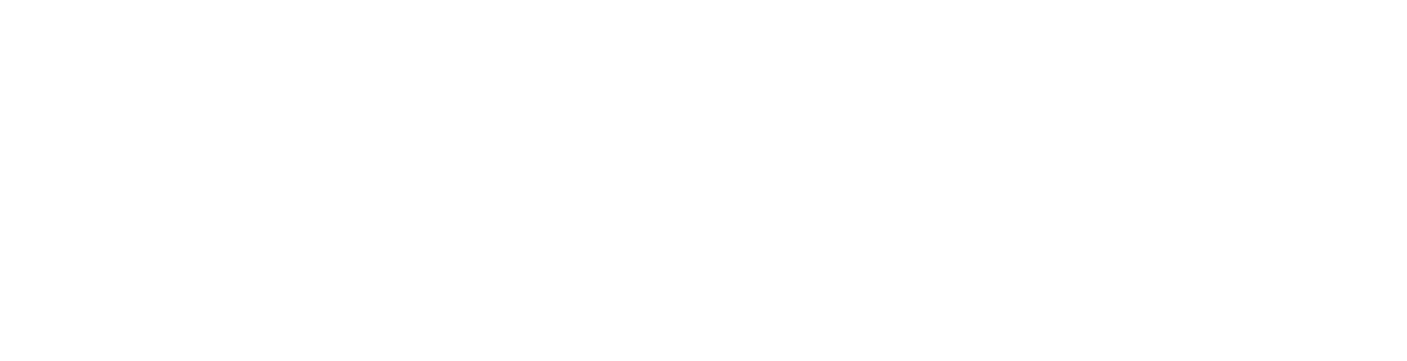


Base de Dados I

Trabalho de avaliação



Hugo Jeremias - 16399

DOCENTE: ELSA RODRIG

UES

Índice

[Índice de Figuras 2](#_Toc197773514)

[Introdução 3](#_Toc197773515)

[Diagrama entidade-relação 4](#_Toc197773516)

[Normalização de dados – 3ª Forma Normal 5](#_Toc197773517)

[Apresentar o Modelo Relacional 5](#_Toc197773518)

[Criação da base de dados 5](#_Toc197773519)

[Modelo Físico 5](#_Toc197773520)

[Inserção de dados em SQL 6](#_Toc197773521)

[Consultas em SQL 6](#_Toc197773522)

[Conclusão 6](#_Toc197773523)

[Bibliografia 6](#_Toc197773524)

# Índice de Figuras

[Figura 1 - Diagrama Entidade-Relação 4](#_Toc198911260)

[Figura 2 - Estrutura de tabelas em forma não normalizada 5](#_Toc198911261)

[Figura 3 - Base de dados na 3º forma normal sem relações 6](#_Toc198911262)

[Figura 4 - Base de dados na 3º forma normal com relações 7](#_Toc198911263)

# Introdução

O seguinte trabalho tem como objetivo o desenvolvimento de uma base de dados que ajude a cadeia de arranjos de roupa Fernandes & Lda a introduzir um sistema de gestão que suporte a sua atividade, inclusive o registo dos trabalhos solicitados pelos

clientes e respetivamente a emissão do recibo.

A aplicação deverá funcionar nas várias lojas, o gestor da empresa pretende, de qualquer

computador com acesso à Internet, ter a possibilidade de monitorizar a atividades das lojas e

efetuar algumas operações de gestão (como efetuar encomendas a fornecedores).

Os pedidos solicitados pelos clientes resultam de combinações de duas constantes: peça e

arranjo.

# Diagrama entidade-relação

O modelo de Entidade-Relação deste projeto tem como objetivo definir a estrutura geral da base de dados que ira´ ser implementada, delineando as relações entre as diferentes tabelas de forma clara e legível.

No mesmo podemos ver a definição e relações das entidades:  
- Cliente: Identifica quem efetua o pedido de arranjo, entrega as peças, aprova orçamentos caso existam, paga os recibos referentes ao trabalho, e levanta as peças;

- Pedido: Identifica as peças que foram entregues, o arranjo requisitado, a necessidade de orçamentação e o estado em que se encontra;

- DocumentoVenda: Identifica os documentos que são gerados pela aplicação para em nome do cliente;

- Loja: Identifica a loja em que os Documentos estão a ser emitidos, bem como o stock de artigos disponível;

- DocumentosCompra: Identifica os documentos que são emitidos pelos fornecedores em nome da empresa que também servem como entradas de stock para as diferentes lojas.

Apresentação do diagrama

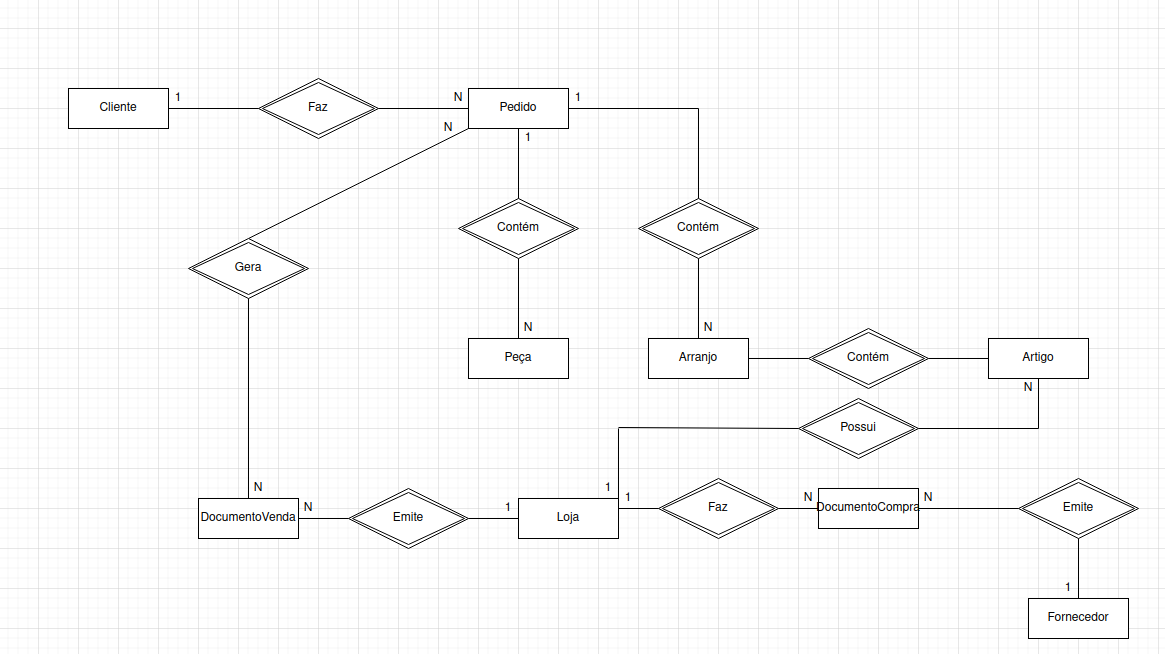


Figura - Diagrama Entidade-Relação

Apresentadas as relações entre entidades, e´ necessário agora qualificar as mesmas como tabelas de forma não normalizada.

Uma imagem com texto, número, diagrama, Paralelo

Os conteúdos gerados por IA poderão estar incorretos.

Figura - Estrutura de tabelas em forma não normalizada

# Normalização de dados – 3ª Forma Normal

Segue figura da base de dados na 3ª Forma Normal, sem relações

Uma imagem com texto, Paralelo, diagrama, número

Os conteúdos gerados por IA poderão estar incorretos.

Figura - Base de dados na 3º forma normal sem relações

## Apresentar o Modelo Relacional

Uma imagem com texto, diagrama, Esquema, Retângulo

Os conteúdos gerados por IA poderão estar incorretos.

Figura - Base de dados na 3º forma normal com relações

# Criação da base de dados

CREATE DATABASE FERNANDESELDA;

GO

USE FERNANDESELDA;

CREATE TABLE Cliente(

IDCliente INT NOT NULL PRIMARY KEY IDENTITY(1,1),

Nome varchar(50) NOT NULL,

NIF varchar(20) NOT NULL,

Telefone varchar(14) NOT NULL,

Morada varchar(100) NOT NULL,

Email varchar(50) NULL);

CREATE TABLE Loja(

IDLoja INT NOT NULL PRIMARY KEY IDENTITY(1,1),

Nome varchar(50) NOT NULL,

Morada varchar(100) NOT NULL);

CREATE TABLE Fornecedor(

IDFornecedor INT NOT NULL PRIMARY KEY IDENTITY(1,1),

Nome varchar(50) NOT NULL,

NIF varchar(20) NOT NULL,

Telefone varchar(14) NOT NULL,

Morada varchar(100) NOT NULL,

Email varchar(50) NULL);

CREATE TABLE Estado(

IDEstado INT NOT NULL PRIMARY KEY IDENTITY(1,1),

Nome VARCHAR(50) NOT NULL);

CREATE TABLE Pedido(

IDPedido INT NOT NULL PRIMARY KEY IDENTITY(1,1),

IDCliente INT NOT NULL,

IDEstado INT NOT NULL,

IDLoja INT NOT NULL,

Descricao varchar(250) NULL,

FOREIGN KEY (IDCliente) REFERENCES Cliente(IDCliente) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (IDEstado) REFERENCES Estado(IDEstado) ON DELETE NO ACTION ON UPDATE NO ACTION,

FOREIGN KEY (IDLoja) REFERENCES Loja(IDLoja) ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE Peca(

IDPeca INT NOT NULL PRIMARY KEY IDENTITY(1,1),

Cor VARCHAR(10) NOT NULL,

Tamanho VARCHAR(10) NOT NULL,

Tecido VARCHAR(15) NOT NULL);

CREATE TABLE PecaPedido(

IDPecaPedido INT NOT NULL PRIMARY KEY IDENTITY(1,1),

IDPedido INT NOT NULL,

IDPeca INT NOT NULL,

Quantidade INT NOT NULL,

FOREIGN KEY (IDPedido) REFERENCES Pedido(IDPedido) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (IDPeca) REFERENCES Peca(IDPeca) ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE TipoArranjo(

IDTipoArranjo INT NOT NULL PRIMARY KEY IDENTITY(1,1),

Nome varchar(50) NOT NULL,

Descricao varchar(250) NOT NULL);

CREATE TABLE Arranjo(

IDArranjo INT NOT NULL PRIMARY KEY IDENTITY(1,1),

IDTipoArranjo INT NOT NULL,

Duracao INT NOT NULL,

Preco FLOAT NULL,

FOREIGN KEY (IDTipoArranjo) REFERENCES TipoArranjo(IDTipoArranjo) ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE Artigo(

IDArtigo INT NOT NULL PRIMARY KEY IDENTITY(1,1),

Nome VARCHAR(50) NOT NULL);

CREATE TABLE ArranjoArtigo(

IDArranjoArtigo INT NOT NULL PRIMARY KEY IDENTITY(1,1),

IDArranjo INT NOT NULL,

IDArtigo INT NOT NULL,

Quantidade FLOAT NOT NULL,

FOREIGN KEY (IDArranjo) REFERENCES Arranjo(IDArranjo) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (IDArtigo) REFERENCES Artigo(IDArtigo) ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE ArranjoPedido(

IDArranjoPedido INT NOT NULL PRIMARY KEY IDENTITY(1,1),

IDPedido INT NOT NULL,

IDArranjo INT NOT NULL,

Observacoes varchar(250) NULL,

FOREIGN KEY (IDPedido) REFERENCES Pedido(IDPedido) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (IDArranjo) REFERENCES Arranjo(IDArranjo) ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE DocumentoVenda(

IDDocumentoVenda INT NOT NULL PRIMARY KEY IDENTITY(1,1),

IDPedido INT NOT NULL,

IDEstado INT NOT NULL,

IDTipoDoc INT NOT NULL,

FOREIGN KEY (IDPedido) REFERENCES Pedido(IDPedido) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (IDEstado) REFERENCES Estado(IDEstado) ON DELETE NO ACTION ON UPDATE NO ACTION);

CREATE TABLE DocumentoCompra(

IDDocumentoCompra INT NOT NULL PRIMARY KEY IDENTITY(1,1),

IDFornecedor INT NOT NULL,

IDLoja INT NOT NULL,

FOREIGN KEY (IDFornecedor) REFERENCES Fornecedor(IDFornecedor) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (IDLoja) REFERENCES Loja(IDLoja) ON DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE LinhasDocumentoCompra(

IDLinhasDocumentoCompra INT NOT NULL PRIMARY KEY IDENTITY(1,1),

IDDocumentoCompra INT NOT NULL,

IDArtigo INT NOT NULL,

Quantidade FLOAT NOT NULL,

FOREIGN KEY (IDDocumentoCompra) REFERENCES DocumentoCompra(IDDocumentoCompra) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (IDArtigo) REFERENCES Artigo(IDArtigo) ON DELETE CASCADE ON UPDATE CASCADE);

# Modelo Físico

*Figura 2 Modelo Físico*

# Inserção de dados em SQL

Comandos de inserção de dados iniciais para funcionamento da aplicação:

INSERT INTO Loja VALUES

('LOJA 1', 'RUA NUMERO 1'),

('LOJA 2', 'RUA NUMERO 2');

INSERT INTO Cliente VALUES

('CLIENTE 1', '999999990', '912345678', 'RUA NUMERO 1', 'cliente@1.pt'),

('CLIENTE 2', '999999990', '912345678', 'RUA NUMERO 2', 'cliente@2.pt'),

('CLIENTE 3', '999999990', '912345678', 'RUA NUMERO 3', 'cliente@3.pt'),

('CLIENTE 4', '999999990', '912345678', 'RUA NUMERO 4', 'cliente@4.pt');

INSERT INTO Artigo VALUES

('BOTAO'),

('FECHO'),

('LINHA ALGODAO BRANCA - ROLO'),

('LINHA AZUL - ROLO'),

('ALGODAO - M2'),

('LINHA POLYESTER BRANCA - ROLO'),

('SEDA - FIO'),

('SEDA - M2')

INSERT INTO TipoArranjo VALUES

('REMENDO', 'Aplicaçao de pedaço de tecido em buraco ou rasgao em peça (Cozido a maquina, ou prensado a calor)'),

('BAINHA', 'Corte na ponta da peça para diminuir o comprimento da mesma, incluindo a reconstruçao do ponto original da peça'),

('REPOSICAO/TROCA DE BOTOES', ''),

('REPARAÇAO GERAL', '');

INSERT INTO Arranjo VALUES

(1, 1, 15),

(1, 2, 25),

(1, 3, NULL),

(2, 1, 25),

(2, 2, 35),

(2, 3, NULL),

(3, 1, 15),

(3, 2, 25),

(3, 3, NULL),

(4, 1, NULL),

(4, 2, NULL),

(4, 3, NULL);

INSERT INTO ArranjoArtigo VALUES

(1, 3, 10),

(1, 4, 1),

(2, 3, 20),

(2, 4, 2),

(4, 6, 15),

(5, 6, 30),

(7, 1, 10),

(7, 6, 10),

(8, 1, 20),

(8, 6, 20);

INSERT INTO Estado VALUES

('Recebido'),

('Orçamentado'),

('Adjudicado'),

('Iniciado'),

('Em progresso'),

('Concluido'),

('Levantado'),

('Abandonado'),

('Pago'),

('Pendente Pagamento');

INSERT INTO Fornecedor VALUES

('FORNECEDOR 1', '999999990', '912345678', 'RUA NUMERO 1', 'FORNECEDOR@1.pt'),

('FORNECEDOR 2', '999999990', '912345678', 'RUA NUMERO 2', 'FORNECEDOR@2.pt'),

('FORNECEDOR 3', '999999990', '912345678', 'RUA NUMERO 3', 'FORNECEDOR@3.pt'),

('FORNECEDOR 4', '999999990', '912345678', 'RUA NUMERO 4', 'FORNECEDOR@4.pt');

INSERT INTO Peca VALUES

('Calças', 'Azul', 'L', 'Ganga'),

('T-shirt', 'Verde', 'S', 'Algodao'),

('Camisa', 'Branca', 'XL', 'Polyester');

INSERT INTO Pedido VALUES

(1, 1, 1, 'Remendo'),

(2, 1, 1, 'Bainha'),

(3, 1, 2, NULL);

INSERT INTO PecaPedido VALUES

(1, 2, 2),

(2, 1, 1),

(3, 3, 1);

INSERT INTO ArranjoPedido VALUES

(1, 2, 'Arranjo Tabelado - Remendo Simples'),

(2, 4, 'Arranjo Tabelado - Bainha Simples'),

(3, 7, 'Arranjo Tabelado - Reposiçao de botoes');

# Consultas em SQL

- CONSULTA 1: Pedidos com nome do cliente, estado e loja:

SELECT

Pedido.IDPedido,

Cliente.Nome AS NomeCliente,

Estado.Nome AS EstadoPedido,

Loja.Nome AS NomeLoja

FROM Estado

INNER JOIN (Loja

INNER JOIN (Cliente

INNER JOIN Pedido ON Cliente.IDCliente = Pedido.IDCliente)

ON Loja.IDLoja = Pedido.IDLoja)

ON Estado.IDEstado = Pedido.IDEstado;

- CONSULTA 2: Arranjos de um pedido:

SELECT

TipoArranjo.Nome AS TipoArranjo,

Arranjo.Duracao,

Arranjo.Preco,

ArranjoPedido.Observacoes

FROM TipoArranjo

INNER JOIN (Arranjo

INNER JOIN ArranjoPedido ON Arranjo.IDArranjo = ArranjoPedido.IDArranjo)

ON TipoArranjo.IDTipoArranjo = Arranjo.IDTipoArranjo

WHERE ArranjoPedido.IDPedido = [Insira o ID do Pedido];

- CONSULTA 3: Artigos usados num arranjo:

SELECT

Artigo.Nome AS NomeArtigo,

ArranjoArtigo.Quantidade

FROM Artigo

INNER JOIN ArranjoArtigo ON Artigo.IDArtigo = ArranjoArtigo.IDArtigo

WHERE ArranjoArtigo.IDArranjo = [Insira o ID do Arranjo];

- CONSULTA 4: Pedidos com peças e caracterização:

SELECT

Cliente.Nome AS NomeCliente,

Pedido.Descricao,

Peca.Tipo,

Peca.Cor,

Peca.Tamanho,

Peca.Tecido

FROM Cliente

INNER JOIN (Pedido

INNER JOIN (Peca

INNER JOIN PecaPedido ON Peca.IDPeca = PecaPedido.IDPeca)

ON Pedido.IDPedido = PecaPedido.IDPedido)

ON Cliente.IDCliente = Pedido.IDCliente;

- CONSULTA 5: Total de artigos utilizados:

SELECT

Artigo.Nome AS NomeArtigo,

SUM(ArranjoArtigo.Quantidade) AS TotalUsado

FROM Artigo

INNER JOIN ArranjoArtigo ON Artigo.IDArtigo = ArranjoArtigo.IDArtigo

GROUP BY Artigo.Nome;

- CONSULTA 6: Pedidos não pagos:

SELECT

DocumentoVenda.IDDocumentoVenda,

Pedido.IDPedido,

Cliente.Nome,

Estado.Nome AS Estado

FROM Cliente

INNER JOIN (Pedido

INNER JOIN (Estado

INNER JOIN DocumentoVenda ON Estado.IDEstado = DocumentoVenda.IDEstado)

ON Pedido.IDPedido = DocumentoVenda.IDPedido)

ON Cliente.IDCliente = Pedido.IDCliente

WHERE Estado.Nome <> 'Pago';

- CONSULTA 7: Artigos fornecidos por fornecedor:

SELECT

DocumentoCompra.IDDocumentoCompra,

Artigo.Nome,

LinhasDocumentoCompra.Quantidade

FROM DocumentoCompra

INNER JOIN (Artigo

INNER JOIN LinhasDocumentoCompra ON Artigo.IDArtigo = LinhasDocumentoCompra.IDArtigo)

ON DocumentoCompra.IDDocumentoCompra = LinhasDocumentoCompra.IDDocumentoCompra

WHERE DocumentoCompra.IDFornecedor = [Insira o ID do Fornecedor];

- CONSULTA 8: Consumo total de artigos por loja:

SELECT

Loja.Nome AS Loja,

Artigo.Nome AS Artigo,

SUM(ArranjoArtigo.Quantidade) AS TotalUsado

FROM Loja

INNER JOIN (Pedido

INNER JOIN (ArranjoPedido

INNER JOIN (ArranjoArtigo

INNER JOIN Artigo ON ArranjoArtigo.IDArtigo = Artigo.IDArtigo)

ON ArranjoPedido.IDArranjo = ArranjoArtigo.IDArranjo)

ON Pedido.IDPedido = ArranjoPedido.IDPedido)

ON Loja.IDLoja = Pedido.IDLoja

GROUP BY Loja.Nome, Artigo.Nome;

# Conclusão

# Bibliografia