDATE Exemplar
      SET Status = 'Indisponivel'
      WHERE Codigo = NEW.CodigoExemplar;
      RETURN NULL;
END
$$ LANGUAGE PLPGSQL;

CREATE TRIGGER AtualizarStatusExemplar
AFTER INSERT ON PedidoTemExemplar
FOR EACH ROW
EXECUTE PROCEDURE AtualizarStatusExemplar();
```

2. Atualiza status do pedido ao adicionar novo pedidoPresencial

```
CREATE OR REPLACE FUNCTION AtualizarStatusPedido()
```

```
RETURNS TRIGGER AS
$$
BEGIN
      UPDATE Pedido
      SET Status = 'Entregue'
      WHERE NEW.NumeroPedido = Numero;
      RETURN NULL;
END
$$ LANGUAGE PLPGSQL;

CREATE TRIGGER AtualizaStatusPedido
AFTER INSERT ON PedidoPresencial
FOR EACH ROW
EXECUTE PROCEDURE AtualizarStatusPedido();
```

3. Atualiza forma de pagamento do pedido ao adicionar novo pedidoOnline

```
CREATE OR REPLACE FUNCTION AtualizarFormaDePagamentoPedido()
RETURNS TRIGGER AS
$$
BEGIN
      UPDATE Pedido
      SET Forma_de_pagamento = 'Cartão'
      WHERE NEW.NumeroPedido = Numero;
      RETURN NULL;
END
$$ LANGUAGE PLPGSQL;

CREATE TRIGGER AtualizaFormaDePagamentoPedido
AFTER INSERT ON PedidoOnline
FOR EACH ROW
EXECUTE PROCEDURE AtualizarFormaDePagamentoPedido();
```