



# Estácio

Campus: Polo Centro – São Bernardo do Campo

Curso: Desenvolvimento Full-Stack

Disciplina: Digital

Turma: Virtual

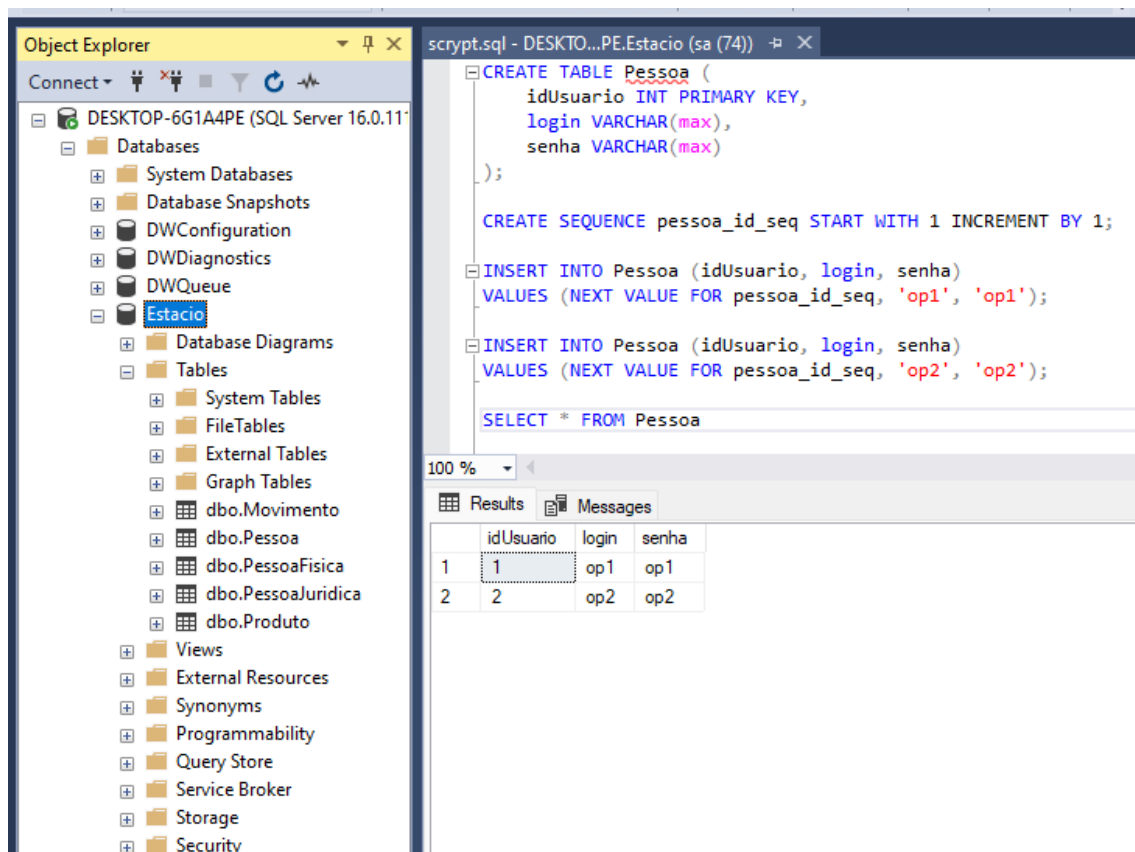
Semestre letivo: 2024.1

Integrantes: Fabio de Almeida

## Criando o Banco de Dados

Esta prática tem como objetivo aprofundar o conhecimento dos alunos em SQL e em conceitos avançados de bancos de dados relacionais. Serão exploradas diferenças entre o uso de *sequences* e *identity*, a importância das chaves estrangeiras para a consistência do banco, os operadores de álgebra e do cálculo relacional, bem como o agrupamento em consultas SQL.

A seguir, são apresentados os códigos solicitados neste roteiro de aula juntamente com os resultados de sua execução.



The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Object Explorer' pane shows the 'Estacio' database selected under 'Databases'. The 'Tables' folder is expanded, showing a list of tables including 'dbo.Pessoa'. The main window shows a script named 'script.sql' with the following SQL code:

```
CREATE TABLE Pessoa (  
    idUsuario INT PRIMARY KEY,  
    login VARCHAR(max),  
    senha VARCHAR(max)  
);  
  
CREATE SEQUENCE pessoa_id_seq START WITH 1 INCREMENT BY 1;  
  
INSERT INTO Pessoa (idUsuario, login, senha)  
VALUES (NEXT VALUE FOR pessoa_id_seq, 'op1', 'op1');  
  
INSERT INTO Pessoa (idUsuario, login, senha)  
VALUES (NEXT VALUE FOR pessoa_id_seq, 'op2', 'op2');  
  
SELECT * FROM Pessoa
```

Below the script, the 'Results' pane shows the output of the query, displaying two rows of data:

	idUsuario	login	senha
1	1	op1	op1
2	2	op2	op2

Object Explorer

Connect

DESKTOP-6G1A4PE (SQL Server 16.0.11)

- Databases
  - System Databases
  - Database Snapshots
  - DWConfiguration
  - DWDiagnostics
  - DWQueue
  - Estacio
    - Database Diagrams
    - Tables
      - System Tables
      - FileTables
      - External Tables
      - Graph Tables
      - dbo.Movimento
      - dbo.Pessoa
      - dbo.PessoaFisica
      - dbo.PessoaJuridica
      - dbo.Produto
    - Views
    - External Resources

script.sql - DESKTO...PE.Estacio (sa (74))

```
CREATE TABLE Produto (  
    idProduto INT PRIMARY KEY,  
    nome VARCHAR(MAX),  
    quantidade INT,  
    precoVenda DECIMAL(10, 2)  
)  
  
CREATE SEQUENCE produto_id_seq START WITH 1 INCREMENT BY 1;  
  
INSERT INTO Produto (idProduto, nome, quantidade, precoVenda)  
VALUES (NEXT VALUE FOR produto_id_seq, 'Manga', 800, 4.00)  
  
SELECT * FROM Produto
```

100 %

Results Messages

	idProduto	nome	quantidade	precoVenda
1	1	Banana	100	5.00
2	2	Laranja	500	2.00
3	3	Manga	800	4.00

Object Explorer

Connect

DESKTOP-6G1A4PE (SQL Server 16.0.11)

- Databases
  - System Databases
  - Database Snapshots
  - DWConfiguration
  - DWDiagnostics
  - DWQueue
  - Estacio
    - Database Diagrams
    - Tables
      - System Tables
      - FileTables
      - External Tables
      - Graph Tables
      - dbo.Movimento
      - dbo.Pessoa
      - dbo.PessoaFisica
      - dbo.PessoaJuridica
      - dbo.Produto
    - Views
    - External Resources
    - Synonyms

script.sql - DESKTO...PE.Estacio (sa (74))

```
CREATE TABLE PessoaFisica (  
    idPf INT PRIMARY KEY,  
    nome varchar(max),  
    logradouro varchar(max),  
    cidade varchar(max),  
    estado varchar(2),  
    telefone varchar(11),  
    email varchar(max),  
    cpf varchar(max),  
    FOREIGN KEY (idPf) REFERENCES Pessoa(idUsuario)  
)  
  
INSERT INTO PessoaFisica (idPf, nome, logradouro, cidade, estado, telefone, email, cpf)  
VALUES (1, 'João', 'Rua 12, casa 3, Quitanda', 'Riacho do Sul', 'PA', '1111-1111', 'joao@riacho.com', '111111111111')  
  
select * from PessoaFisica
```

100 %

Results Messages

	idPf	nome	logradouro	cidade	estado	telefone	email	cpf
1	1	João	Rua 12, casa 3, Quitanda	Riacho do Sul	PA	1111-1111	joao@riacho.com	111111111111

Object Explorer

Connect

DESKTOP-6G1A4PE (SQL Server 16.0.11)

- Databases
  - System Databases
  - Database Snapshots
  - DWConfiguration
  - DWDiagnostics
  - DWQueue
  - Estacio
    - Database Diagrams
    - Tables
      - System Tables
      - FileTables
      - External Tables
      - Graph Tables
      - dbo.Movimento
      - dbo.Pessoa
      - dbo.PessoaFisica
      - dbo.PessoaJuridica
      - dbo.Produto
    - Views

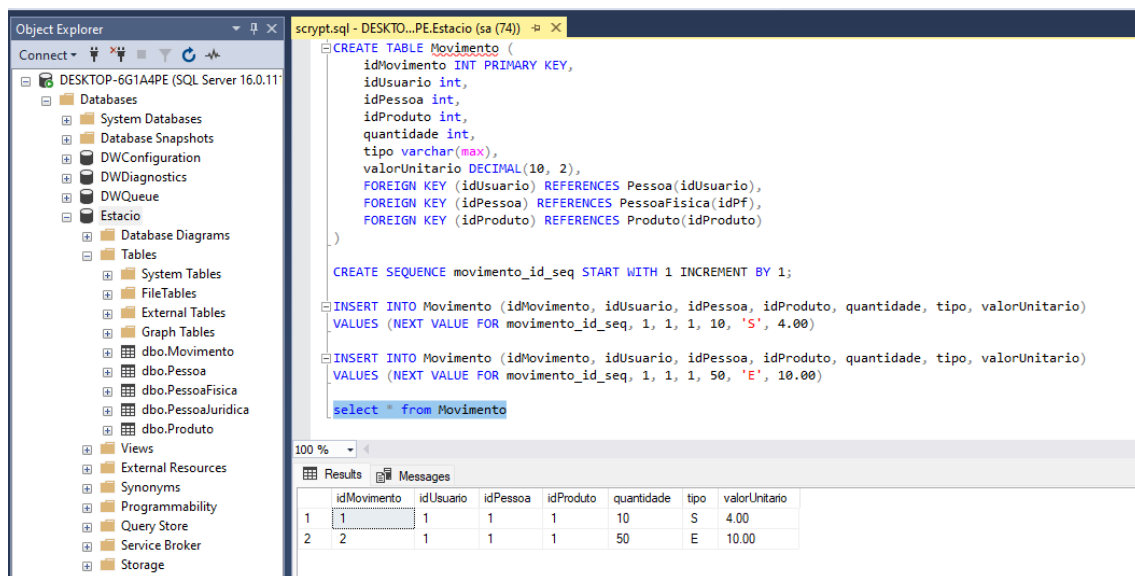
script.sql - DESKTO...PE.Estacio (sa (74))

```
CREATE TABLE PessoaJuridica (  
    idPj INT PRIMARY KEY,  
    nome varchar(max),  
    logradouro varchar(max),  
    cidade varchar(max),  
    estado varchar(2),  
    telefone varchar(11),  
    email varchar(max),  
    cnpj varchar(max),  
    FOREIGN KEY (idPj) REFERENCES Pessoa(idUsuario)  
)  
  
INSERT INTO PessoaJuridica (idPj, nome, logradouro, cidade, estado, telefone, email, cnpj)  
VALUES (2, 'JJC', 'Rua 12, casa 3, Quitanda', 'Riacho do Sul', 'PA', '1111-1111', 'joao@riacho.com', '1111111111111111')  
  
select * from PessoaJuridica
```

100 %

Results Messages

	idPj	nome	logradouro	cidade	estado	telefone	email	cnpj
1	2	JJC	Rua 12, casa 3, Quitanda	Riacho do Sul	PA	1111-1111	joao@riacho.com	1111111111111111



As *sequences* e *identities* são mecanismos utilizados para gerar valores únicos automaticamente em bancos de dados. A principal diferença entre elas está na sua implementação e na forma como são acessadas. As *sequences* são objetos independentes do tipo *sequence*, enquanto as *identities* são propriedades associadas às colunas de uma tabela. Isso significa que as *sequences* podem ser acessadas e utilizadas em várias tabelas, enquanto as *identities* estão ligadas a uma tabela específica.

As chaves estrangeiras desempenham um papel crucial na garantia da consistência dos dados em um banco de dados relacional. Elas estabelecem as relações entre as tabelas, permitindo a integridade referencial e evitando inconsistências nos dados. Ao definir chaves estrangeiras, estamos estabelecendo regras que garantem que os valores em uma tabela relacionada sempre existam e sejam válidos.

Alguns operadores do SQL, como SELECT, PROJECT, JOIN, UNION, entre outros, pertencem à álgebra relacional. Eles são fundamentais para a manipulação e recuperação de dados em bancos de dados relacionais. Por outro lado, o SQL também possui operadores que são definidos no cálculo relacional, como EXISTS e FOR ALL, que são utilizados em consultas mais complexas e específicas.

O agrupamento em consultas SQL é realizado através da cláusula GROUP BY. Essa cláusula permite agrupar linhas que têm valores iguais em uma ou mais colunas especificadas. É importante ressaltar

que ao utilizar o agrupamento, é obrigatório incluir uma função de agregação, como SUM, AVG, MAX, MIN, COUNT, entre outras, para especificar como os valores das colunas agrupadas serão tratados.

Dessa forma, concluímos que esta prática proporcionou uma exploração mais aprofundada de conceitos avançados em SQL e bancos de dados relacionais, permitindo aos alunos uma compreensão mais ampla e sólida desses tópicos.