FACULDADE ESTÁCIO DE SÁ

CURSO: DESENVOLVIMENTO FULL STACK

3º SEMESTRE – MATRÍCULA 202302595341

ALAIM ALMEIDA DE OLIVEIRA

Vamos manter as informações?

Modelagem e implementação de um banco de dados simples, utilizando como base o SQL Server

Salvador – BA

2024

1. Introdução

Um banco de dados é um sistema organizado para armazenar, gerenciar e recuperar dados de forma eficiente. Ele é projetado para permitir a criação, manipulação e consulta de informações de maneira estruturada, proporcionando acesso rápido e seguro aos dados armazenados.

Os bancos de dados desempenham um papel fundamental em praticamente todas as áreas da tecnologia da informação e são utilizados em uma ampla variedade de aplicativos, desde sistemas de gerenciamento de conteúdo até aplicativos de comércio eletrônico, sistemas de gerenciamento de relacionamento com o cliente (CRM), sistemas de gestão empresarial (ERP), entre outros.

2. Modelagem de Dados

A modelagem de dados é o processo de representar os dados e as relações entre eles de forma abstrata e estruturada. Seu objetivo é criar uma representação visual ou conceitual dos dados que ajude a entender e projetar o sistema de banco de dados de forma eficaz.

3. Objetivo da Prática

Com isso, na disciplina do Nível 2: Vamos Manter as Informações? no mundo 3, foi realizado a prática de modelagem de dados e implementação com intuito de praticar tudo o que foi adquirido de forma teórica da disciplina, com projeto fictício de uma loja com clientes, fornecedores, compradores e toda a movimentação de compra e venda sendo estruturada em tabelas, inserindo todos os dados e com elaboração de consultas SQL.

A prática da implementação e modelagem de dados, eleva a compreensão profunda e abordagem importantes adquiridas nos conteúdos das aulas.

1. Entender os requisitos do sistema: Permite aos desenvolvedores e analistas compreender as necessidades dos usuários e os requisitos de dados do sistema antes de iniciar o desenvolvimento do banco de dados.
2. Projetar a estrutura do banco de dados: Ajuda a definir a estrutura dos dados, incluindo as tabelas, campos, chaves primárias, chaves estrangeiras e relacionamentos entre as entidades como pessoas físicas e jurídicas, usuários, produtos, e movimentações de entrada e saída.

4. Estrutura

A metodologia adotada neste processo visava facilitar a criação e implementação eficiente do banco de dados, utilizando a ferramenta DB Designer Online para modelagem visual.

Primeiramente, foi realizada a definição da estrutura do banco de dados por meio da modelagem visual, utilizando o DB Designer Online para identificar entidades, atributos e relacionamentos. Essa abordagem interativa e intuitiva permitiu validar o design conceitual.

Em seguida, os códigos SQL correspondentes ao esquema foram gerados automaticamente pela ferramenta e utilizados como base para a implementação no SQL Server Management Studio. O ambiente do SQL Server Management Studio foi então empregado para a implementação efetiva da estrutura do banco de dados no servidor SQL Server, adaptando os códigos conforme necessário. Para simular um ambiente real, foram inseridos dados fictícios nas tabelas do banco, seguindo diretrizes definidas.

Por fim, consultas SQL foram elaboradas e executadas para análise e extração de informações do banco de dados, abordando desde dados específicos de pessoas físicas e jurídicas até análises de movimentações de produtos. Essa abordagem integrada entre modelagem visual e implementação prática proporcionou uma transição suave e coesa para o desenvolvimento do banco de dados, facilitando a visualização conceitual e garantindo uma implementação efetiva no ambiente de servidor.

5. Resultado

Em resumo, a modelagem de dados é uma etapa crucial no processo de desenvolvimento de sistemas de banco de dados, pois ajuda a garantir que o design do banco de dados atenda aos requisitos do usuário e seja eficiente, escalável e fácil de manter.

Com isso, segue logo abaixo todas as informações da prática de modelagem e implementação do banco para a elaboração do projeto fictício de uma loja.

Na prática, foi utilizado o DB Designer Online, para a criação da estrutura do banco de dados SQL Server Management Studio com dados também fictícios para implementar as tabelas e chegar a um resultado semelhante ao projeto real.

O início do projeto foi dedicado a modelagem dos dados, a parte conceitual elaborada no DB Designer Online, onde foram inseridas todas as informações necessárias para a construção da estrutura do sistema da loja. Segue abaixo a modelagem de dados:

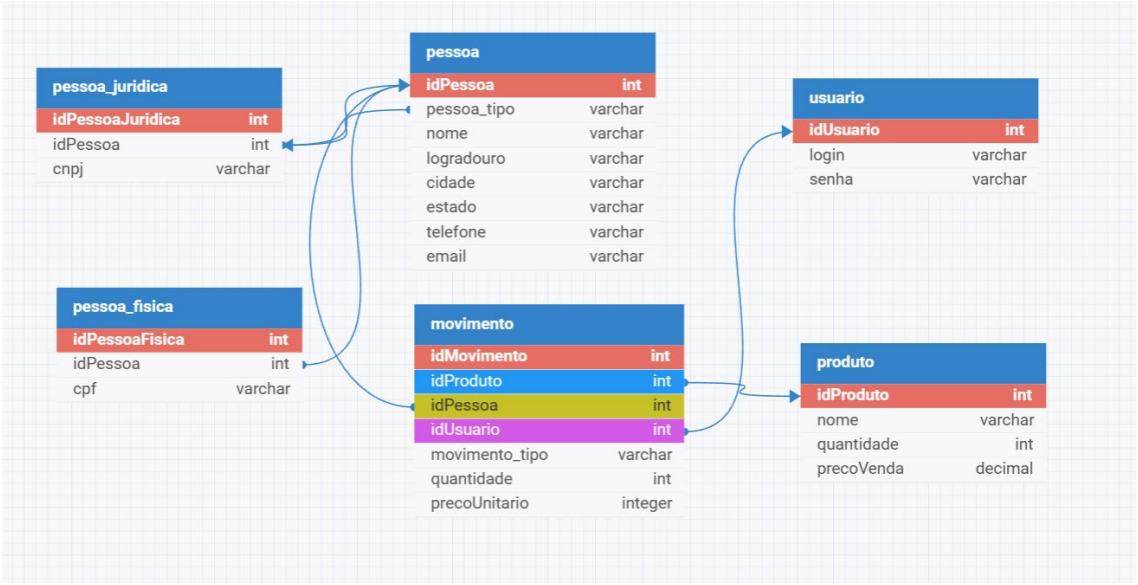

Imagem 01 - Modelagem de dados

Na imagem acima, mostra a visão geral da modelagem de dados, com todas as suas entidades e seus principais atributos compostos. Nela consta também, todas as ligações de cada atributos das entidades.

Com base na modelagem e suas interligações, podemos classificar as dependências de uma tabela para a outra como herança, assunto que foi abortado na disciplina anterior.

Podemos perceber que a mesma, herda informações de outras tabelas como o idPessoa das tabelas pessoa_fisica e pessoa_juridica e também da tabela usuário.

Tendo em vista a modelagem, podemos notar as ligações de uma tabela para a outra, quais são as chaves primárias e as estrangeiras.

Com base em todas essas informações, vamos iniciar os procedimentos de criação das tabelas.

```
USE master;
GO

CREATE USER loja FOR LOGIN loja;
GO

CREATE DATABASE Loja;
GO

USE Loja;
GO
```

Imagem 02 – Criação de usuário e banco de dados

Logo em seguida, depois que o banco de dados foi criado, chegou a vez de criar as tabelas pessoa, pessoa_fisica, pessoa_juridica, movimento, produto e

usuário. A tabela pessoa, herda e armazena as informações das tabelas pessoa_fisica e pessoa_juridica. A tabela produto armazena as informações de todos os produtos registrados pelo usuário e a tabela movimento, registra todas as transações realizadas.

```sql
USE master;
GO

CREATE USER loja FOR LOGIN loja;
GO

CREATE DATABASE Loja;
GO

USE Loja;
GO


CREATE TABLE pessoa(
idPessoa INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
pessoa_tipo VARCHAR(100),
nome VARCHAR(255) NOT NULL,
logradouro VARCHAR(255) NOT NULL,
cidade VARCHAR(255) NOT NULL,
estado CHAR(2) NOT NULL,
telefone VARCHAR(11) NOT NULL,
email VARCHAR(255) NOT NULL,
CONSTRAINT check_tipo_pessoa CHECK (pessoa_tipo IN ('fisica', 'juridica'))
)
GO

CREATE TABLE pessoa_fisica(
idPessoaFisica INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
idPessoa INT NULL,
cpf VARCHAR(11) NOT NULL,
FOREIGN KEY (idPessoa) REFERENCES pessoa(idPessoa)
)

CREATE TABLE pessoa_juridica (
idPessoaJuridica INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
idPessoa INT NULL,
cnpj VARCHAR(14) NOT NULL
FOREIGN KEY (idPessoa) REFERENCES pessoa(idPessoa)
)

CREATE TABLE produto (
idProduto INT PRIMARY KEY NOT NULL,
nome VARCHAR(255) NOT NULL,
quantidade INT NOT NULL,
precoVenda DECIMAL(10,2) NOT NULL
)

CREATE TABLE movimento (
idMovimento INT PRIMARY KEY NOT NULL,
idProduto INT NOT NULL,
idPessoa INT NOT NULL,
idUsuario INT NOT NULL,
movimento_tipo VARCHAR(100), -- 'E' para entrada, 'S' para saída
quantidade INT NOT NULL,
precoUnitario DECIMAL(10,2) NOT NULL,
CONSTRAINT check_tipo_movimento CHECK (movimento_tipo IN ( 'entrada', 'saida')),
FOREIGN KEY (idProduto) REFERENCES produto(idProduto),
FOREIGN KEY (idPessoa) REFERENCES pessoa(idPessoa),
FOREIGN KEY (idUsuario) REFERENCES usuario(idUsuario)
)

CREATE TABLE usuario (
idUsuario INT UNIQUE NOT NULL,
login VARCHAR(255) NOT NULL,
senha VARCHAR(255) NOT NULL
)
```

Imagem 03 – Estrutura das tabelas

Depois que as tabelas foram criadas, vem o momento que os dados fictícios são inseridos para compor as tabelas, dar vida a todo o processo criado, simulando um sistema de loja com seus usuários, produtos, fornecedores e compradores. Segue abaixo os códigos para inserir os dados nas tabelas.

```
USE Loja;
GO

INSERT INTO [usuario](idUsuario, login, senha)
VALUES
    (1,'silviasantos', 'silvia1234'),
    (2, 'juciara23', 'juju198'),
    (3, 'luciana242', 'lulu2345'),
    (4, 'rosenilda44678', 'rose3124')

INSERT INTO [produto](idProduto, nome, quantidade, precoVenda)
VALUES
    (1,'Banana', 100, 5.00),
    (2,'Laranja', 500, 2.00),
    (3,'Manga', 800, 4.00),
    (4, 'Pêra', 450, 9.00)

--Inserir pessoas--
INSERT INTO [pessoa] ( pessoa_tipo,  nome, logradouro, cidade, estado, telefone, email)
VALUES ( 'fisica', 'Silvia Santos Conceição', 'Rua Jubiramba Castro', 'São Paulo', 'SP', '11982377465', 'silvia@email.com');

INSERT INTO [pessoa_fisica] (idPessoa, cpf) VALUES (1, 37570270726);
--------
INSERT INTO [pessoa] ( pessoa_tipo, nome, logradouro, cidade, estado, telefone, email)
VALUES ('fisica', 'Judiciara Albuquerque', 'Avenida Julho Maia 4983', 'Rio de Janeiro', 'RJ', '21934753764', 'judiciara@email.com');

INSERT INTO [pessoa_fisica] (idPessoa, cpf) VALUES (2, 09384752932);
--------
INSERT INTO [pessoa] ( pessoa_tipo, nome, logradouro, cidade, estado, telefone, email)
VALUES ( 'juridica', 'Luciana Costa', 'Travessa paz e amor', 'Belo Horizonte', 'MG', '31976756383', 'luciana@email.com');

INSERT INTO [pessoa_juridica] (idPessoa, cnpj) VALUES (3, 85665739393404);
--------
INSERT INTO [pessoa] (pessoa_tipo, nome, logradouro, cidade, estado, telefone, email)
VALUES ('juridica', 'Rosenilda do Carmo', 'Rua Fonte do Saber', 'Porto Alegre', 'RS', '51975730803', 'rosenilda@email.com');

INSERT INTO [pessoa_juridica] (idPessoa, cnpj) VALUES (4, 96783757389230);

-- Inserir movimentação fornecedor
INSERT INTO [movimento] (idMovimento, idProduto, idPessoa, idUsuario, movimento_tipo, quantidade, precoUnitario)
VALUES
    (1, 1, 1, 1, 'entrada', 20, 4.5),
    (2, 2, 2, 2, 'entrada', 15, 3.00),
    (3, 1, 1, 1, 'saida', 10, 3.35),
    (4, 1, 1, 1, 'entrada', 25, 4.50),
    (5, 1, 1, 1, 'saida', 5, 2.50)
```

Imagem 04 – A imagem mostra como inserir os dados fictícios nas tabelas e seus relacionamentos

Nessa parte, colocamos o código “USE Loja” para confirmar que estamos no banco de dados correto e logo em seguida, vamos inserindo os dados de usuário, com seu id, login e senha para liberar o acesso ao sistema Loja.

Depois inserimos os produtos com seus dados, no qual os fornecedores e os compradores iram fazer negócios de compra e venda.

Mais adiante, temos o cadastramento das pessoas, sejam elas como pessoa física ou jurídica, sendo compradores como pessoa jurídica e vendedor como pessoa física.

A parte da movimentação, é onde fica armazenada toda a relacionamento de produtos, usuários, pessoas, a quantidade dos produtos e o preço unitário.

```sql
--Dados completos de pessoas físicas
SELECT pessoa.idPessoa, pessoa.nome, pessoa.logradouro, pessoa.cidade, pessoa.estado, pessoa.telefone, pessoa.email, pessoa_fisica.cpf
FROM pessoa INNER JOIN pessoa_fisica ON pessoa.idPessoa = pessoa_fisica.idPessoaFisica

--Dados completos de pessoas jurídica
SELECT pessoa.idPessoa, pessoa.nome, pessoa.logradouro, pessoa.cidade, pessoa.estado, pessoa.telefone, pessoa.email, pessoa_juridica.cnpj
FROM pessoa INNER JOIN pessoa_juridica ON pessoa.idPessoa = pessoa_juridica.idPessoaJuridica

--Movimentação de entrada
SELECT
P.nome AS Fornecedor, Prod.nome AS Produto, Mov.quantidade, Mov.precoUnitario AS PrecoUnitario,
Mov.quantidade * Mov.precoUnitario AS ValorTotal
FROM movimento Mov
JOIN Produto Prod ON Mov.idProduto = Prod.idProduto
JOIN Pessoa P ON Mov.idPessoa = P.idPessoa
WHERE movimento_tipo = 'entrada';

--Movimentação de saída
SELECT
P.nome AS Comprador, Prod.nome AS Produto, Mov.quantidade, Mov.precoUnitario AS PrecoUnitario,
Mov.quantidade * Mov.precoUnitario AS ValorTotal FROM Movimento Mov
JOIN Produto Prod ON Mov.idProduto = Prod.idProduto
JOIN Pessoa P ON Mov.idPessoa = P.idPessoa
WHERE movimento_tipo = 'saida';

--Valor total das entradas agrupadas por produto
SELECT Mov.idProduto, SUM(Mov.quantidade * Mov.precoUnitario) AS TotalEntradas
FROM Movimento Mov
WHERE movimento_tipo = 'entrada'
GROUP BY Mov.idProduto;

--Valor total das saídas agrupadas por produto
SELECT Mov.idProduto, SUM(Mov.quantidade * Mov.precoUnitario) AS TotalSaidas
FROM Movimento Mov
WHERE movimento_tipo = 'saida'
GROUP BY Mov.idProduto;

--Operadores que não efetuaram movimentações de entrada
SELECT DISTINCT U.*
FROM usuario U
LEFT JOIN Movimento M ON U.idUsuario = M.idUsuario AND movimento_tipo = 'entrada'
WHERE M.idUsuario IS NULL;

--Valor total de entrada, agrupado por operador
SELECT M.idUsuario, SUM(M.quantidade * M.precoUnitario) AS TotalEntradas FROM Movimento M
WHERE movimento_tipo = 'entrada'
GROUP BY M.idUsuario;

--Valor total de saída, agrupado por operador
SELECT M.idUsuario, SUM(M.quantidade * M.precoUnitario) AS TotalSaidas FROM Movimento M
WHERE movimento_tipo = 'saida'
GROUP BY M.idUsuario;

--Valor médio de venda por produto, utilizando média ponderada
SELECT
Prod.nome AS Produto, Mov.idProduto, SUM(Mov.quantidade * Mov.precoUnitario) / SUM(Mov.quantidade) AS MediaPonderada
FROM Movimento Mov
JOIN Produto Prod ON Mov.idProduto = Prod.idProduto
WHERE movimento_tipo = 'saida'
GROUP BY Mov.idProduto, Prod.nome;
```

Imagem 05 – Consultas das tabelas

Resultados | Mensagens

	idPessoa	nome	logradouro	cidade	estado	telefone	email	cpf
1	1	Silvia Santos Conceição	Rua Jubiramba Castro	São Paulo	SP	11982377465	silvia@email.com	37570270726
2	2	Judiciara Albuquerque	Avenida Julho Maia ...	Rio de J...	RJ	21934753764	judiciara@emai...	9384752932

	idPessoa	nome	logradouro	cidade	estado	telefone	email	cnpj
1	1	Silvia Santos Conceição	Rua Jubiramba Castro	São Paulo	SP	11982377465	silvia@email.com	85665739393404
2	2	Judiciara Albuquerque	Avenida Julho Maia 4983	Rio de Janeiro	RJ	21934753764	judiciara@email.com	96783757389230

	Fornecedor	Produto	quantidade	PrecoUnitario	ValorTotal
1	Silvia Santos Conceição	Banana	20	4.50	90.00
2	Judiciara Albuquerque	Laranja	15	3.00	45.00
3	Silvia Santos Conceição	Banana	25	4.50	112.50

	Comprador	Produto	quantidade	PrecoUnitario	ValorTotal
1	Silvia Santos Conceição	Banana	10	3.35	33.50
2	Silvia Santos Conceição	Banana	5	2.50	12.50

	idProduto	TotalEntradas
1	1	202.50
2	2	45.00

	idProduto	TotalSaidas
1	1	46.00

	idUsuario	login	senha
1	3	luciana242	lulu2345
2	4	rosenilda44678	rose3124

	idUsuario	TotalEntradas
1	1	202.50
2	2	45.00

	idUsuario	TotalSaidas
1	1	46.00

	Produto	idProduto	MediaPonderada
1	Banana	1	3.066666

Imagem 06 – Resultado das consultas

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

a) **1 para 1 (1X1):** Isso é geralmente implementado usando uma chave estrangeira em uma das tabelas para referenciar a chave primária na outra tabela. Por exemplo, uma tabela de "Pessoa" pode ter uma chave primária única, e uma tabela "Endereço" pode ter uma chave estrangeira que referência essa chave primária.
b) **1 para N (1XN):** Isso é implementado da mesma forma que 1 para 1, mas várias linhas na tabela relacionada podem fazer referência à mesma linha na tabela principal. Por exemplo, uma tabela de "Departamento" pode ter várias linhas em uma tabela de "Funcionário" referenciando o mesmo departamento.
c) **N para N (NXN):** Isso é implementado usando uma tabela de junção que mapeia as chaves primárias das duas tabelas que estão relacionadas. Por exemplo, em um relacionamento "Estudante" e "Curso", você pode ter uma tabela de junção que lista quais estudantes estão matriculados em quais cursos.

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

a) Para representar herança em bancos de dados relacionais, uma abordagem comum é a utilização de uma tabela para cada classe concreta e uma tabela adicional para a classe base. Cada tabela de classe concreta terá uma chave estrangeira que referência a tabela da classe base. Isso é chamado de modelagem de herança por tabela. Por exemplo, em uma hierarquia de classes como "Veículo" sendo a classe base e "Carro" e "Moto" sendo classes derivadas, você teria uma tabela "Veículo" e tabelas separadas "Carro" e "Moto", cada uma referenciando a tabela "Veículo".

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

O SSMS oferece uma variedade de recursos que ajudam a melhorar a produtividade no gerenciamento de bancos de dados SQL Server, como:

a) Interface intuitiva e amigável para gerenciamento de bancos de dados.
b) Recursos de edição de esquema de banco de dados, como criação de tabelas, índices, procedimentos armazenados, etc., através de GUIs ou scripts.
c) Ferramentas de consulta SQL avançadas, incluindo realce de sintaxe, sugestões de código, execução de consultas e exibição de resultados.
d) Capacidade de monitorar e otimizar o desempenho do banco de dados através de ferramentas de perfil e diagnóstico.

e) Recursos de segurança para gerenciar usuários, permissões e auditoria.
f) Integração com outras ferramentas e serviços da Microsoft, como Azure, Power BI, entre outros.

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

a) **Sequence:** Uma sequence é um objeto no banco de dados que gera uma sequência de números exclusivos de acordo com uma especificação definida pelo usuário. Ela pode ser usada em várias tabelas e em várias colunas dentro de uma tabela. No entanto, o controle de acesso e o valor atual da sequence precisam ser gerenciados explicitamente pelo usuário.
b) **Identity:** O identity é uma propriedade de coluna que é usada para gerar automaticamente valores exclusivos para essa coluna. Geralmente é mais simples de usar do que uma sequence, pois o banco de dados cuida automaticamente da geração e controle dos valores

10. Qual a importância das chaves estrangerias para a consistência do banco?

As chaves estrangeiras desempenham um papel fundamental na garantia da integridade referencial e consistência dos dados em um banco de dados relacional. Elas estabelecem relacionamentos entre tabelas, garantindo que cada valor em uma coluna de chave estrangeira corresponda a um valor existente na coluna de chave primária de outra tabela. Isso impede a ocorrência de referências a dados inexistentes e ajuda a manter a integridade dos dados durante operações de inserção, atualização e exclusão.

11. Quais operadores do SQL pertencem à álgebra relacional e quais são definidos no cálculo relacional?

a) **Álgebra Relacional:** Alguns dos operadores da álgebra relacional incluem projeção, seleção, união, interseção, diferença, produto cartesiano e junção.
b) **Cálculo Relacional:** No cálculo relacional, os operadores são geralmente definidos como predicados ou expressões que descrevem conjuntos de tuplas. Exemplos incluem operadores de seleção, projeção, união, interseção e diferença.

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

a) O agrupamento em consultas é feito usando a cláusula GROUP BY, que agrupa as linhas de uma consulta com base nos valores de uma ou mais colunas. O requisito obrigatório ao usar GROUP BY é que todas as colunas selecionadas que não fazem parte de uma função de agregação devem estar presentes na cláusula GROUP BY. Isso garante que o SQL saiba como agrupar corretamente as linhas e calcular as funções de agregação, como SUM, COUNT, AVG, etc.

Segue abaixo o repositório no GitHub de todo o projeto:

alaimalmeida/vamosMaterAsInformacoes (github.com)