**Missão Prática | Tirando proveito da nuvem para projetos de software 💻**

Nessa atividade revisaremos tudo o que utilizamos nas microatividades anteriores. Revisaremos todos os componentes e técnicas implementados no desenvolvimento do banco de dados SQL Azure para a LogiMove Transportes. Isto inclui a configuração do ambiente Azure, a criação e o gerenciamento das tabelas, e a inserção e consulta de dados.

**Contextualização**

A LogiMove Transportes, uma empresa renomada no setor de logística de transporte, enfrenta desafios significativos em sua operação diária. A coordenação entre agendadores, despachantes, motoristas e clientes é crucial, mas o processo atual, baseado em formulários de papel e comunicações telefônicas, tem se mostrado ineficiente.

**Problemas Identificados**:

* Excesso de papelada, muitas vezes incompleta ou sem assinaturas.
* Dificuldade na disponibilidade dos distribuidores, resultando em atrasos.
* Motoristas frequentemente parados, esperando por coordenação.
* Atraso nas remessas, afetando negativamente a satisfação do cliente e os negócios recorrentes.

**Solução Proposta**:

Para resolver esses desafios, a empresa decide migrar para um sistema digital, substituindo formulários de papel e chamadas telefônicas por documentos digitais e comunicação online. A implementação de autenticação digital permitirá uma coordenação e acompanhamento eficazes das remessas, acessíveis vias navegador web ou aplicativo móvel.

**Projeto de Banco de Dados**:

Como líder de desenvolvimento de software, você propõe o desenvolvimento de um protótipo que inclui a criação de um banco de dados no Azure SQL. Este banco de dados será projetado para armazenar informações cruciais, incluindo:

* Dados dos motoristas: informações pessoais, qualificações, histórico de viagens.
* Informações dos clientes: detalhes de contato, histórico de pedidos, preferências.
* Detalhes dos pedidos: informações do pedido, status, cronograma de entrega.
* O protótipo servirá como base para o aplicativo de produção futuro. Portanto, as escolhas tecnológicas feitas agora devem ser escaláveis e compatíveis com as soluções finais.

**Objetivos do Projeto**:

* Desenvolver um banco de dados robusto e seguro no Azure SQL.
* Garantir que o banco de dados possa escalar conforme a empresa cresce.
* Facilitar a integração com outras plataformas e serviços.

O projeto visa transformar radicalmente a maneira como a LogiMove Transportes

opera, aumentando a eficiência, reduzindo atrasos e melhorando a satisfação do

cliente. A adoção de uma solução baseada em Azure SQL é um passo significativo em

direção à digitalização e modernização das operações da empresa.

**Roteiro de prática 📝**

**- Material necessário para a prática**

* Conta na Azure.
* Navegador Web: Google Chrome, Firefox, MS Edge, Safari ou Opera.

**- Procedimentos**

Esta atividade tem por objetivo desenvolver um banco de dados no Azure SQL para a LogiMove Transportes, uma empresa de logística. O objetivo é migrar de um sistema baseado em papel para uma solução digital, utilizando autenticação digital para melhor coordenação e rastreamento de remessas. O banco de dados armazenará informações sobre motoristas, clientes e pedidos.

1. Configuração do Ambiente Azure:

* Criar uma conta no Azure.
* Configurar uma instância do Azure SQL Database.
* Estabelecer os parâmetros de segurança, como firewalls e regras de acesso.

2. Design do Banco de Dados:

* Definir a arquitetura do banco de dados considerando as necessidades da empresa.
* Criar um diagrama de entidade-relacionamento (ER) para visualizar as relações entre as tabelas.

3. Implementação do Banco de Dados:

* Utilizar T-SQL para criar tabelas, definir chaves primárias, chaves estrangeiras e índices.

**Sugestão de Estrutura das Tabelas**

**1. Tabela de Motoristas (Drivers):**

* DriverID (Chave Primária)
* Nome
* CNH
* Endereço
* Contato

**2. Tabela de Clientes (Clients):**

* ClientID (Chave Primária)
* Nome
* Empresa
* Endereço
* Contato

**3. Tabela de Pedidos (Orders):**

* OrderID (Chave Primária)
* ClientID (Chave Estrangeira)
* DriverID (Chave Estrangeira)
* Detalhes do Pedido
* Data de Entrega
* Status

**Comandos T-SQL**

1. **Criação de Tabelas:**

CREATE TABLE Drivers (

DriverID INT PRIMARY KEY,

Nome VARCHAR (100),

CNH VARCHAR (20),

Endereço VARCHAR (200),

Contato VARCHAR (50)

);

CREATE TABLE Clients (

ClientID INT PRIMARY KEY,

Nome VARCHAR (100),

Empresa VARCHAR (100),

Endereço VARCHAR (200),

Contato VARCHAR (50)

);

CREATE TABLE Orders (

OrderID INT PRIMARY KEY,

ClientID INT,

DriverID INT,

DetalhesPedido TEXT,

DataEntrega DATE,

Status VARCHAR (50),

FOREIGN KEY (ClientID) REFERENCES Clients (ClientID),

FOREIGN KEY (DriverID) REFERENCES Drivers (DriverID)

);

**- Resultados esperados ✨**

Ao concluir a atividade espera-se que o aluno provisione um banco de dados para a LogiMove Transportes e que esteja funcional e otimizado para operações diárias e preparado para escalabilidade futura. Para isso as seguintes etapas deverão ser concluídas e apresentadas:

**1. Configuração e Acesso ao Banco de Dados:**

* Banco de dados configurado corretamente no Azure SQL.
* Acesso ao banco de dados estabelecido sem problemas, garantindo conectividade e segurança.

**2. Criação e Estruturação das Tabelas:**

* Tabelas criadas no banco de dados de acordo com a estrutura sugerida, incluindo tabelas para Motoristas, Clientes e Pedidos.

**3. Inserção e Gestão de Dados:**

* Dados de teste inseridos nas tabelas, cobrindo diferentes cenários e casos de uso.

**4. Execução e Validação de Consultas:**

* Consultas T-SQL executadas com sucesso, com capacidade de recuperar, filtrar e ordenar dados conforme necessário.

**5. Operações CRUD Eficientes:**

* Demonstração de operações CRUD - Criar, Ler, Atualizar e Deletar dados.
* Testes para assegurar que as operações CRUD estão funcionando conforme esperado, com respostas rápidas e precisas.

**Segue abaixo passo a passo as implementações que foram solicitados na documentação acima:**

Criando o Banco de Dados

* **Edição:** GeneralPurpose (propósito geral).
* **Nível de serviço:** GP\_Gen5\_2 (2 vCores).
* **Tamanho máximo:** 32 GB.
* **Ordenação:** SQL\_Latin1\_General\_CP1\_CI\_AS (não diferencia maiúsculas/minúsculas).
* **Ledger:** OFF (rastreamento imutável desativado).

CREATE DATABASE [MoveTransporteDB] (EDITION = 'GeneralPurpose', SERVICE\_OBJECTIVE ='GP\_Gen5\_2', MAXSIZE = 32 GB) WITH CATALOG\_COLLATION =SQL\_Latin1\_General\_CP1\_CI\_AS, LEDGER = OFF;

GO

Criando as Tabelas:

**1. Criação da Tabela Drivers**

* Essa tabela armazena informações sobre motoristas.
* Colunas principais:
* DriverID (chave primária): identificador único do motorista.
* Nome: nome do motorista.
* CNH: número da Carteira Nacional de Habilitação (CNH).
* Endereço: endereço do motorista.
* Contato: informações de contato.

CREATE TABLE [Drivers]([DriverID] [int] NOT NULL, [Nome] [varchar](100) NULL,[CNH] [varchar](20) NULL, [Endereço] [varchar](200) NULL, [Contato] [varchar](50) NULL, PRIMARY KEY CLUSTERED ([DriverID] ASC) WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY] ) ON [PRIMARY]

GO

**2. Criação da Tabela DriverQualifications**

* Registra as qualificações dos motoristas.
* Colunas principais:
* QualificationID (chave primária): identificador único da qualificação.
* DriverID (chave estrangeira): referência ao motorista na tabela Drivers.
* Qualificação: descrição da qualificação.
* DataObtenção: data em que a qualificação foi obtida.
* Validade: validade da qualificação.
* **Adição de chave estrangeira (DriverID)**: garante a integridade referencial com a tabela Drivers.

CREATE TABLE [DriverQualifications]([QualificationID] [int] NOT NULL, [DriverID] [int] NULL, [Qualificação] [varchar](100) NULL, [DataObtenção] [date] NULL,[Validade] [date] NULL, PRIMARY KEY CLUSTERED ([QualificationID] ASC) WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]) ON [PRIMARY]

GO

ALTER TABLE [DriverQualifications] WITH CHECK ADD FOREIGN KEY([DriverID]) REFERENCES [Drivers] ([DriverID])

GO

**3. Criação da Tabela DriverTravelHistory**

* Registra o histórico de viagens dos motoristas.
* Colunas principais:
* TravelID (chave primária): identificador único da viagem.
* DriverID (chave estrangeira): referência ao motorista na tabela Drivers.
* DataViagem: data da viagem.
* Origem: ponto de partida da viagem.
* Destino: destino da viagem.
* DistanciaPercorrida: distância total percorrida na viagem.
* **Adição de chave estrangeira (DriverID)**: assegura que cada viagem esteja associada a um motorista válido.

CREATE TABLE [DriverTravelHistory]([TravelID] [int] NOT NULL, [DriverID] [int] NULL, [DataViagem] [date] NULL, [Origem] [varchar](200) NULL, [Destino] [varchar](200) NULL, [DistanciaPercorrida] [float] NULL, PRIMARY KEY CLUSTERED ([TravelID] ASC) WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]) ON [PRIMARY]

GO

ALTER TABLE [DriverTravelHistory] WITH CHECK ADD FOREIGN KEY([DriverID]) REFERENCES [Drivers] ([DriverID])

GO

**4. Criação da Tabela Clients**

* Registra informações sobre clientes.
* Colunas principais:
* ClientID (chave primária): identificador único do cliente.
* Nome: nome do cliente.
* Empresa: empresa associada ao cliente.
* Endereço: endereço do cliente.
* Contato: informações de contato.

CREATE TABLE [Clients]([ClientID] [int] NOT NULL,[Nome] [varchar](100) NULL,[Empresa] [varchar](100) NULL,[Endereço] [varchar](200) NULL, [Contato] [varchar](50) NULL, PRIMARY KEY CLUSTERED ([ClientID] ASC) WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]) ON [PRIMARY]

GO

**5. Criação da Tabela ClientPreferences**

* Registra preferências dos clientes.
* Colunas principais:
* PreferenceID (chave primária): identificador único da preferência.
* ClientID (chave estrangeira): referência ao cliente na tabela Clients.
* Preferência: descrição das preferências do cliente.
* **Adição de chave estrangeira (ClientID)**: assegura a relação com a tabela Clients.

CREATE TABLE [ClientPreferences]([PreferenceID] [int] NOT NULL, [ClientID] [int] NULL, [Preferencia] [text] NULL, PRIMARY KEY CLUSTERED ([PreferenceID] ASC) WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

ALTER TABLE [ClientPreferences] WITH CHECK ADD FOREIGN KEY([ClientID]) REFERENCES [Clients] ([ClientID])

GO

**6. Criação da Tabela Orders**

* Armazena informações sobre pedidos feitos pelos clientes.
* Colunas principais:
* OrderID (chave primária): identificador único do pedido.
* ClientID (chave estrangeira): referência ao cliente que fez o pedido.
* DriverID (chave estrangeira): referência ao motorista responsável pela entrega.
* DetalhesPedido: detalhes do pedido.
* DataEntrega: data prevista para entrega.
* Status: status atual do pedido.
* **Adição de chave estrangeira (ClientID)**: vincula o pedido a um cliente específico.
* **Adição de chave estrangeira (DriverID)**: vincula o pedido a um motorista específico.

CREATE TABLE [Orders]([OrderID] [int] NOT NULL, [ClientID] [int] NULL,[DriverID] [int] NULL, [DetalhesPedido] [text] NULL, [DataEntrega] [date] NULL, [Status] [varchar](50) NULL, PRIMARY KEY CLUSTERED ([OrderID] ASC) WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

ALTER TABLE [Orders] WITH CHECK ADD FOREIGN KEY([ClientID]) REFERENCES [Clients]([ClientID])

GO

ALTER TABLE [Orders] WITH CHECK ADD FOREIGN KEY([DriverID]) REFERENCES [Drivers] ([DriverID])

GO

**7. Criação da Tabela OrderStatusHistory**

* Registra o histórico de status dos pedidos.
* Colunas principais:
* StatusHistoryID (chave primária): identificador único do histórico.
* OrderID (chave estrangeira): referência ao pedido na tabela Orders.
* StatusAnterior: status anterior do pedido.
* StatusAtual: status atualizado do pedido.
* DataMudança: data da alteração de status.
* **Adição de chave estrangeira (OrderID)**: assegura a vinculação ao pedido correspondente.

CREATE TABLE [OrderStatusHistory]([StatusHistoryID] [int] NOT NULL, [OrderID] [int] NULL, [StatusAnterior] [varchar](50) NULL, [StatusAtual] [varchar](50) NULL, [DataMudança] [date] NULL, PRIMARY KEY CLUSTERED ([StatusHistoryID] ASC) WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]) ON [PRIMARY]

GO

ALTER TABLE [OrderStatusHistory] WITH CHECK ADD FOREIGN KEY([OrderID]) REFERENCES [Orders] ([OrderID])

GO

**8. Criação da Tabela ClientOrderHistory**

* Registra o histórico de pedidos feitos pelos clientes.
* Colunas principais:
* HistoryID (chave primária): identificador único do histórico.
* OrderID (chave estrangeira): referência ao pedido na tabela Orders.
* ClientID (chave estrangeira): referência ao cliente na tabela Clients.
* DataPedido: data em que o pedido foi realizado.
* ResumoPedido: resumo dos detalhes do pedido.
* **Adição de chave estrangeira (ClientID)**: garante a relação com a tabela Clients.
* **Adição de chave estrangeira (OrderID)**: assegura a vinculação ao pedido correspondente.

CREATE TABLE [ClientOrderHistory] ([HistoryID] [int] NOT NULL, [OrderID] [int] NULL, [ClientID] [int] NULL, [DataPedido] [date] NULL, [ResumoPedido] [text] NULL, PRIMARY KEY CLUSTERED ([HistoryID] ASC) WITH (STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

ALTER TABLE [ClientOrderHistory] WITH CHECK ADD FOREIGN KEY([ClientID]) REFERENCES [Clients] ([ClientID])

GO

ALTER TABLE [ClientOrderHistory] WITH CHECK ADD FOREIGN KEY([OrderID]) REFERENCES [Orders] ([OrderID])

GO

**Conclusão**

O conjunto de scripts cria um banco de dados relacional para gerenciar motoristas, clientes, pedidos e históricos de entregas. As tabelas possuem relações bem definidas através de **chaves estrangeiras**, garantindo a integridade dos dados. Essa estrutura facilita a gestão de entregas e monitoramento de motoristas e clientes. 🚚📦

**1. Inserção de Qualificações dos Motoristas**

INSERT INTO DriverQualifications (QualificationID, DriverID, Qualificação, DataObtenção, Validade)

VALUES

(1, 1, 'Transporte de Combustiveis', '2024-12-14', '2025-01-15'),

(2, 2, 'Transporte de Granito tipo 2', '2024-12-20', '2025-01-20');

* Adiciona qualificações para dois motoristas.
* Cada registro contém a qualificação específica, a data em que foi obtida e a validade.

**2. Inserção de Histórico de Viagens dos Motoristas**

INSERT INTO DriverTravelHistory (TravelID, DriverID, DataViagem, Origem, Destino, DistanciaPercorrida)

VALUES

(1, 1, '2025-01-15', 'Porto Alegre', 'Espirito Santo', 2.261),

(2, 2, '2025-01-20', 'São Paulo', 'João Pessoa', 2.764);

* Registra viagens realizadas por motoristas, incluindo a data, origem, destino e a distância percorrida.

**3. Inserção de Novos Clientes**

INSERT INTO Clients (ClientID, Nome, Empresa, Endereço, Contato)

VALUES

(1,'Empresa Vespor', 'VesporAuto Ltda', 'Rua Jairo Pena, 88', '11333885544'),

(2, 'Empresa SOSGranito', 'S O.S.','Avenida Darly santos, 40', '11222554644');

* Adiciona informações de dois clientes, incluindo nome, empresa, endereço e contato.

**4. Inserção das Preferências dos Clientes**

INSERT INTO ClientPreferences (PreferenceID, ClientID, Preferencia)

VALUES

(1,1, 'Preferência por transportes rápidos e seguros'),

(2, 2, 'Preferência por custo baixo');

* Registra preferências de clientes sobre o transporte.

**5. Inserção de Pedidos**

INSERT INTO Orders (OrderID, ClientID, DriverID, DetalhesPedido, DataEntrega, Status)

VALUES

(1, 1, 1, '2000 amortecedores ','2025-01-12', 'Entregue'),

(2, 2, 2, '100 unidades de Explosivos tipo C','2025-01-06', 'Em trânsito');

* Insere pedidos realizados pelos clientes, associando cada pedido a um motorista responsável.

**6. Inserção do Histórico de Status dos Pedidos**

INSERT INTO OrderStatusHistory (StatusHistoryID, OrderID, StatusAnterior, StatusAtual, DataMudança)

VALUES

(1, 1, 'Em preparação', 'Entregue', '2025-12-10'),

(2, 2, 'Aguardando coleta', 'Em trânsito', '2024-12-15');

* Registra mudanças de status dos pedidos ao longo do tempo.

**7. Inserção do Histórico de Pedidos dos Clientes**

INSERT INTO ClientOrderHistory (HistoryID, OrderID, ClientID, DataPedido, ResumoPedido)

VALUES

(1, 1, 1, '2025-01-11', 'Pedido de 70 caixas de Sapatos'),

(2, 2, 2, '2025-01-12', 'Pedido de 200 unidades de vassoura');

* Adiciona o histórico de pedidos feitos pelos clientes.

**8. Consulta das Qualificações dos Motoristas**

SELECT d. Nome, dq. Qualificação, dq. DataObtenção, dq. Validade

FROM Drivers d

JOIN DriverQualifications dq ON d.DriverID = dq.DriverID;

* Retorna os nomes dos motoristas e suas respectivas qualificações.

**9. Consulta de Pedidos Disponíveis**

SELECT o. OrderID, c. Nome as Cliente, d. Nome as Motorista, o. DetalhesPedido, o. Status

FROM Orders o

JOIN Clients c ON o. ClientID = c. ClientID

JOIN Drivers d ON o. DriverID = d. DriverID;

* Lista os pedidos, associando-os aos clientes e motoristas responsáveis.

**10. Consulta do Histórico de Viagens por Data**

SELECT dt. DataViagem, dt. Origem, dt. Destino, dt. DistanciaPercorrida

FROM DriverTravelHistory dt

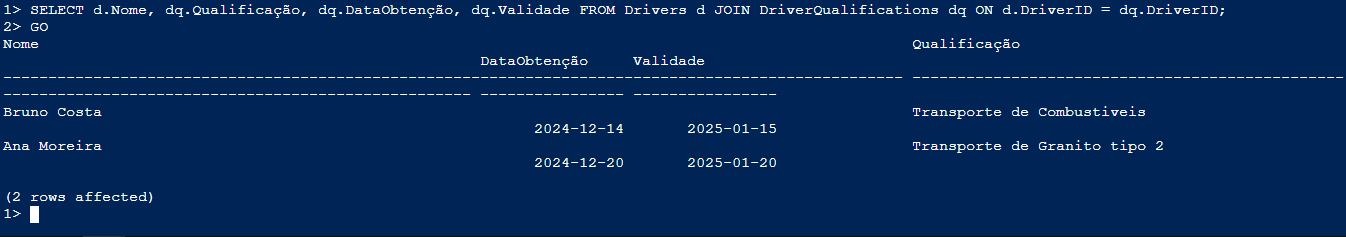
WHERE dt. DataViagem BETWEEN '2025-01-15' AND '2025-01-30';

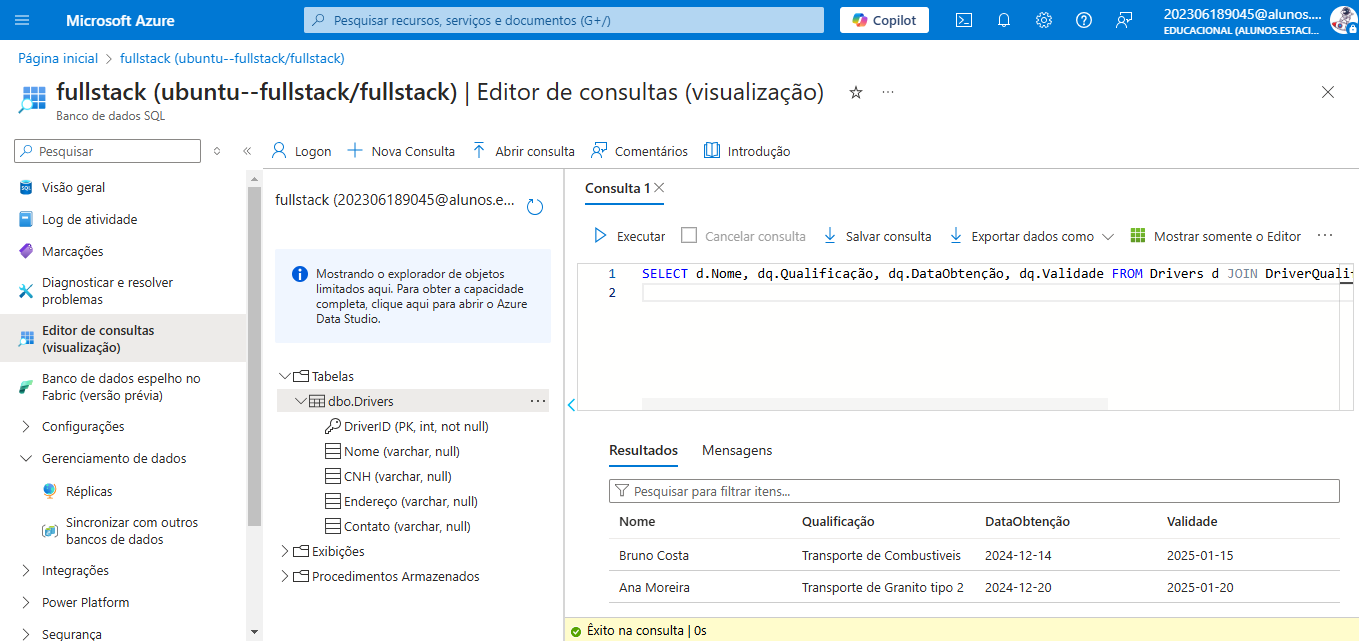
* Retorna viagens realizadas dentro do período especificado.

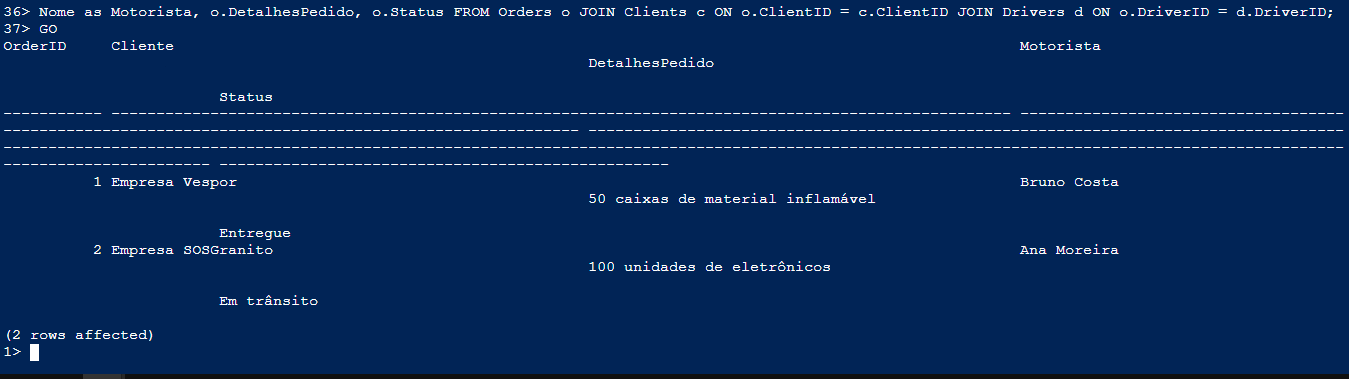
**Conclusão**

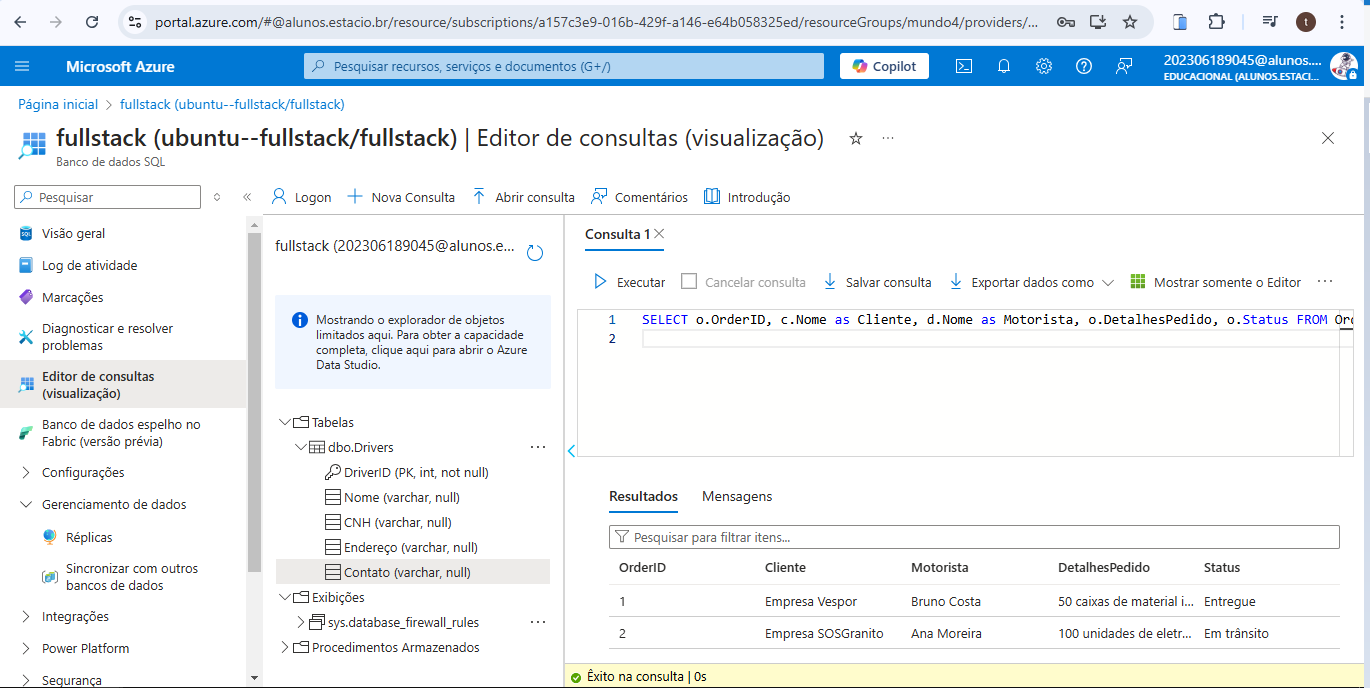
Esses scripts inserem dados nas tabelas previamente criadas e realizam consultas importantes para o gerenciamento de motoristas, clientes e pedidos. 🚛📦

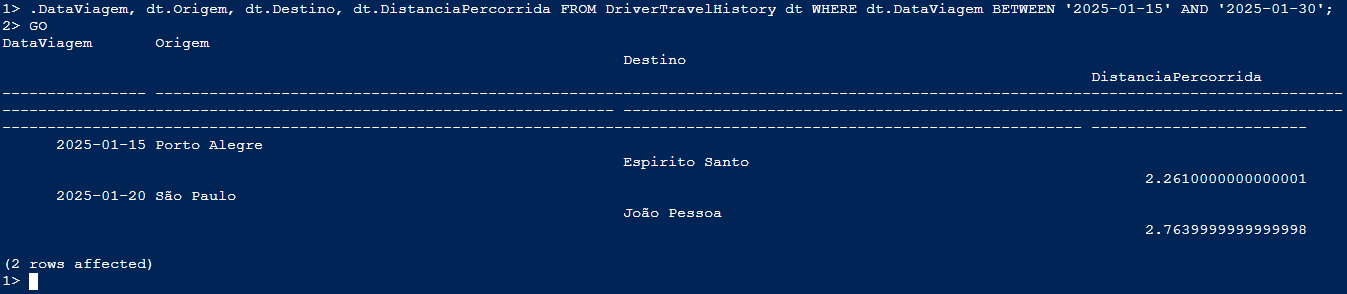
Segue Abaixo alguns resultados Referente as Implementações que foram feitas no Banco de Dados da MoveTransportes:

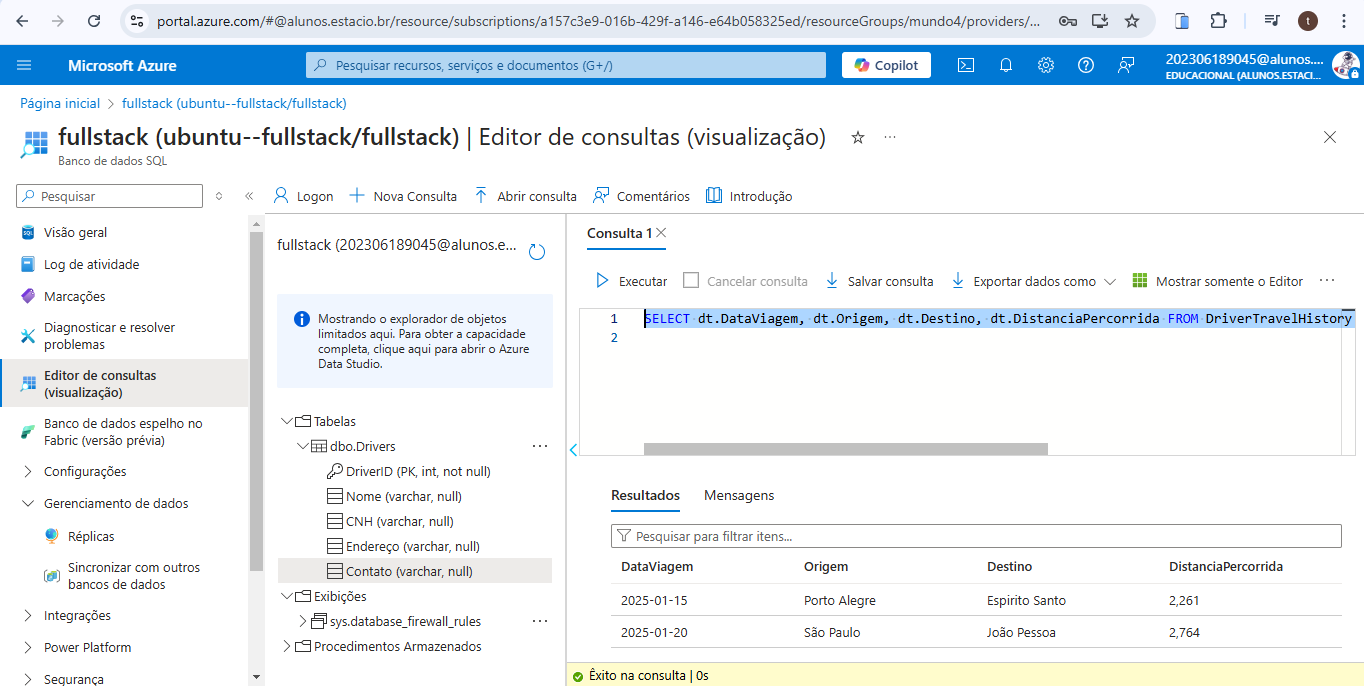












**Conclusão**

A **LogiMove Transportes**, uma empresa consolidada no setor logístico, identificou desafios operacionais significativos que impactam diretamente a eficiência e a satisfação do cliente. O excesso de papelada, a dificuldade de comunicação entre os envolvidos, a falta de coordenação na disponibilidade de motoristas e os constantes atrasos nas remessas são problemas críticos que dificultam o crescimento sustentável da empresa.

Para solucionar essas dificuldades, a LogiMove decidiu adotar **meios tecnológicos** para otimizar seus processos, eliminando burocracias manuais e garantindo maior controle e rastreabilidade sobre suas operações. A implementação de um **banco de dados relacional** foi uma das primeiras iniciativas para estruturar e armazenar informações essenciais de maneira organizada e acessível.

Os **scripts SQL** desenvolvidos criam e estruturam esse banco de dados, permitindo a digitalização e automação dos principais processos logísticos da empresa. A seguir, destacam-se os principais pontos abordados:

1. **Criação do banco de dados MoveTransporteDB**
   * Banco de dados configurado para operar em um ambiente seguro e escalável, garantindo alta disponibilidade.
2. **Estruturação das tabelas**
   * Tabelas foram criadas para armazenar **informações de motoristas, clientes, pedidos, históricos de viagens e status dos pedidos**.
   * Relações foram estabelecidas por meio de **chaves primárias e estrangeiras**, assegurando integridade referencial.
3. **Inserção de dados operacionais**
   * Foram adicionadas **qualificações dos motoristas**, garantindo a alocação correta conforme suas habilidades.
   * O **histórico de viagens** foi registrado para melhorar a rastreabilidade das operações.
   * **Clientes e suas preferências** foram armazenados para oferecer um atendimento mais personalizado.
   * **Pedidos e seu histórico de status** foram registrados, permitindo maior controle sobre cada remessa.
4. **Consultas estratégicas para tomada de decisão**
   * Consultas SQL foram criadas para visualizar as **credenciais dos motoristas**, os **pedidos disponíveis** e o **histórico de viagens**.
   * Essas consultas proporcionam uma visão clara das operações, ajudando na alocação eficiente de motoristas e no monitoramento do andamento dos pedidos.

**Impacto e Benefícios**

A implementação desse sistema baseado em banco de dados traz diversos **benefícios estratégicos** para a LogiMove Transportes:  
✅ **Redução de burocracia** – Eliminação de formulários em papel, minimizando erros e aumentando a eficiência administrativa.  
✅ **Maior controle operacional** – Registros detalhados de viagens, pedidos e qualificações dos motoristas melhoram a **tomada de decisão**.  
✅ **Agilidade na comunicação** – Integração dos dados entre motoristas, clientes e despachantes reduz o tempo de espera e melhora o fluxo logístico.  
✅ **Satisfação do cliente** – Com menos atrasos e maior previsibilidade nas entregas, a experiência do cliente melhora significativamente.

Com esse novo sistema digitalizado, a **LogiMove Transportes** moderniza sua gestão logística, melhora seu desempenho operacional e fortalece sua posição no mercado. 🚛📦