ISEL
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

SI2 Trabalho Prático - Primeira Parte

Inês Franco nº44860
Raul Santos nº44806
Diogo Bilé nº46908

Docente Nuno Datia

Relatório do Trabalho Prático 1- Primeira Parte da cadeira Sistemas de Informação 2,
do curso de licenciatura em Engenharia Informática e de Computadores
Semestre de Inverno 2021/2022

Dezembro de 2021

1.Introdução

Na primeira parte do trabalho prático foi pedida a realização de um sistema de informação para gestão de manutenção de ativos físicos Este sistema de informação deve cumprir todos os requisitos e regras de negócio para a empresa de manutenção Maintain4ver.
O código T-SQL para além da base de dados deveria conter 5 scripts, os quais representam as alíneas do trabalho pedidas. Os pedidos eram mecanismos de alteração dos dados numa tabela, procedimentos armazenados, vistas e funções.

Palavras-chave: T-SQL; Modelo de Dados; Base de Dados; Procedimentos Armazenados; Vistas; Funções; Tabelas.

2. Modelo de Dados

Iniciou-se o processo de elaboração da base de dados através do desenho do modelo de dados, com todas as suas restrições estabelecidas através da leitura cuidadosa do texto do enunciado.

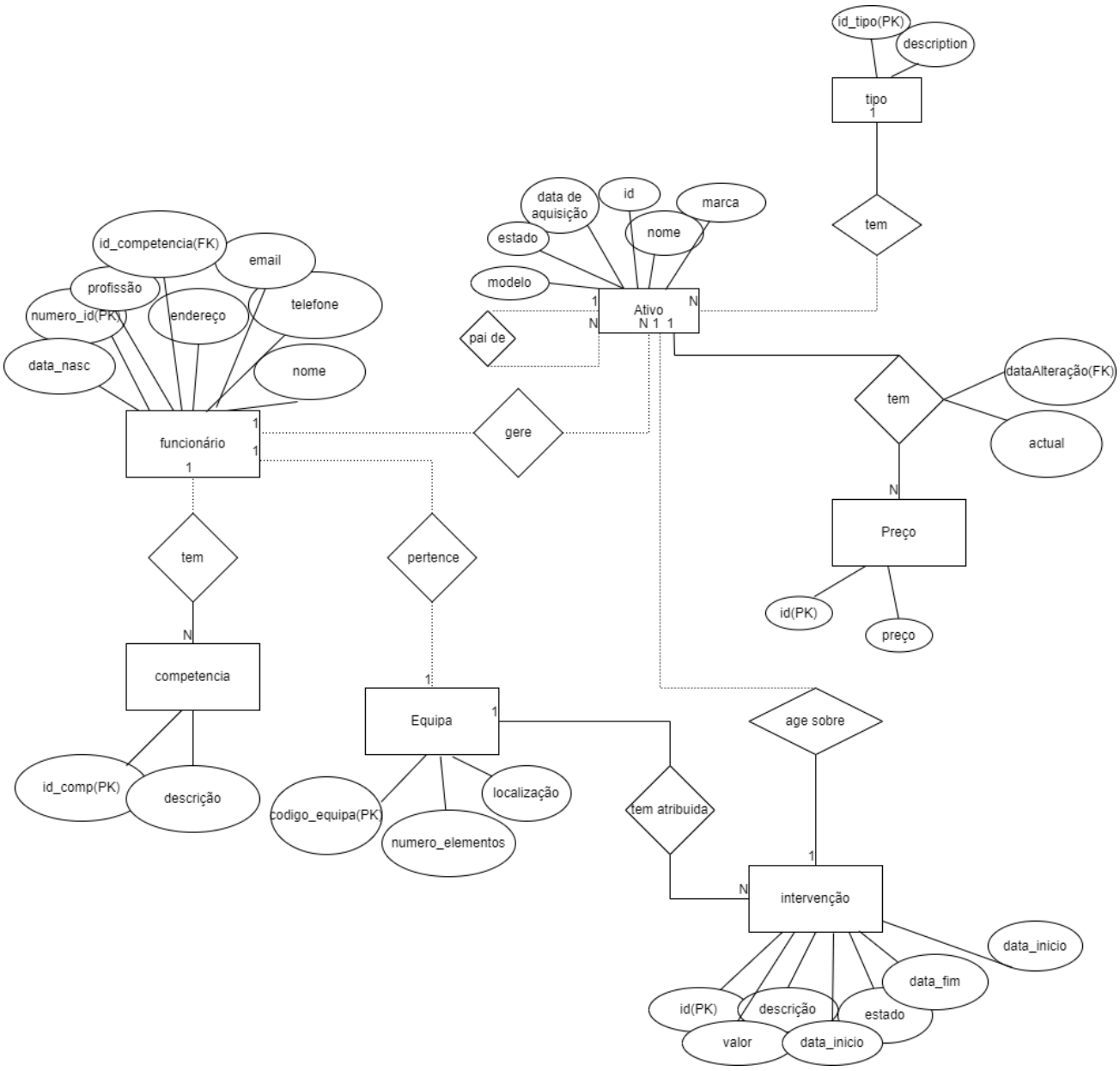

Figura 1 -Modelo de dados.

2.1 Restrições:

Restrições activos:

- A marca e o modelo do activo não são obrigatória
- Um activo é gerido por um funcionário obrigatoriamente
- O estado pode apenas ser ‘0’ ou ‘1’(logo será um bit)
- Um activo ‘pai’ tem que ser do mesmo tipo dos activos ‘filhos’

Restrições Funcionário:

- Um funcionário apenas pode pertencer a uma equipa
- A pessoa que gere o ativo não pode participar também na intervenção

Restrições Equipas:

- Pressupõe-se que se uma equipa permanece na base de dados mesmo quando esta não possui elementos
- Uma equipa apenas pode ter atribuída a si até 3 intervenções
- Uma equipa apenas pode fazer uma intervenção de cada vez
- Cada elemento da equipa tem registado as suas competências

Restrições Intervenções:

- Cada Intervenção atua sobre um activo
- A equipa que efectua a intervenção tem que ter pelo menos dois elementos
- O estado pode ser apenas: ‘por atribuir’, ‘em análise’, ‘em execução’ ou 'concluído'
- A data de intervenção deve ser superior à data de obtenção do activo.
- Uma intervenção apenas é atribuída a uma equipa que pelo menos um dos membros tenha uma competência igual à descrição da intervenção

Após a normalização até à 3FN e criação das tabelas chegou-se ao seguinte diagrama de visualização.

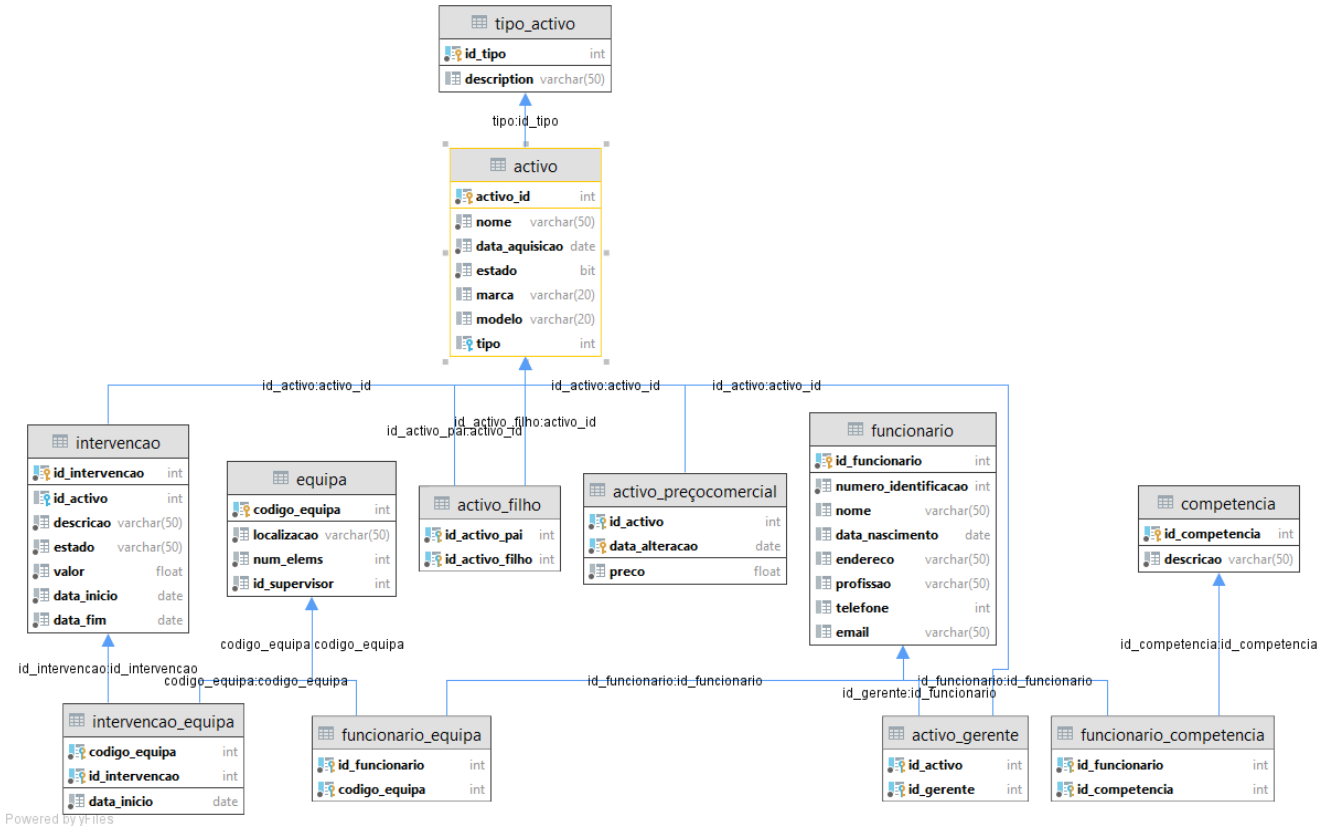

Figura 2 - Diagrama de Visualização

3. Implementação

Para concretizar inserções, remoções e atualizações sobre a entidade funcionária(ponto d) foram criados procedimentos armazenados visto que estas operações são realizadas com muita frequência e requerem manter a consistência na base de dados. Quando um funcionário é removido é necessário removê-lo também da equipa que possa estar associado, terá que sair também da tabela 'activo_gerente' antes de poder ser retirado da tabela funcionário. Para efetuar a atualização dos seus dados, decidiu-se que apenas o endereço, profissão, telefone e email poderiam ser atualizados. Para atualizar os dados apenas se passam ao procedimento armazenado os dados que se quer atualizar. Os argumentos estão em NULL como default.

Para encontrar uma equipa que possa efectuar uma intervenção foram criados um total de três funções:

- encontrarEquipaParaIntervencao:
    - Esta função recebe o id de uma intervenção e retorna o código da equipa capaz de satisfazer os requisitos. Itera sobre cada equipa e insere numa tabela custom as equipas que tenham as condições reunidas para efetuar a intervenção. Por cada equipa é chamada a função *verificarCompetenciasEquipa;*

- verificarCompetenciasEquipa:
    - Esta função recebe o id da equipa, a competência necessária e o id do activo para verificar se algum dos funcionários da equipa já é gerente do activo. Retorna um bit. Para facilitar na procura de uma equipa capaz criou-se uma vista que expõe o código da equipa, o número de elementos e a quantidade de intervenções já associadas a ela. Ao consultar esta vista consegue-se concluir se a equipa contém pelo menos 2 elementos e se já tem associada a si 3 intervenções. Caso alguma destas condições não esteja reunida é retornado de imediato um 0. Caso estejam reunidas a função itera sobre cada funcionário de forma a aferir se estes possuem as competências necessárias. Para Fazer esta verificação chama a função *verificarCompetenciasFuncionario*

- verificarCompetenciasFuncionario:
    - Esta função recebe o id do funcionário e a competência necessária e retorna um bit. O seu propósito é recolher todas as competências do funcionário da tabela ‘funcionario_competencia’ e verificar se alguma destas é igual à descrição da intervenção.

Na concretização dos pontos que necessitem de iterações sobre várias linhas de uma tabela, evitou-se utilizar Cursores pois estes têm um custo elevado no SQL Server. Foram usados cursores apenas na concretização do ponto k. Visto que a chamada sobre a view não irá ser frequente o suficiente para gerar overhead.

No procedimento armazenado ‘p_criaIntervencao’ coloca-se o estado do activo acabado de criar como ‘por atribuir’. Verifica-se também que a data de obtenção do activo é inferior à data de iniciação da intervenção e que a data de início é inferior à data de conclusão da intervenção.

Para efectuar o ponto k) foi criada uma vista que expõe os dados da intervenção e do activo sobre o qual esta age. Sobre esta foi criado um *trigger instead of update* que irá actuar quando há um *update* a mesma. Se na tabela *inserted* apenas existir uma intervenção então apenas se actualiza o estado dessa intervenção na tabela ‘intrevencao’ . Caso tenham sido feitos vários updates então é necessário criar um cursor para iterar sobre cada valor da tabela *inserted*. Por cada tuplo é chamado o procedimento armazenado ‘p_updateInter’ para atualizar o estado.

1. Conclusão

Neste trabalho desenvolveu-se competências no que toca a escolher o melhor mecanismo para resolver cada problema em SQL Server. Desenvolveu-se prática no uso de Procedimentos armazenados, Triggers, Funções e Cursores. Também houve oportunidade de converter restrições e regras de negócios num modelo de dados coeso.