



## Vamos manter informações

Israel dos Santos Hamdan D'Araujo, 202303592086

**Inserir aqui o Campus:** Polo Centro - Jequié (BA)

**Nível 2 – Vamos manter informações, turma 2023.1, semestre 2024.1**

### Objetivo da Prática

O objetivo desta missão prática foi a criação de um banco de dados usando o Microsoft SQL Server, neste caso, foi usado inicialmente o SQL Server Express, mas posteriormente foi feita uma alteração para o SQL Server Developer, pois a versão anterior apresentava limitações para o seguimento de outras missões práticas

### 1º Procedimento | Criando o Banco de Dados

Inserir neste campo, de forma organizada, todos os códigos do roteiro do 1º Procedimento da Atividade Prática, os resultados da execução do código e a Análise e Conclusão:

- Criando tabela de pessoas

```
CREATE TABLE Pessoas(
    idPessoa INT NOT NULL,
    Nome VARCHAR(255) NOT NULL,
    Logradouro VARCHAR(255),
    Cidade VARCHAR(255),
    Estado CHAR(2) NOT NULL,
    Telefone VARCHAR(11) NOT NULL,
    Email VARCHAR(255) NOT NULL
    PRIMARY KEY([idPessoa])
    CONSTRAINT DF_Pessoa_idPessoa DEFAULT (NEXT VALUE FOR seq_pessoa)
);
```

Mensagens

Comandos concluídos com êxito.

Horário de conclusão: 2024-05-06T00:11:19.1774569-03:00

- Criando tabelas de pessoas físicas

```
CREATE TABLE PessoasFisicas (  
    idPessoaFisica INT PRIMARY KEY NOT NULL,  
    CPF VARCHAR(11) UNIQUE NOT NULL,  
    CONSTRAINT FK_PessoaFisica_Pessoa FOREIGN KEY (idPessoaFisica) REFERENCES Pessoas(idPessoa)  
);
```

Mensagens  
Comandos concluídos com êxito.  
Horário de conclusão: 2024-05-06T00:12:18.0678086-03:00

- Criando tabela de pessoas juridicas

```
CREATE TABLE PessoasJuridicas (  
    idPJ INT PRIMARY KEY NOT NULL,  
    CNPJ VARCHAR(11) UNIQUE NOT NULL,  
    CONSTRAINT FK_PessoaJuridica_Pessoa FOREIGN KEY (idPJ) REFERENCES Pessoas(idPessoa)  
);
```

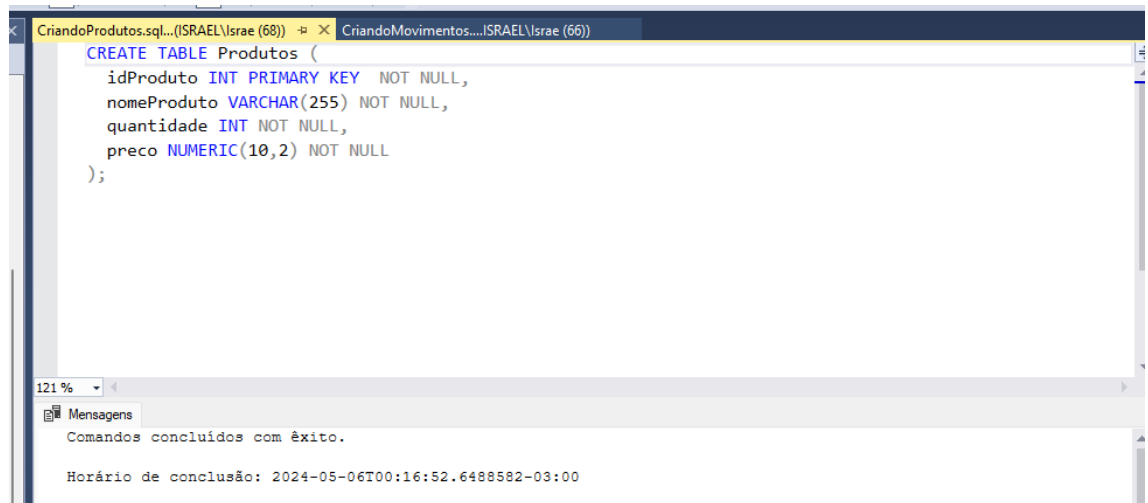
Mensagens  
Comandos concluídos com êxito.  
Horário de conclusão: 2024-05-06T00:13:44.6771176-03:00

- Criando tabela de usuários

```
CREATE TABLE Usuarios (  
    idUser INT PRIMARY KEY NOT NULL,  
    loginUser VARCHAR(20) UNIQUE NOT NULL,  
    Senha VARCHAR(30) NOT NULL,  
    CONSTRAINT C_User Foreign key (idUser) REFERENCES Pessoas(idPessoa)  
);
```

Mensagens  
Comandos concluídos com êxito.  
Horário de conclusão: 2024-05-06T00:14:31.3225771-03:00

- Criação da tabela de produtos



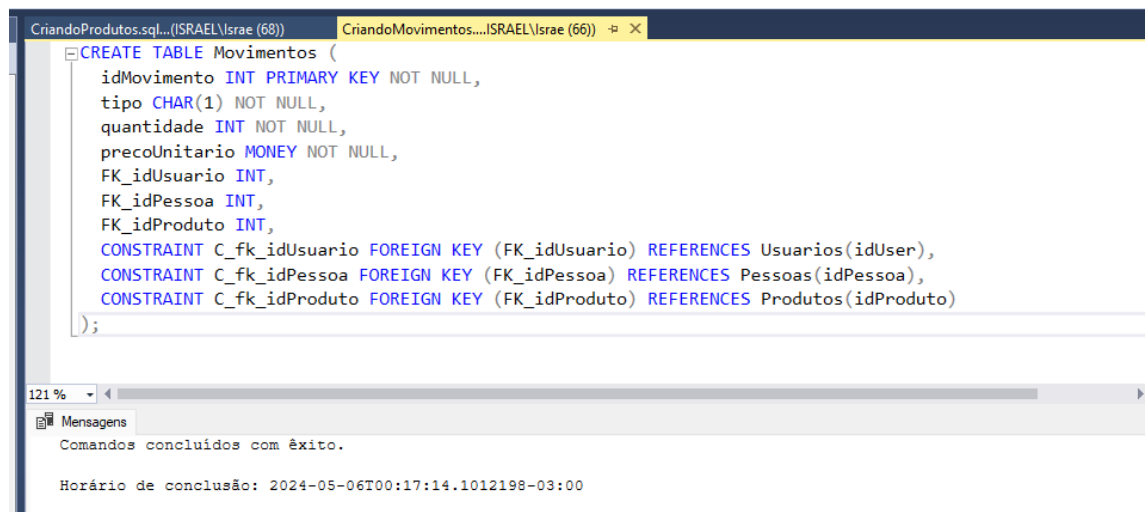
The screenshot shows a SQL Server Enterprise Manager window with two tabs: 'CriandoProdutos.sql... (ISRAEL\Israe (68))' and 'CriandoMovimentos....ISRAEL\Israe (66))'. The active tab displays the following SQL command:

```
CREATE TABLE Produtos (  
    idProduto INT PRIMARY KEY NOT NULL,  
    nomeProduto VARCHAR(255) NOT NULL,  
    quantidade INT NOT NULL,  
    preco NUMERIC(10,2) NOT NULL  
);
```

Below the command editor, the 'Mensagens' (Messages) pane shows the following output:

```
Comandos concluídos com êxito.  
  
Horário de conclusão: 2024-05-06T00:16:52.6488582-03:00
```

- Criado tabela de movimentos



The screenshot shows the same SQL Server Enterprise Manager window. The active tab now displays the following SQL command:

```
CREATE TABLE Movimentos (  
    idMovimento INT PRIMARY KEY NOT NULL,  
    tipo CHAR(1) NOT NULL,  
    quantidade INT NOT NULL,  
    precoUnitario MONEY NOT NULL,  
    FK_idUsuario INT,  
    FK_idPessoa INT,  
    FK_idProduto INT,  
    CONSTRAINT C_fk_idUsuario FOREIGN KEY (FK_idUsuario) REFERENCES Usuarios(idUser),  
    CONSTRAINT C_fk_idPessoa FOREIGN KEY (FK_idPessoa) REFERENCES Pessoas(idPessoa),  
    CONSTRAINT C_fk_idProduto FOREIGN KEY (FK_idProduto) REFERENCES Produtos(idProduto)  
);
```

The 'Mensagens' pane shows the following output:

```
Comandos concluídos com êxito.  
  
Horário de conclusão: 2024-05-06T00:17:14.1012198-03:00
```

- Criando sequencia:

The screenshot shows a SQL query window with the following content:

```
SQLQuery11.sql - ISR...(ISRAEL\Israe (58))* Sequencia.sql - ISRA...(ISRAEL\Israe (51))
```

```
CREATE SEQUENCE seq_pessoa
AS INT
START WITH 1
INCREMENT BY 1
MINVALUE 1
MAXVALUE 999999
NO CYCLE;
```

Below the query, a status bar indicates the zoom level is 121 %.

A message pane at the bottom shows the following text:

Mensagens  
Comandos concluídos com êxito.  
Horário de conclusão: 2024-05-06T15:03:04.8449115-03:00

a) Como são implementadas as diferentes cardinalidades, basicamente 1x1, 1xN, ou NxN em um banco de dados relacional?

Resposta:

- Cardinalidade 1x1: É um relacionamento direto entre tabelas de um banco de dados, um bom exemplo disso é uma tabela de clientes e uma tabela de endereço de entrega, onde cada cliente possui um endereço e cada endereço possui apenas 1 cliente
- Cardinalidade 1xN: É um relacionamento de um para muitos, esse relacionamento ocorre quando temos que implementar um dado associado a vários outros dados, por exemplo, uma tabela de departamentos e uma tabela de funcionários, onde cada departamento tem N funcionários trabalhando dentro dele. Representamos isso com uma chave estrangeira (*foreign key*) na tabela de funcionários, relacionando a chave primária (*primary key*) do departamento
- Cardinalidade NxN: Ocorre quando precisamos implementar uma relação de muitos dados com outros muitos dados, por exemplo, uma tabela de

disciplinas e uma tabela de alunos, muitos alunos podem estar relacionados a várias disciplinas e cada disciplina pode ter vários alunos matriculados nela. Para fazer isso precisamos de uma tabela intermediária que juntará as duas tabelas em uma só, esta por sua vez contém chaves estrangeiras que referenciam as tabelas relacionadas

b) Que tipo de relacionamento deve ser utilizado para representar o uso de herança em bancos de dados relacionais?

- Herança única – tabela por classe: Cada classe de herança para uma tabela no banco de dados, as tabelas filhas contêm apenas atributos específicos de cada classe. Nesta missão foi utilizado esse tipo de herança, por exemplo a tabela pessoas (tabela mãe) e as tabelas de pessoas físicas e jurídicas, que são as tabelas filhas, nas tabelas filhas há uma chave estrangeira (*foreign key*) referenciando a tabela mãe, que neste caso era o id
- Herança múltipla – tabela por subclasse: Cada classe na hierarquia é herança mapeada para uma tabela no banco de dados, incluindo os atributos herdados da classe pai.
- Tabela de junção: Cada classe tem sua própria tabela no banco de dados e o relacionamento é feito por uma tabela de junção para mapear os relacionamentos entre elas

c) Como o SQL Server Management Studio permite a melhoria para produtividade nas tarefas relacionadas?

Resposta: Uma ferramenta própria para o gerenciamento de banco de dados é fundamental para o desenvolvimento desta missão e no dia a dia de um profissional de banco de dados, ou de um desenvolvedor backend, pois trazem ferramentas que auxiliam muito a produtividade para implementação de recursos de segurança, criação de tabelas de maneira ágil e consulta das tabelas.

## 2º Procedimento | Alimentando a Base

Inserir neste campo, **de forma organizada**, todos os códigos do roteiro do 2º Procedimento da Atividade Prática, os resultados da execução do código e a Análise e Conclusão:

- Inserindo dados na tabela Pessoas:

```
INSERT INTO Pessoas (idPessoa, Nome, Logradouro, Cidade, Estado, Telefone, Email)
VALUES
(1, 'João Silva', 'Rua', 'São Luis', 'Ma', '11999999999', 'joaosilva@email.com'),
(2, 'Maria Souza', 'Avenida', 'Petropolis', 'RJ', '21999999999', 'mariasouza@email.com'),
(3, 'Pedro Oliveira', 'Condominio', 'Belo Horizonte', 'MG', '31999999999', 'pedrooliveira@email.com'),
(4, 'Budwiser', 'Rua', 'São Luis', 'Ma', '11999999999', 'budwiser@email.com'),
(5, 'Ambev', 'Avenida', 'Petropolis', 'RJ', '21999999999', 'ambev@email.com'),
(6, 'Heiniken', 'Parque', 'Belo Horizonte', 'MG', '31999999999', 'Heiniken@email.com'),
(7, 'Corona', 'Parque', 'Belo Horizonte', 'MG', '31999999999', 'Corona@email.com'),
(8, 'Coca-cola', 'Fabrica', 'Uberlandia ', 'MG', '31336541325', 'cocacolabr@gmail.com');
```

%

Mensagens

(8 linhas afetadas)

Horário de conclusão: 2024-05-06T14:37:43.6075265-03:00

- Inserindo dados na tabela de PessoasJuricas:

```
INSERT INTO PessoasJuridicas (idPJ, CNPJ)
VALUES
(4, '2151321531'),
(5, '1232151546'),
(6, '1516541351'),
(7, '5531546535'),
(8, '58796413561');
```

%

Mensagens

(5 linhas afetadas)

Horário de conclusão: 2024-05-06T14:38:39.7799231-03:00

- Inserindo dados na tabela de pessoas físicas:

```
INSERT INTO PessoasFisicas (idPessoaFisica , CPF)
VALUES
(1, '41412651561'),
(2, '51651151516'),
(3, '7784116516');
```

21 %

Mensagens

(3 linhas afetadas)

Horário de conclusão: 2024-05-06T14:40:21.7366652-03:00

- Inserindo dados na tabela de usuários:

```
INSERT INTO Usuarios (idUser, loginUser, Senha)
Values
(1, 'Op1', 'op1'),
(2, 'Op2', 'op2');
```

1 %

Mensagens

(2 linhas afetadas)

Horário de conclusão: 2024-05-06T14:41:28.1567437-03:00

- Inserindo dados na tabela de produtos:

```
INSERT INTO Produtos (idProduto, nomeProduto, quantidade, preco)
VALUES
(1, 'Cerveja Budwiser', 800, 5.00),
(2, 'Cerveja Heiniken', 800, 7.80),
(3, 'Cerveja Heiniken zero Álcool', 800, 7.80),
(4, 'Cerveja Corona', 800, 8.00),
(5, 'Coca-cola 3L', 1000, 12.00);
```

11 %

Mensagens

(5 linhas afetadas)

Horário de conclusão: 2024-05-06T14:42:44.3146657-03:00

- Inserindo dados na tabela Movimentos:

```
INSERT INTO Movimentos (idMovimento, tipo, quantidade, precoUnitario, FK_idUsuario, FK_idPessoa, FK_idProduto)
VALUES
(1, 'S', 10, 5.00, 1, 1, 1),
(2, 'E', 100, 2.00, 1, 7, 4);
```

Mensagens

(2 linhas afetadas)

Horário de conclusão: 2024-05-06T14:44:40.8290637-03:00

- Exibindo dados da tabela pessoas físicas



ExibindoPessoasFisic...(ISRAEL\Israe (63)) X ExibindoPessoasJuri...(ISRAEL\Israe (52))

```

SELECT
    PF.idPessoaFisica,
    PF.CPF,
    P.Nome,
    P.Logradouro,
    P.Cidade,
    P.Estado,
    P.Telefone,
    P.Telefone
From PessoasFisicas PF
JOIN Pessoas P ON PF.idPessoaFisica = P.idPessoa;

```

121 %

Resultados Mensagens

	idPessoaFisica	CPF	Nome	Logradouro	Cidade	Estado	Telefone	Telefone
1	1	41412651561	João Silva	Rua	São Luis	Ma	11999999999	11999999999
2	2	51651151516	Maria Souza	Avenida	Petropolis	RJ	21999999999	21999999999
3	3	7784116516	Pedro Oliveira	Condominio	Belo Horizonte	MG	31999999999	31999999999

- Exibindo dados da tabela de pessoas juridicas:

ExibindoPessoasJuri...(ISRAEL\Israe (52)) X

```

Select
    PJ.idPJ,
    PJ.CNPJ,
    P.Nome,
    P.Logradouro,
    P.Cidade,
    P.Estado,
    P.Telefone,
    P.Telefone
From PessoasJuridicas PJ JOIN Pessoas P ON PJ.idPJ = P.idPessoa;

```

121 %

Resultados Mensagens

	idPJ	CNPJ	Nome	Logradouro	Cidade	Estado	Telefone	Telefone
1	5	1232151546	Ambev	Avenida	Petropolis	RJ	21999999999	21999999999
2	6	1516541351	Heiniken	Parque	Belo Horizonte	MG	31999999999	31999999999
3	4	2151321531	Budwiser	Rua	São Luis	Ma	11999999999	11999999999
4	7	5531546535	Corona	Parque	Belo Horizonte	MG	31999999999	31999999999
5	8	58796413561	Coca-cola	Fabrica	Uberlandia	MG	31336541325	31336541325

- Exindo entradas

MovimentosDeVenda...ISRAEL\Israe (52))\*    MovimentosDeEntra...ISRAEL\Israe (51))\*

```

Select
    M.idMovimento,
    P.nomeProduto as Produto,
    PJ.Nome as Fornecedor,
    M.quantidade,
    M.precoUnitario,
    M.quantidade * M.precoUnitario as Total
From Movimentos M
INNER JOIN Produtos P ON M.FK_idProduto = P.idProduto
INNER JOIN Pessoas PJ ON M.FK_idPessoa = PJ.idPessoa
Where M.tipo = 'E'

```

121 %

Resultados    Mensagens

	idMovimento	Produto	Fornecedor	quantidade	precoUnitario	Total
1	2	Cerveja Corona	Corona	100	2.00	200.00

- Exibindo de Saida (Venda):

MovimentosDeVenda...ISRAEL\Israe (52))\*    MovimentosDeEntra...ISRAEL\Israe (51))\*

```

Select
    M.idMovimento,
    P.nomeProduto as Produto,
    PF.Nome as Comprador,
    M.quantidade,
    M.precoUnitario,
    M.quantidade * M.precoUnitario as total
From Movimentos M
INNER JOIN Produtos P ON M.FK_idProduto = P.idProduto
INNER JOIN Pessoas PF ON M.FK_idPessoa = PF.idPessoa
Where M.tipo = 'S';

```

121 %

Resultados    Mensagens

	idMovimento	Produto	Comprador	quantidade	precoUnitario	total
1	1	Cerveja Budwiser	João Silva	10	5,00	50,00

- Operadores que não compraram:

Naocompraram.sql -...ISRAEL\Israe (75))\* X MovimentosDeVenda...ISRAEL\Israe (52))\* MovimentosDeEntrada

```
Select
    U.idUser,
    U.loginUser
From Usuarios U
Left join Movimentos M on U.idUser = M.FK_idUsuario
Where
    M.idMovimento IS NULL OR M.tipo <> 'S';
```

121 %

Resultados Mensagens

	idUser	loginUser
1	1	Op1
2	2	Op2

- Soma das entradas por usuário

SomaDeVendasPorUs...ISRAEL\Israe (65)) SomaDeEntradasPor...ISRAEL\Israe (62)) X Naocompraram

```
Select
    U.idUser as id,
    U.loginUser as nome,
    SUM(M.precoUnitario) as Soma
From Usuarios U
Left Join Movimentos M on U.idUser = M.FK_idUsuario
Where M.tipo = 'E'
Group by U.idUser, U.loginUser
```

121 %

Resultados Mensagens

	id	nome	Soma
1	1	Op1	2.00

- Soma das saídas (vendas) por usuário

SomaDeVendasPorUs...ISRAEL\Israe (65))\* X SomaDeEntradasPor...ISRAEL\Israe (62)) NaoCompra

```

Select
    U.idUser as id,
    U.loginUser as nome,
    SUM(M.precoUnitario) as Soma
From Usuarios U
Left Join Movimentos M on U.idUser = M.FK_idUsuario
Where M.tipo = 'S'
Group by U.idUser, U.loginUser

```

121 %

Resultados Mensagens

	id	nome	Soma
1	1	Op1	5,00

- Media das vendas por produto

MediaDasVendas.sql...(ISRAEL\Israe (67))\* X SomaDeVendasPorUs...ISRAEL\Israe (65))\* SomaDeEntradasPor...ISRAEL\Israe (62))

```

Select
    M.Fk_idProduto as id,
    P.nomeProduto as produto,
    SUM(M.quantidade * M.precoUnitario) / SUM(M.quantidade) as Media
From Movimentos M
inner join Produtos P on M.FK_idProduto = idProduto
Where M.tipo = 'S'
Group by M.FK_idProduto, P.nomeProduto

```

121 %

Resultados Mensagens

	id	produto	Media
1	1	Cerveja Budwiser	5,00

a) Quais as diferenças no uso de sequence e identity?

Resposta: *Sequence* é um objeto no DB, que é responsável por gerar uma sequência de números exclusivos, é mais flexível e pode ser compartilhado entre

várias tabelas. *Identity* é uma propriedade passada quando a tabela é criada no MSSQL também gera valores automaticamente, mas é para a coluna associada e são exclusivos para aquela tabela

- b) Qual a importância de usar chaves estrangeiras para a consistência do banco?

Resposta: As *foreign key* também chamadas de chave estrangeira, são fundamentais para a consistência do banco de dados, pois relacionam diretamente o item da tabela filha com a tabela pai

- c) Quais operadores SQL pertencem à álgebra relacional e quais são definidos no cálculo relacional?

**Resposta:**

Álgebra relacional: No SQL existem algumas palavras-chave que fazem o relacionamento por exemplo o *Where*, que diz aonde especificamente aquela instrução deve ocorrer, o Join para juntar tabelas, o *Union* que une os dados de duas ou mais consultas e o *Intersect* é usado para retornar apenas registros que são comuns a duas ou mais consultas.

Cálculo relacional: Os operadores incluem projeção ( $\pi$ ), seleção ( $\sigma$ ), junção natural ( $\bowtie$ ), junção externa ( $\bowtie$ ,  $\ltimes$ ,  $\rtimes$ ), porém o SQL padrão não adota explicitamente o cálculo relacional, mas muitos de seus operadores tem seus equivalentes na álgebra relacional

- d) Como é feito o agrupamento em consultas e qual o requisito obrigatório?

Resposta: É feito usando o *GROUP BY*, seguido das colunas que precisam ser agrupadas, por exemplo *GROUP BY P.idPessoa, PO.idProduto*. É obrigatório para usar o *Group by* e que todas as colunas selecionadas na lista que não são agregadas devem estar presentes na cláusula *GROUP BY*, garantindo que cada linha esteja claramente associada

## Conclusão

Elabore uma análise crítica da sua Missão Prática.