BD: Trabalho Prático APF-T

Grupo: P1G2

- Filipe Barbosa, NMEC: 103064
- Rafael Pinto, NMEC: 103379

Introdução

O nosso projeto é baseado numa empresa que faz construção e manunteção de máquinas. Neste trabalho temos como objetivo criar uma base de dados que permita ao administrador da empresa gerir os pedidos de trabalho feito pelos clientes, gerir o seu stock de material e apontar os custos de cada trabalho.

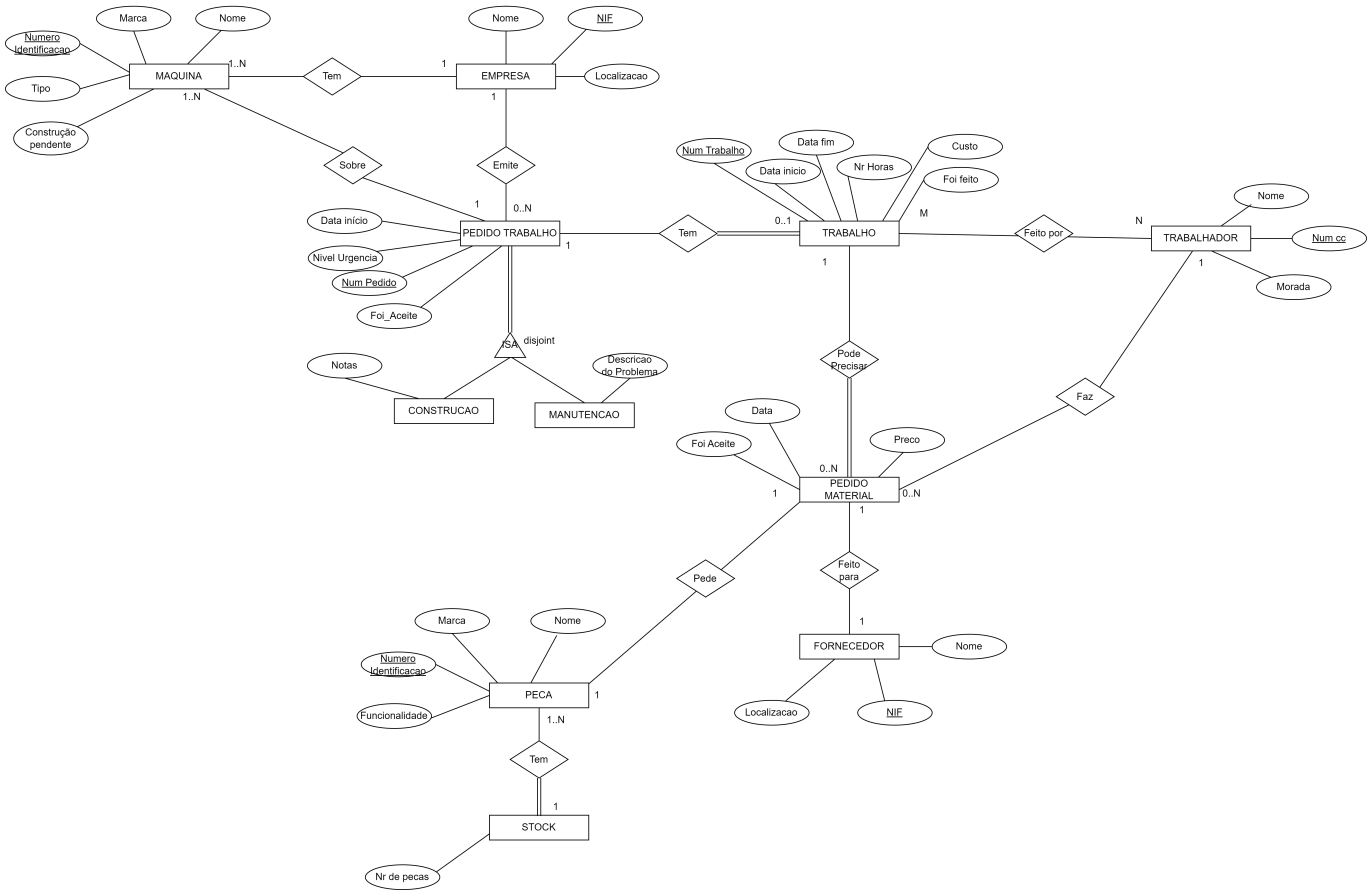
Análise de Requisitos

- O sistema tem de suportar três tipos de utilizador: Administrador da empresa, Trabalhadores da empresa e Clientes.
- Cada cliente (tabela Empresa) pode ter uma ou mais máquinas.
- O cliente pode fazer pedidos de trabalho para construir uma máquina nova ou fazer manutenção de uma existente.
- Um pedido de trabalho tem vários níveis de urgência (Pouco Urgente, Normal, Muito Urgente).
- O administrador pode aceitar/recusar pedidos de trabalho pendentes.
- Quando um pedido de trabalho é aceite este passa a ser um trabalho pendente.
- O administrador associa um dado trabalho a um ou mais trabalhadores.
- Um trabalhador pode estar associado a vários trabalhos.
- Um trabalho pode precisar um ou mais pedidos de material.
- Um pedido de material só pode ser sobre um tipo de peça, isto é, tem de haver um pedido por cada tipo de peça diferente.
- Todas as peças estão no stock, mesmo tendo 0 de quantidade.
- Um pedido de material pode ter um fornecedor caso não haja peças em stock

DER - Diagrama Entidade Relacionamento

- Uma **Máquina** é caracterizada por um número de identificação, um nome, uma marca, um tipo e se já foi ou não construída.
- Uma **Empresa** é caracterizada por um NIF, um nome e uma localização. Pode ter várias máquinas e pode emitir vários pedidos de trabalho.
- Uma **Pedido de Trabalho** é caracterizado por uma data, um número de pedido, um nível de urgência e um estado (pendente, aceite ou rejeitado). Pode ser um pedido de **Construção** ou um pedido de **Manutenção**.
- Um **Trabalho** é caracterizado por um número de trabalho, uma data de início, uma data de fim, um custo, um número de horas trabalhadas e se foi ou não feito. Pode ter vários **Pedidos de Material** e vários **Trabalhadores**.
- Um **Trabalhador** é caracterizado por um número de CC, um nome e uma morada. Pode estar associado a vários **Trabalhos** e pode fazer vários **Pedidos de Material**.
- Um **Pedido de Material** é caracterizado por uma data, um estado e um preço. É feito sobre uma **Peça** e pode ter um **Fornecedor**.
- Um **Forncedor** é caracterizado por um nIF, um nome e uma localização.
- Uma **Peça** é caracterizada por um número de identificação, um nome, uma marca e uma funcionalidade.
- Um **Stock** é caracterizado por uma quantidade e está associado a uma **Peça**.

Versão final



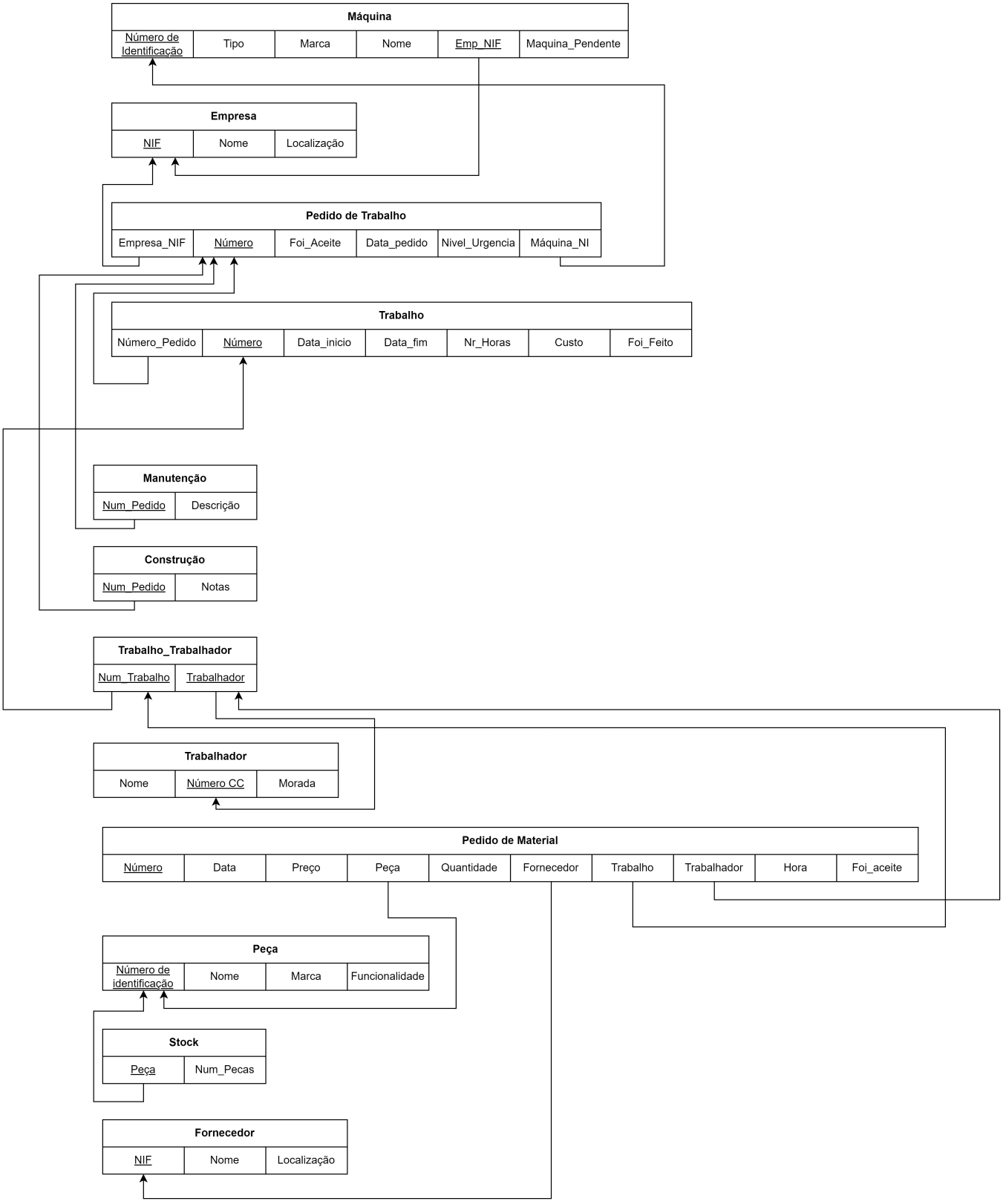
APFE

Depois de uma breve análise fez-se as seguintes alterações em relação à primeira entrega:

- Retirar a tabela "Despesas" visto não se enquadrar totalmente no foco do nosso trabalho.
- Um trabalhador poder fazer pedidos de material.

ER - Esquema Relacional

Versão final



APFE

Após a primeira entrega, as alterações feitas no diagrama entidade-relacionamento resultaram em alterações no esquema relacional. As alterações feitas foram:

- Colocar uma chave estrangeira "Trabalho" e "Trabalhador" na tabela Pedido de Material para que um trabalhador pudesse fazer um ou mais pedidos de material sobre um trabalho.
- Colocar uma chave estrangeira da tabela Empresa na tabela Maquina e não o contrário, visto que uma empresa pode ter várias máquinas mas uma máquina só pode pertencer a uma empresa.

SQL DDL - Data Definition Language

O código para a criação das tabelas do nosso projeto encontra-se em sql/01_ddl.sql.

Normalização

Como podemos verificar no esquema relacional, todas as tabelas estão na 3ª forma normal. Durante a construção do esquema relacional teve-se sempre atenção para que não existissem dependências transitivas nem dependências parciais.

Índices/Indexes

Os índices criados encontram-se no ficheiro sql/07_indexes.sql. Para se escolher que índices deveriam ser implementados, teve-se em conta quais eram as pesquisas que iriam ser feitas e chegou-se à conclusão que deveria ser possível pesquisar as peças e os trabalhadores pelo nome.

Índice	**Descrição**
pesquisa_nome	Índice na tabela Peca, coluna nome
pesquisa_trabalhador	Índice na tabela Trabalhador, coluna nome

SQL Programming: Views, Stored Procedures, Triggers, UDFs

Views

As views criadas encontram-se no ficheiro sql/05_views.sql.
Criaram-se nove views:

- lista de todos os trabalhadores;
- lista de todas as empresas;
- lista de todos os fornecedores;
- lista de todas as peças;
- lista de pedidos de material pendentes;
- lista de stock;
- lista de todos os trabalhos;
- lista de trabalhos pendentes;
- lista de trabalhos com trabalhadores associados;

Stored Procedures

As Stored Procedures criadas encontram-se no ficheiro sql/02_sp_functions.sql. Criou-se um conjunto de procedures que funcionam como uma camada de abstração entre a base de dados e interface, de forma a preservar a integridade da base de dados.

Stored Procedure	**Descrição**
inserir_construcao	Criar pedido de trabalho de construção. Recebe como parâmetros de entrada os dados necessários para realizar uma entrada nas tabelas Maquina, Pedido_de_Trabalho e Construcao. Foi implementado com uma Transaction uma vez que se faz três inserts em três tabelas diferentes
inserir_manutencao	Criar pedido de trabalho de manutenção. Recebe como parâmetros de entrada os dados necessários para realizar uma entrada nas tabelas Pedido_de_Trabalho e Construcao. Foi implementado com uma Transaction uma vez que se faz dois inserts em duas tabelas diferentes
inserir_nova_peca	Inserir nova peça. Recebe como parâmetros de entrada os dados necessários para inserir uma peça na Tabela Peca. Foi implementada com uma Transaction uma vez que faz dois inserts (um na tabela Peca e outro na tabela Stock)
inserir_pedido_de_material	Criar um pedido de material. Recebe como parâmetros de entrada os dados necessários para fazer um pedido de material e insere na tabela Pedido_de_Material
inserir_trabalho_aceite	Aceitar/Rejeitar pedido de trabalho. Verifica se o utilizador rejeitou ou aceitou o trabalho. Se recusou apenas dá update do estado foi_aceite da tabela Pedido_de_Trabalho para 'Rejeitado'. Se aceitou dá update do estado foi_aceite da tabela Pedido_de_Trabalho para 'Aceite' e insere o trabalho na tabela Trabalho. Foi implementado com uma Transaction, uma vez que se o pedido for aceite é necessário manipular duas tabelas
inserir_trabalho_trabalhador	Associa um trabalhador a um dado trabalho. Recebe como parâmteros de entrada o número de pedido e o numero de CC do trabalhador e cria uma entrada na tabela Trabalho_Trabalhador
ordenar_stock	Recebe como parâmetros de entrada o tipo de ordenação e retorna uma lista ordenada da tabela Stock pela ordem pretendida
ordenar_trabalhadores	Recebe como parâmetro de entrada o tipo de ordenação e retorna uma lista ordenada da tabela Trabalhadores pela ordem pretendida
update_material	Atualiza o stock quando um pedido de material é aceite. Recebe como parâmetro de entrada o número do pedido de material e atualiza as tabelas de Stock e de Pedido_de_Material. Foi implementada uma Transaction, porque são atualizadas duas tabelas diferentes

Stored Procedure	Descrição
concluir_trabalho	Usada para dar um trabalho como concluido. Recebe como parâmetros de entrada o número do trabalho, data final, custo e número de horas trabalhadas e atualiza a entrada correspondente da tabela Trabalho
add_fornecedor	Adiciona um fornecedor a um pedido de material. Recebe como parâmetros de entrada o número do pedido, o número do fornecedor e o preço e atualiza a entrada correspondente da tabela Pedido_de_Trabalho
add_stock	Adiciona peças ao stock de material. Recebe como parâmetros de entrada o número de identificação da peça e o número de peças a adicionar e atualiza entrada correspondente da tabela Stock
remove_trabalhador	Remove um trabalhador da base de dados. Recebe como parâmetros de entrada o número de cc do trabalhador e remove a entrada correspondente da tabela Trabalhador
insert_trabalhador	Insere um trabalhador na base de dados. Recebe como parâmetros de entrada os dados necessários para inserir um trabalhador na base de dados e insere o trabalhador na tabela Trabalhador

Triggers

Os Triggers criados encontram-se no ficheiro sql/03_triggers.sql.

Trigger	Descrição
verificar_data_pedidos	Verifica a data do pedido e se houver já três pedidos para o mesmo dia, não permite inserir mais nenhum pedido nesse dia. É do tipo Instead of
verify_update_material	Verifca se há stock suficiente para realizar a operação de update/insert na tabela Stock. É do tipo Instead of

UDFs

As UDFs criadas encontram-se no ficheiro sql/06_udfs.sql. Tal como as Stored Procedures, as UDFs funcionam como uma camada de abstração entre a base de dados e interface, de forma a preservar a integridade da base de dados.

UDF	Descrição
historico_pedidos	Recebe como parâmetros de entrada o NIF da empresa e retorna uma tabela com historico de pedidos dessa empresa
historico_pedidos_manutencao	Recebe como parâmetros de entrada o NIF da empresa e retorna uma tabela com historico de pedidos de manutencao dessa empresa
list_Machines	Recebe como parâmetro de entrada o NIF da empresa e retorna uma tabela com as máquinas dessa empresa

UDF	Descrição
pesquisa_peça	Recebe como parâmetro de entrada um texto e retorna uma tabela com as peças, da tabela Peca, com nome idêntico ao texto
pesquisar_trabalhador	Recebe como parâmetro de entrada um texto e retorna uma tabela cos trabalhadores, da tabela Trabalhador, com nome idêntico ao texto
trabalho_interface_trabalhador	Recebe como parâmetro de entrada o numero de CC do trabalhador e retorna uma tabela com os trabalhos associados ao mesmo
check_trabalhador	Recebe como parâmetro de entrada o numero de CC do trabalhador e retorna uma tabela com o utilizador correspondente
check_empresa	Recebe como parâmetro de entrada o NIF da empresa e retorna uma tabela com o utilizador correspondente
listar_pedidos_com_filtros	Recebe como parâmeto de entrada o estado do pedido e retorna uma tabela com os pedidos de trabalho com o estado pretendido
listar_pedidos	Retorna uma tabela com os pedidos de trabalho
listar_trabalhos	Retorna uma tabela com os trabalhos

Outras notas/Other notes

Local onde deve alterar o utilizador e a password da base de dados

Para mudar as credenciais da base de dados deve-se alterar o ficheiro Projeto/BDProxy.cs.
Deve alterar o uid e a password tanto na linha 16 como na linha 20. Segue-se a imagem referente ao trecho de código onde deve ser feita a alteração.

```
12  namespace BD_Interface
13  {
14      public class BDProxy
15      {
16          private SqlConnection cn = new SqlConnection("data source= tcp:mednat.ieeta.pt\\SQLSERVER,8101; initial catalog=p1g2; uid=p1g2; password=filipe.rafa2002");
17
18          private SqlConnection getSGBDConnection()
19          {
20              return new SqlConnection("data source= tcp:mednat.ieeta.pt\\SQLSERVER,8101; initial catalog=p1g2; uid=p1g2; password=filipe.rafa2002");
21          }
```

Dados iniciais da dabase de dados/Database init data

Database Init File