	**UNIVERSIDADE ESTACIO DE SÁ** POLO INDAIATUBA – INDAIATUBA/SP DESENVOLVIMENTO FULL STACK - 22.3 Relatório da Missão Prática | Nível 2 | Mundo 3
Aluno:	Saulo Henrique dos Santos
Professor:	Simone Ingrid Monteiro Gama
Repositório:	https://github.com/SauloHenriqueSantos/Mundo03Nivel02

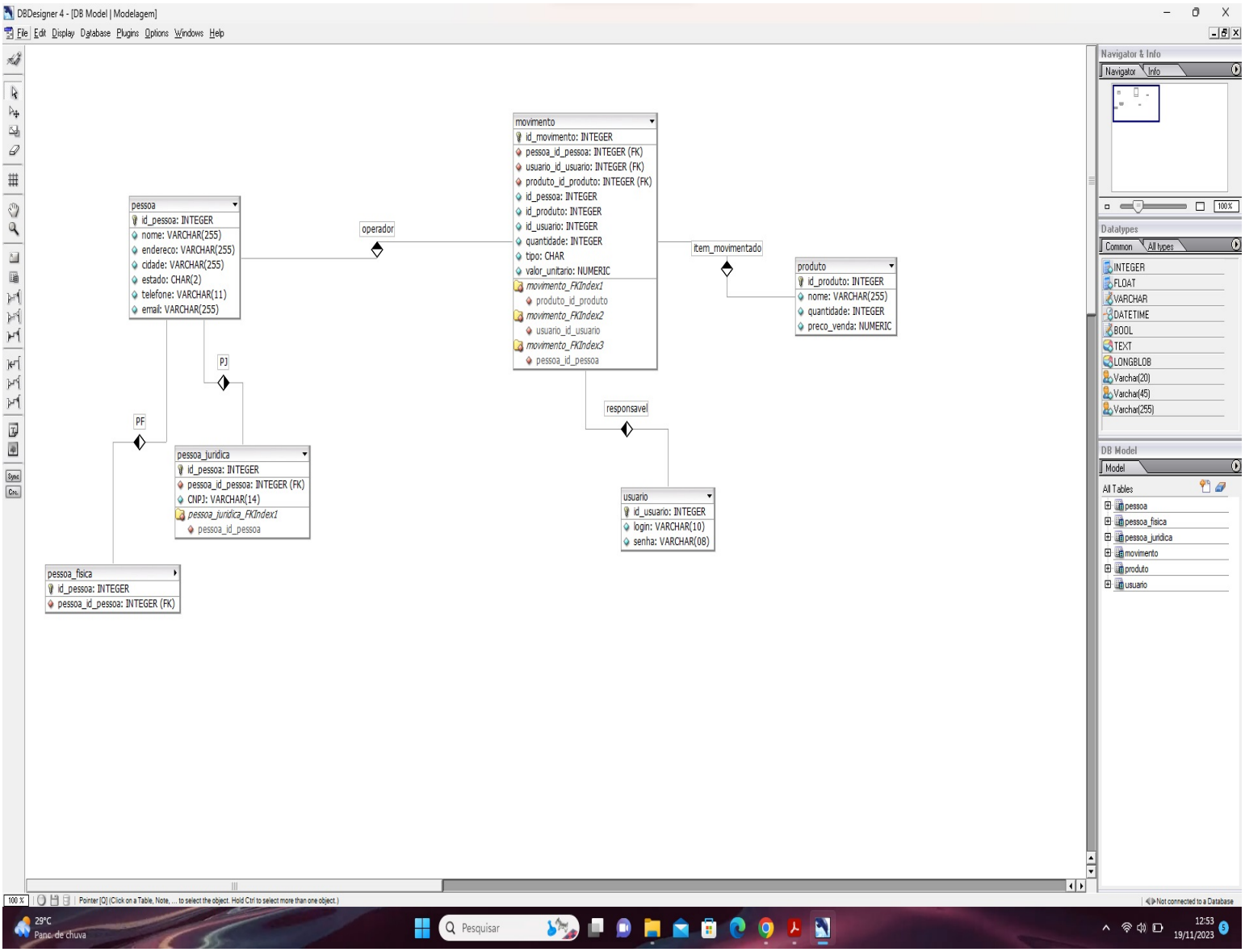
Título da Prática: 1º Procedimento | Criando o Banco de Dados

Objetivos da Prática:

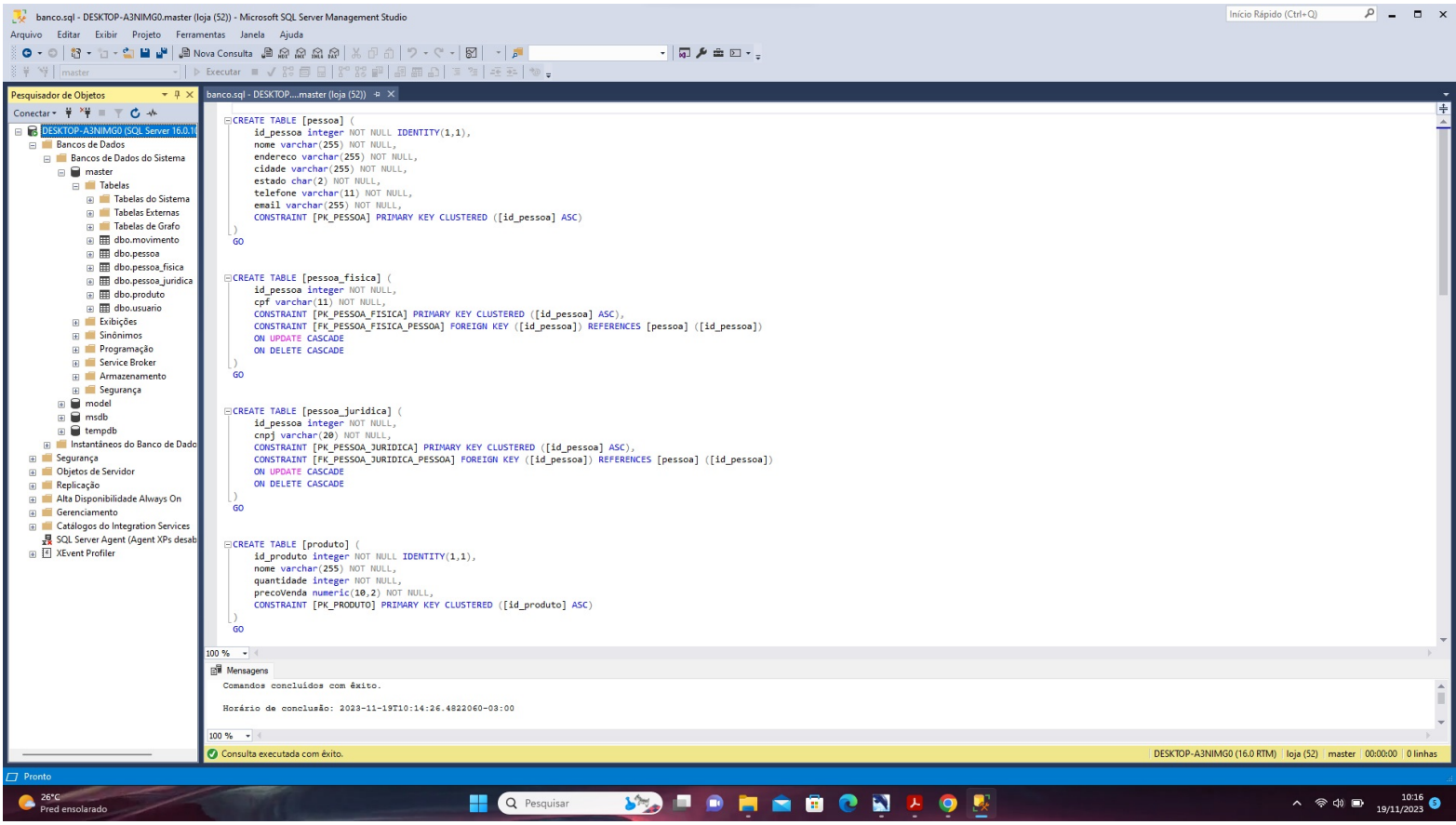
- Identificar os requisitos de um sistema e transformá-los no modelo adequado. Utilizar ferramentas de modelagem para bases de dados relacionais.
- Explorar a sintaxe SQL na criação das estruturas do banco (DDL).
- Explorar a sintaxe SQL na consulta e manipulação de dados (DML)
- No final do exercício, o aluno terá vivenciado a experiência de modelar a base de dados para um sistema simples, além de implementá-la, através da sintaxe SQL, na plataforma do SQL Server.

Códigos:

Tabela modelada no DBDesigner 4



Importação do DBDesigner4, com a ferramenta SQL Create Scripts:



Análise e Conclusão:

1. **Como são implementadas as diferentes cardinalidades, basicamente 1X1, 1XN ou NxN, em um banco de dados relacional?**

   1x1 (Um para Um): Você usa duas tabelas, onde uma tem uma chave primária que também é usada como chave estrangeira na outra tabela. Geralmente você faz isso quando quer separar informações que não são sempre necessárias. Por exemplo, informações pessoais em uma tabela e informações sensíveis como dados de login em outra.

   1xN (Um para Muitos): Neste caso, você tem uma chave primária em uma tabela e uma chave estrangeira na outra. É o tipo mais comum de relação. Por exemplo, um cliente (`ClienteID` como chave primária) pode fazer vários pedidos (cada pedido com um `ClienteID` como chave estrangeira).

   NxN (Muitos para Muitos): Aqui você precisa de uma tabela extra, chamada tabela de junção, que tem duas chaves estrangeiras que juntas formam uma chave primária composta. Isso é usado quando itens de uma tabela podem se relacionar com vários itens de outra tabela. Um bom exemplo é estudantes e cursos: um estudante pode se inscrever em vários cursos e um curso pode ter vários estudantes.

2. **Que tipo de relacionamento deve ser utilizado para representar o uso de herança em bancos de dados relacionais?**

   Para representar herança em bancos de dados relacionais, o tipo de relacionamento mais comum é o "relacionamento de subtipo/supertipo". Nesse modelo, uma tabela base, que representa o supertipo, contém as colunas comuns a todas as entidades, enquanto tabelas adicionais representam os subtipos e contêm colunas específicas para as características únicas de cada subtipo. Cada tabela de subtipo tem uma chave estrangeira que se refere à chave primária da tabela supertipo, estabelecendo uma relação de herança.

3. **Como o SQL Server Management Studio permite a melhoria da produtividade nas tarefas relacionadas ao gerenciamento do banco de dados?**

   O SQL Server Management Studio (SSMS) aumenta a produtividade ao oferecer uma interface gráfica intuitiva para gerenciamento de banco de dados, recursos de autocompletar e coloração de sintaxe para facilitar a escrita de código, ferramentas integradas para monitoramento e otimização de desempenho, e assistentes para simplificar tarefas complexas. Além disso, facilita o debugging, a gestão de segurança e a manutenção do banco de dados, tudo em uma única plataforma.

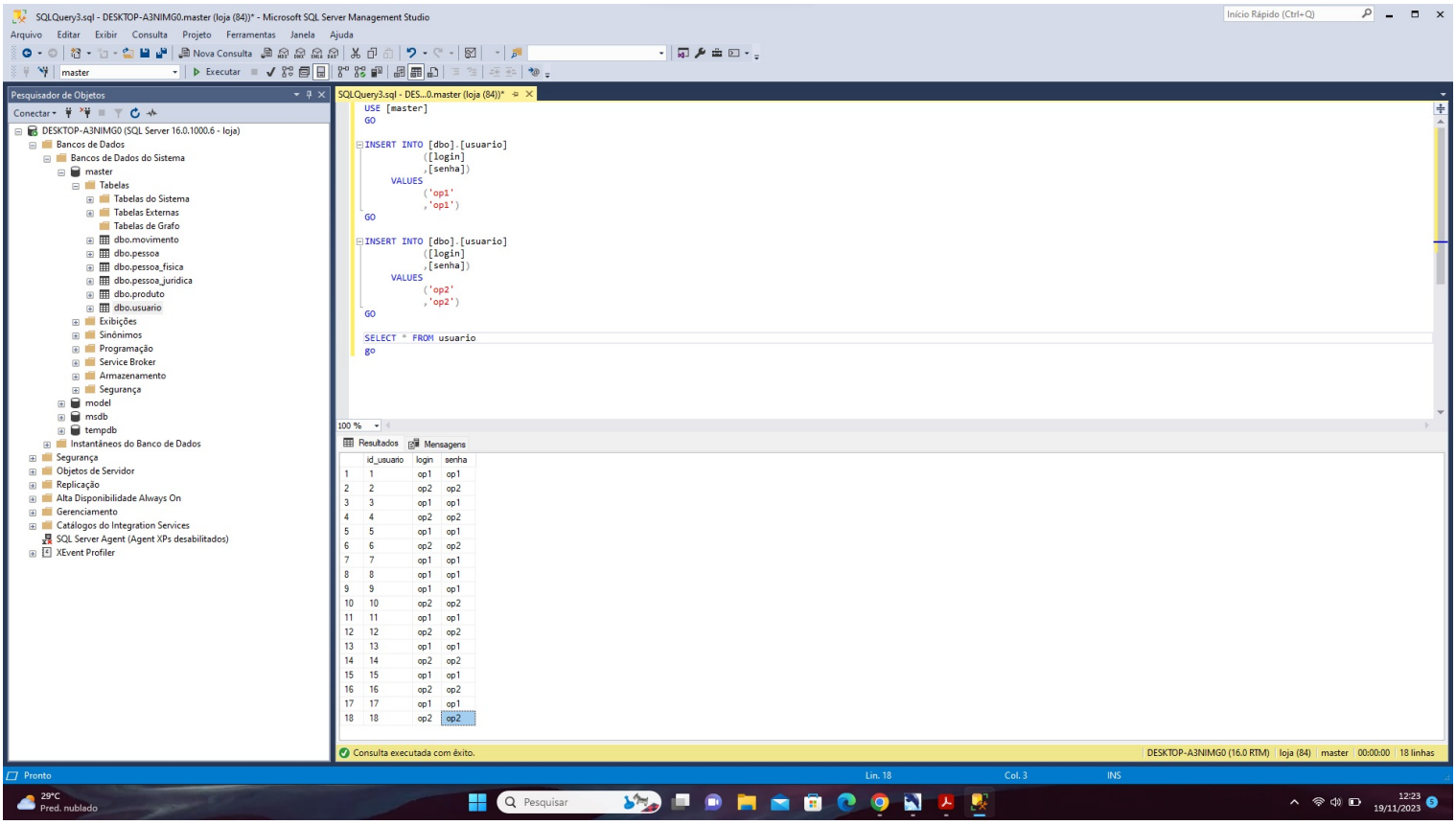
   No meu caso, tive que usar o DBeaver, que é uma ferramenta universal que faz o a mesma coisa que o SSMS faz, porem com algumas melhorias.

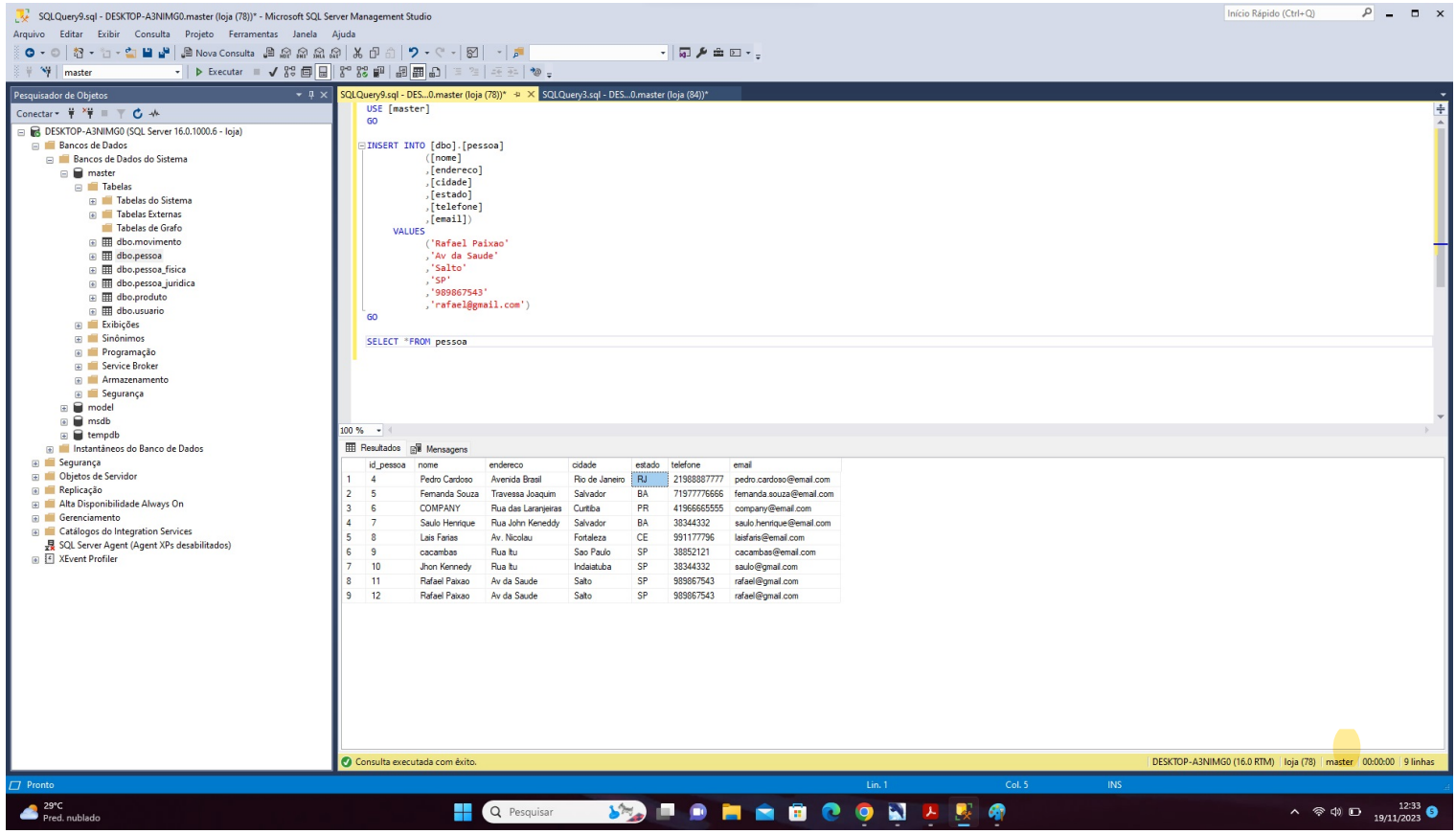
Título da Prática: 2º Procedimento | Alimentando a Base

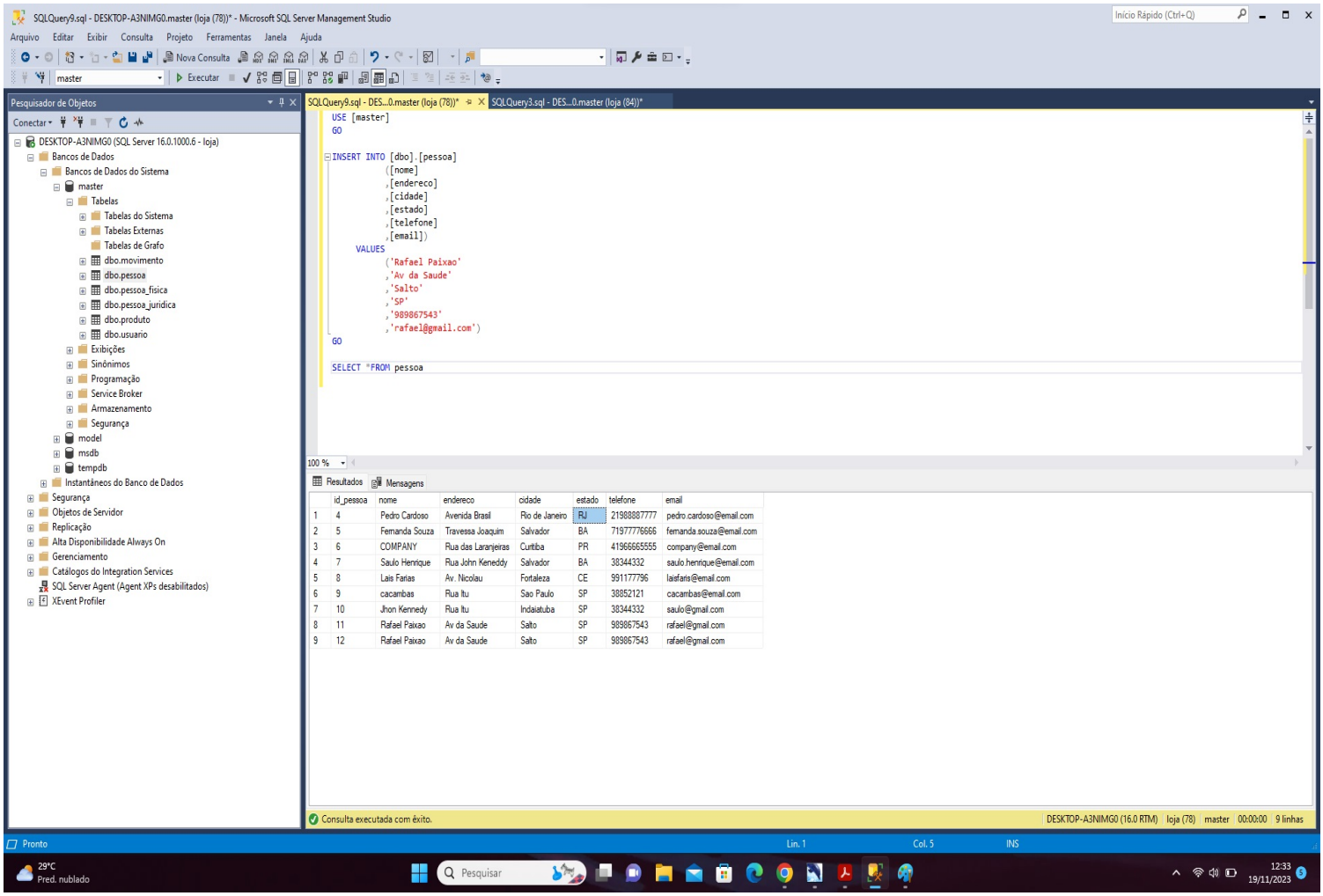
Login: loja

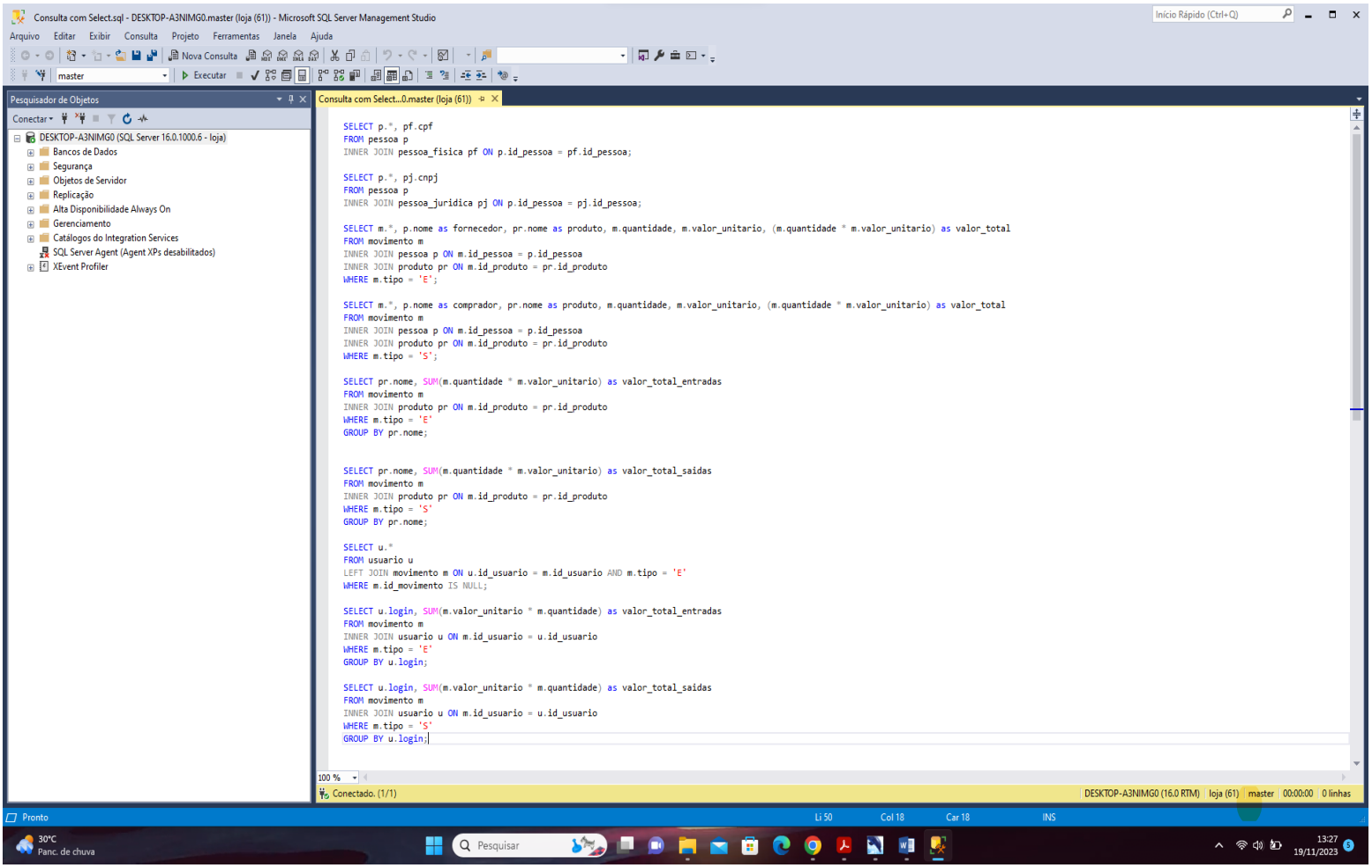
Senha: loja

Esquema de inserção de dados nas tabelas:






SQLQuery9.sql - DESKTOP-A3NIMG0.master (loja (78))* - Microsoft SQL Server Management Studio
Arquivo Editar Exibir Consulta Projeto Ferramentas Janela Ajuda
Nova Consulta
Executar
master
Pesquisador de Objetos
Conectar
DESKTOP-A3NIMG0 (SQL Server 16.0.1000.6 - loja)
Bancos de Dados
Bancos de Dados do Sistema
master
Tabelas
Tabelas do Sistema
Tabelas Externas
Tabelas de Grafo
dbo.movimento
dbo.pessoa
dbo.pessoa_fisica
dbo.pessoa_juridica
dbo.produto
dbo.usuario
Exibições
Sinônimos
Programação
Service Broker
Armazenamento
Segurança
model
msdb
tempdb
Instantâneos do Banco de Dados
Segurança
Objetos de Servidor
Replicação
Alta Disponibilidade Always On
Gerenciamento
Catálogos do Integration Services
SQL Server Agent (Agent XPs desabilitados)
XEvent Profiler
SQLQuery9.sql - DES...0.master (loja (78))*
SQLQuery3.sql - DES...0.master (loja (84))*
USE [master]
GO
INSERT INTO [dbo].[pessoa]
([nome]
,[endereco]
,[cidade]
,[estado]
,[telefone]
,[email])
VALUES
('Rafael Paixao'
,'Av da Saude'
,'Salto'
,'SP'
,'989867543'
,'rafael@gmail.com')
GO
SELECT *FROM pessoa
100 %
Resultados
Mensagens
id_pessoa nome endereco cidade estado telefone email
1 4 Pedro Cardoso Avenida Brasil Rio de Janeiro RJ 21988887777 pedro.cardoso@email.com
2 5 Fernanda Souza Travessa Joaquim Salvador BA 71977776666 fernanda.souza@email.com
3 6 COMPANY Rua das Laranjeiras Curitiba PR 41966665555 company@email.com
4 7 Saulo Henrique Rua John Keneddy Salvador BA 38344332 saulo.henrique@email.com
5 8 Lais Farias Av. Nicolau Fortaleza CE 991177796 laisfaris@email.com
6 9 cacambas Rua Itu Sao Paulo SP 38852121 cacambas@email.com
7 10 Jhon Kennedy Rua Itu Indaiatuba SP 38344332 saulo@gmail.com
8 11 Rafael Paixao Av da Saude Salto SP 989867543 rafael@gmail.com
9 12 Rafael Paixao Av da Saude Salto SP 989867543 rafael@gmail.com
Consulta executada com êxito.
DESKTOP-A3NIMG0 (16.0 RTM) loja (78) master 00:00:00 9 linhas
Pronto
Lin. 1
Col. 5
INS
29°C
Pred. nublado
Pesquisar
12:33
19/11/2023


Consulta com Select.sql - DESKTOP-A3NIMG0.master (loja (61)) - Microsoft SQL Server Management Studio
Arquivo Editar Exibir Consulta Projeto Ferramentas Janela Ajuda
Nova Consulta
Executar
master
Pesquisador de Objetos
Conectar
DESKTOP-A3NIMG0 (SQL Server 16.0.1000.6 - loja)
Bancos de Dados
Segurança
Objetos de Servidor
Replicação
Alta Disponibilidade Always On
Gerenciamento
Catálogos do Integration Services
SQL Server Agent (Agent XPs desabilitados)
XEvent Profiler
Consulta com Select...0.master (loja (61))
SELECT p.*, pf.cpf
FROM pessoa p
INNER JOIN pessoa_fisica pf ON p.id_pessoa = pf.id_pessoa;
SELECT p.*, pj.cnpj
FROM pessoa p
INNER JOIN pessoa_juridica pj ON p.id_pessoa = pj.id_pessoa;
SELECT m.*, p.nome as fornecedor, pr.nome as produto, m.quantidade, m.valor_unitario, (m.quantidade * m.valor_unitario) as valor_total
FROM movimento m
INNER JOIN pessoa p ON m.id_pessoa = p.id_pessoa
INNER JOIN produto pr ON m.id_produto = pr.id_produto
WHERE m.tipo = 'E';
SELECT m.*, p.nome as comprador, pr.nome as produto, m.quantidade, m.valor_unitario, (m.quantidade * m.valor_unitario) as valor_total
FROM movimento m
INNER JOIN pessoa p ON m.id_pessoa = p.id_pessoa
INNER JOIN produto pr ON m.id_produto = pr.id_produto
WHERE m.tipo = 'S';
SELECT pr.nome, SUM(m.quantidade * m.valor_unitario) as valor_total_entradas
FROM movimento m
INNER JOIN produto pr ON m.id_produto = pr.id_produto
WHERE m.tipo = 'E'
GROUP BY pr.nome;
SELECT pr.nome, SUM(m.quantidade * m.valor_unitario) as valor_total_saidas
FROM movimento m
INNER JOIN produto pr ON m.id_produto = pr.id_produto
WHERE m.tipo = 'S'
GROUP BY pr.nome;
SELECT u.*
FROM usuario u
LEFT JOIN movimento m ON u.id_usuario = m.id_usuario AND m.tipo = 'E'
WHERE m.id_movimento IS NULL;
SELECT u.login, SUM(m.valor_unitario * m.quantidade) as valor_total_entradas
FROM movimento m
INNER JOIN usuario u ON m.id_usuario = u.id_usuario
WHERE m.tipo = 'E'
GROUP BY u.login;
SELECT u.login, SUM(m.valor_unitario * m.quantidade) as valor_total_saidas
FROM movimento m
INNER JOIN usuario u ON m.id_usuario = u.id_usuario
WHERE m.tipo = 'S'
GROUP BY u.login;
100 %
Conectado. (1/1)
DESKTOP-A3NIMG0 (16.0 RTM) loja (61) master 00:00:00 0 linhas
Pronto
Li 50
Col 18
Car 18
INS
30°C
Panc. de chuva
Pesquisar
13:27
19/11/2023

Análise e Conclusão

1. Quais as diferenças no uso de sequence e identity?

Sequence (Sequência): Refere-se a uma ordem específica de elementos ou uma série de valores que seguem uma regra particular. Em bancos de dados, uma sequência pode ser usada para gerar números únicos consecutivos, geralmente utilizados como chaves primárias. Na programação, uma sequência pode ser uma lista, um array, ou qualquer outro tipo de coleção ordenada de elementos.

Identity (Identidade): Em matemática, a identidade é um valor que, quando usado em operações, não altera os outros elementos. Por exemplo, em multiplicação, o número 1 é a identidade porque qualquer número multiplicado por 1 permanece inalterado. Em bancos de dados, um campo de identidade é geralmente um campo que tem um valor único para cada registro, como uma chave primária autoincrementável. Em programação, uma função identidade é aquela que sempre retorna o mesmo valor que foi usado como argumento.

2. Qual a importância das chaves estrangeiras para a consistência do banco?

As chaves estrangeiras são essenciais em bancos de dados relacionais, pois garantem a integridade referencial ao vincular tabelas de maneira que as alterações em uma tabela se reflitam em outras relacionadas, prevenindo inconsistências e dados órfãos, e facilitam a manutenção e a compreensão das relações entre os dados.

3. Quais operadores do SQL pertencem a álgebra relacional e quais são definidos no cálculo relacional?

A álgebra relacional e o cálculo relacional são duas linguagens de consulta fundamentais para o modelo relacional em banco de dados, mas elas operam de maneiras distintas. A álgebra relacional utiliza operadores como seleção, projeção, união, diferença, produto cartesiano e junção, que são procedimentos para obter novas relações a partir de relações existentes. Por outro lado, o cálculo relacional é baseado em fórmulas lógicas e utiliza quantificadores universais e existenciais, assim como predicados que descrevem as propriedades que as tuplas devem satisfazer, em vez de operadores explícitos para manipulação de tuplas. Enquanto a álgebra relacional diz "como" obter o resultado, o cálculo relacional define "o que" se deseja obter como resultado, deixando o "como" para o sistema de gerenciamento do banco de dados resolver.

4. **Como é feito o agrupamento em consultas, e qual requisito é obrigatório?**

   O agrupamento em consultas, especialmente em bancos de dados SQL, é feito utilizando a cláusula `GROUP BY`, que agrupa linhas que têm os mesmos valores em colunas específicas em conjuntos resumidos. O requisito obrigatório para utilizar `GROUP BY` é que, para cada coluna que você selecionar na sua consulta que não está contida na cláusula `GROUP BY`, você deve aplicar uma função de agregação, como `COUNT()`, `SUM()`, `AVG()`, `MAX()`, ou `MIN()`. Essas funções de agregação são usadas para realizar cálculos sobre um conjunto de linhas agrupadas, fornecendo um único resultado por grupo. Por exemplo, se você quiser saber a quantidade de vendas por produto, você agruparia as linhas da tabela de vendas pelo identificador do produto e usaria a função `COUNT()` para contar o número de linhas em cada grupo.