Relatório Gestão de Vacinação Covid-19

Base de Dados

P1G1

António Domingues nº 89007

Henrique Silva nº 88857

Índice

Contextualização

A ideia de realizar uma base de dados relacionada com a temática Covid prende-se com a fase de pandemia que o mundo atravessa, surgindo desde logo o interesse de realizar um trabalho que tivesse principal foco neste tema. A gestão do processo de vacinação da Covid-19, pode ser bastante complexa chegando muitas vezes ao caótico. Assim sendo, foi abordado o tema, tentando reduzir toda a complexidade no que diz respeito à gestão do processo de vacinação e tentar criar uma base de dados que a tornasse mais ordeira.

A aplicação em questão disponibiliza um conjunto de ferramentas que permitem gerir um centro de vacinação. Assuntos como o stock de vacinas disponível, vacinas dadas por cada enfermeiro, agendamentos de marcação, entre outros foram abordados neste trabalho, utilizando sempre que possível, todos os recursos disponibilizados e aprendidos na unidade curricular de Base de Dados.

Análise de requisitos

Os requisitos pensados aquando do planeamento do projeto foram os seguintes:

- Um enfermeiro é caracterizado por nome, ID e mail.
- Uma marcação é definida por hora, número da marcação e data.
- Um paciente é identificado por profissão, endereço, nome, NIF, a vacina tomada, número de doses recebidas e idade.
- Uma vacina é composta por número de doses que têm que ser tomadas, nome, lote, validade e número de série.
- Um centro de vacinação é caracterizado por capacidade e endereço.
- Um enfermeiro notifica o paciente da marcação para a vacinação, e o paciente é responsável por aceitar e agendar essa marcação.
- O paciente recebe a vacina dada por um enfermeiro.
- Esta vacinação é feita num centro de vacinação.
- Cada centro de vacinação tem um enfermeiro responsável.

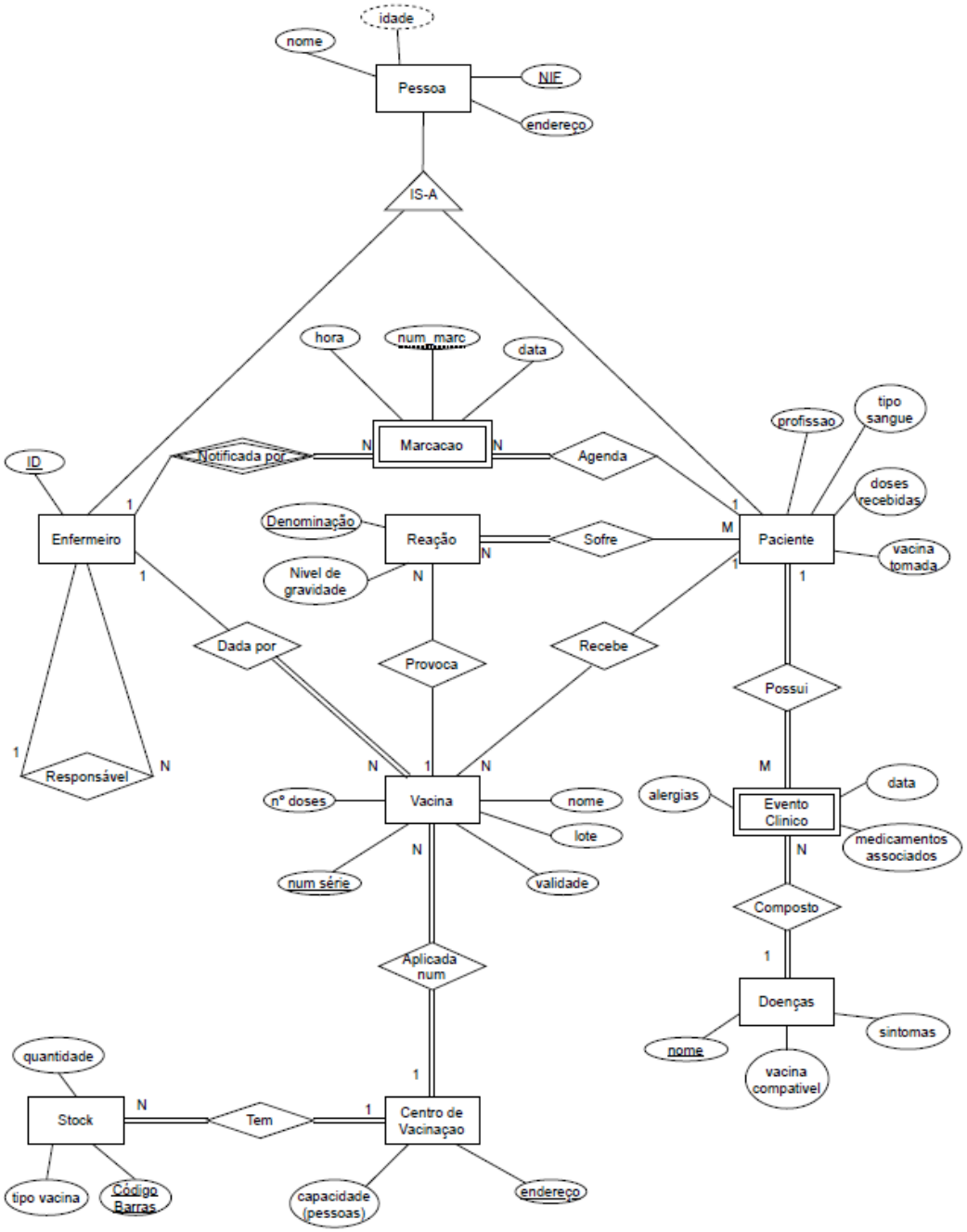
No entanto com o avançar do desenvolvimento do projeto alguns foram retirados, outros adicionados e outros atualizados. O resultado final foi o seguinte:

- Um enfermeiro é caracterizado por um ID e NIF.
- Uma marcação é definida por número da marcação, data (dia e hora), NIF do enfermeiro (que dá a vacina) e NIF do paciente (que recebe a vacina).
- Um paciente é identificado por NIF, profissão, tipo de sangue, doses recebidas e vacina que vai tomar ou que já tomou.
- Tanto um paciente como um enfermeiro têm uma relação IS-A com a entidade pessoa.
- Uma pessoa é caracterizada por NIF, idade, nome e endereço.
- Uma vacina é composta por número de série, doses, nome (da vacina), validade (em meses), endereço (onde foi aplicada), NIF do enfermeiro (que a dá) e NIF do paciente (que a recebe).
- Um centro de vacinação é caracterizado por capacidade (de pessoas) e endereço.
- Um stock é caracterizado por código de barras, quantidade, tipo de vacina e endereço (do centro de vacinação correspondente).
- No desenvolvimento do projeto foi criado apenas um centro de vacinação e todo o processo de vacinação ocorre nele.

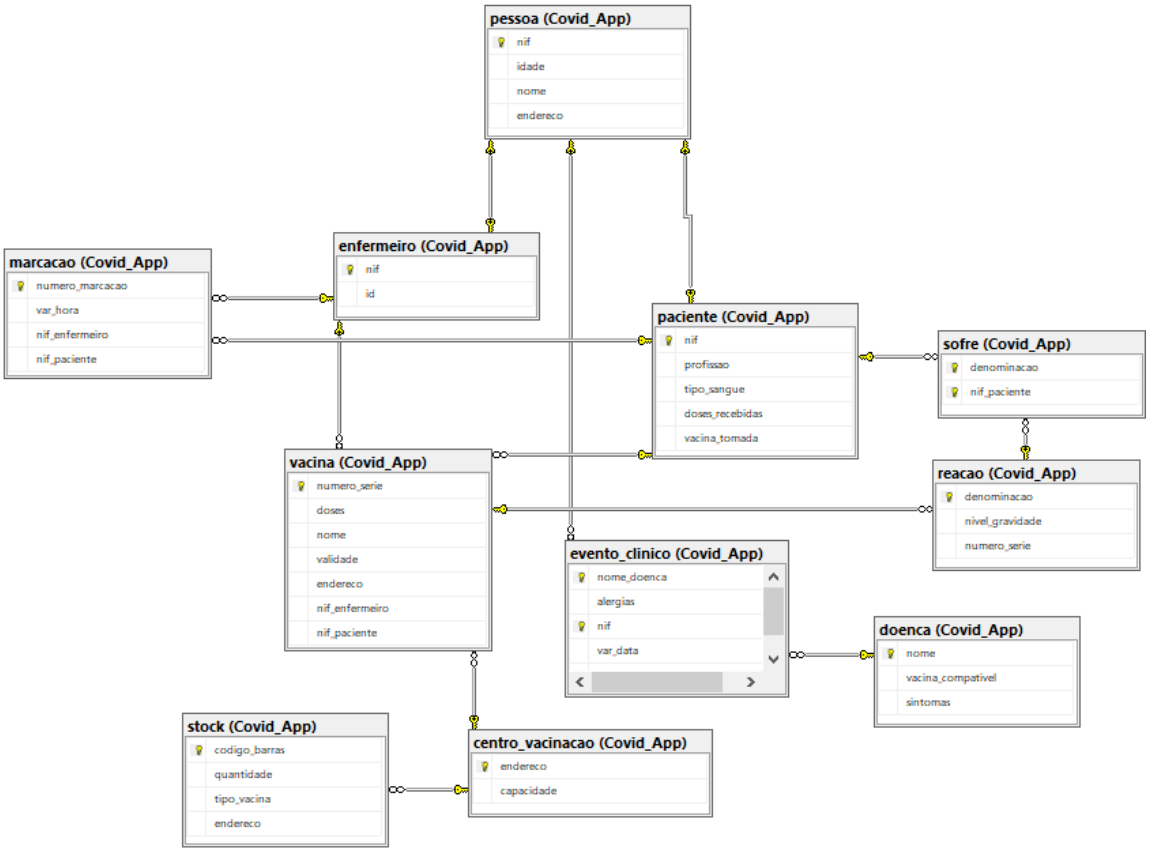
- Um paciente, após tomar uma vacina, tem uma reação. Essa reação tem como atributos uma denominação, um nível de gravidade (de 1 a 3, sendo 3 o pior) e um número de série (da vacina que provocou a reação).
- Para além disso, cada paciente tem também eventos clínicos, caracterizados por nome da doença, alergias, NIF (do paciente), data (do evento), e medicamentos associados.
- Cada doença tem um nome, uma vacina compatível e sintomas.

Diagrama Entidade-Relação

O DER apresentado abaixo representa todas as relações, entidades e atributos da aplicação. Em relação à proposta intermédia a evolução foi considerável e a complexidade aumentou bastante.



Esquema Relacional



Criação Tabelas DDL

Foram criadas em MySQL as tabelas correspondentes às entidades do diagrama e os respetivos atributos. De seguida foram criados objetos e inseridos nas tabelas. Tudo isto foi feito no ficheiro Covid_app_tables_schema.sql .

O exemplo apresentado a seguir corresponde à criação da tabela “evento_clinico” e inserção dos dados nas mesmas.

```
CREATE TABLE Covid_App.evento_clinico(
        nome_doenca VARCHAR(50) FOREIGN KEY REFERENCES Covid_App.doenca(nome) NOT NULL ,
        alergias VARCHAR(50),
        nif DECIMAL(9,0) FOREIGN KEY REFERENCES Covid_App.pessoa(nif) NOT NULL,
        var_data DATE NOT NULL,
        medicamentos_associados VARCHAR(100),
        PRIMARY KEY(nome_doenca, nif)
);
GO

INSERT INTO Covid_App.evento_clinico VALUES ('anemia', 'po, marisco', 149162536, '2013/5/16', null);
INSERT INTO Covid_App.evento_clinico VALUES ('artrite', 'po, marisco', 149162536, '2015/10/18', null);
INSERT INTO Covid_App.evento_clinico VALUES ('tiroide', 'leite, abelha', 563058403, '2002/07/02', null);
INSERT INTO Covid_App.evento_clinico VALUES ('asma', 'leite, abelha', 563058403, '1996/11/27', 'ventilan');
INSERT INTO Covid_App.evento_clinico VALUES ('artrite', 'leite, abelha', 563058403, '2008/12/28', 'voltaren');
INSERT INTO Covid_App.evento_clinico VALUES ('artrite', 'amendoim, marisco, soja', 981347909, '2014/06/17', null);
INSERT INTO Covid_App.evento_clinico VALUES ('osteoporose', 'amendoim, marisco, soja', 981347909, '2015/09/13', null);
INSERT INTO Covid_App.evento_clinico VALUES ('diabetes', 'amendoim, marisco, soja', 981347909, '2006/01/07', 'insulina');
INSERT INTO Covid_App.evento_clinico VALUES ('asma', 'po', 784741084, '2018/05/27', 'ventilan, maizar');
GO
```

Stored Procedures:

Para introduzir valores nas tabelas, posteriormente à sua criação, foram utilizadas Stored Procedures. Isto permite que sejam facilmente executados Inserts e Updates, úteis para popular a base de dados. Além disso, o processo torna-se mais rápido e fácil. De seguida são apresentadas as diversas Stored Procedures e os seus objetivos.

- **Covid_App.addPessoa:** Introduz os dados de uma nova pessoa na base de dados;
- **Covid_App.addEnfermeiro:** Adiciona um enfermeiro à base de dados;
- **Covid_App.addPaciente:** Adiciona um paciente à base de dados;
- **Covid_App.addMarcacao:** Cria uma nova marcação para um determinado paciente, numa determinada hora e com um determinado enfermeiro;
- **Covid_App.addVacina:** Cria uma nova vacina, associada a um paciente e dada por um enfermeiro;
- **Covid_App.addReacao:** Permite que seja adicionada uma reação a uma certa vacina;
- **Covid_App.addStock:** Permite incrementar o stock de certa vacina;
- **Covid_App.removeStock:** Permite decrementar o stock de certa vacina.

Os dois exemplos a seguir mostrados representam duas das stored procedures apresentadas, uma onde é realizado uma operação de insert e outra onde é realizada uma operação de update.

```
CREATE PROCEDURE Covid_App.addEnfermeiro (@nif DECIMAL(9,0))
AS
    BEGIN
        IF NOT EXISTS (SELECT * FROM Covid_App.pessoa WHERE pessoa.nif=@nif)
            BEGIN
                RAISERROR ('The person with that nif doesnt exists', 14, 1);
            END
        IF EXISTS (SELECT * FROM Covid_App.enfermeiro WHERE enfermeiro.nif=@nif)
            BEGIN
                RAISERROR ('The nurse with that nif already exists', 14, 1);
            END
        ELSE
            INSERT Covid_App.enfermeiro(nif)
            VALUES (@nif);
    END
GO
```

```
--Procedure para retirar stock de uma vacina
CREATE PROCEDURE Covid_App.removeStock (@quantidade int, @vacina varchar(20))
AS
    BEGIN
        IF NOT EXISTS (SELECT Covid_App.stock.tipo_vacina FROM Covid_App.stock WHERE stock.tipo_vacina = @vacina)
            BEGIN
                RAISERROR('Nao e possivel retirar stock para essa vacina', 14,1);
            END
        IF NOT EXISTS (SELECT Covid_App.stock.tipo_vacina FROM Covid_App.stock WHERE stock.quantidade > @quantidade)
            BEGIN
                RAISERROR('Nao e possivel retirar essa quantidade de stock para a vacina desejada', 14,1);
            END
        ELSE
            BEGIN
                UPDATE Covid_App.stock
                SET quantidade = quantidade - @quantidade
                WHERE Covid_App.stock.tipo_vacina = @vacina
            END
    END
GO
```

UDFS:

Foram também usadas UDF's com o objetivo de criar funções que retornassem e mostrassem certa informação. As UDF's criadas foram as seguintes:

- **Covid_App.fnGetNumVacinasEnfermeiroTable:** Devolve uma tabela com a quantidade de vacinas que cada enfermeiro deu;
- **Covid_App.fnGetVacinadosAtDate:** Devolve todas as pessoas que foram vacinadas numa certa data;
- **Covid_App.fnGetVacinasDadasEnfermeiro:** Tabela com as vacinas dadas para um determinado enfermeiro (recebe um enfermeiro como parâmetro de entrada)
- **Covid_App.fnGetMarcacoes**: Devolve todas as marcações;
- **Covid_App.fnGetNumNifMarcacoes:** Devolve uma tabela com as marcações de uma respetiva pessoa (pessoa como parâmetro de entrada)
- **Covid_App.fnGetVacCompativelPaciente:** Retorna a vacina recomendada para certo paciente.
- **Covid_App.fnGetReacaoPaciente:** Retorna a reação que certo paciente teve à vacina tomada.
- **Covid_App.fnGetEventoClinicoPaciente:** Retorna os eventos clínicos do paciente.
- **Covid_App.fnGetLocalVacinPaciente:** Informa qual o centro de vacinação onde o paciente foi vacinado (acabou por não ser usada na interface final pois foi criado apenas um centro de vacinação).
- **Covid_App.fnGetStockVacina:** Retorna o stock existente de uma determinada vacina.

O código apresentado corresponde à UDF que permite verificar o stock de uma vacina.

```
--Funcao que retorna o stock de uma vacina especifica
CREATE FUNCTION Covid_App.fnGetStockVacina(@vac AS varchar(20))
RETURNS @table TABLE(nome varchar(20), stock int)
AS
    BEGIN
        INSERT @table
            SELECT Covid_App.stock.tipo_vacina, Covid_App.stock.quantidade
            FROM Covid_App.stock Where stock.tipo_vacina = @vac
        RETURN;
    END;
GO
```

Triggers

O uso de triggers deveu-se à necessidade de garantir que certas condições eram verificadas, nomeadamente aquando da criação de uma nova pessoa, de um novo paciente e/ou enfermeiro. Para tal foram criados 3 triggers:

- **Covid_App.PessoaAvailability:** Trigger after insert. Verifica durante o processo de adição de uma pessoa à base de dados se a mesma já existe.
- **Covid_App.EnfermeiroAvailability:** Trigger after insert. Verifica durante o processo de adição de um enfermeiro à base de dados se o mesmo já existe. Verifica ainda se já existe uma pessoa com esse nif na base de dados, caso contrário não pode ser adicionado esse enfermeiro.
- **Covid_App.addPacienteTrigger:** Em tudo semelhante ao trigger do enfermeiro. Mas criado na tabela do paciente.

```
CREATE TRIGGER Covid_App.addPacienteTrigger ON Covid_App.paciente
AFTER INSERT
AS
    BEGIN
        DECLARE @nif DECIMAL(9,0);
        DECLARE @profissao varchar(50);
        DECLARE @sangue varchar(10);
        DECLARE @dosesRecebidas int;
        DECLARE @vacinaTomada varchar(20);
        DECLARE @Exists INT = 0;
        DECLARE @ExistsPaciente INT = 0;
        SELECT @nif = nif from inserted;
        SELECT @profissao = profissao from inserted;
        SELECT @sangue = tipo_sangue from inserted;
        SELECT @dosesRecebidas = doses_recebidas from inserted;
        SELECT @vacinaTomada = vacina_tomada from inserted;
        SELECT @Exists= COUNT(nif) FROM  Covid_App.pessoa WHERE Covid_App.pessoa.nif = @NIF
        SELECT @ExistsPaciente = COUNT(nif) FROM  Covid_App.paciente WHERE Covid_App.paciente.nif = @NIF
        IF(@Exists = 0)
        BEGIN
            RAISERROR('That person does not exist', 16, 1);
            ROLLBACK TRAN;
        END
        IF(@ExistsPaciente > 1)
        BEGIN
            RAISERROR('That patient already exist', 16, 1);
            ROLLBACK TRAN;
        END
    END
GO
```

Views

Foram usadas Views para retornar valores sem qualquer tipo de restrições, isto é, mais gerais e não de um objeto em específico. As views revelaram-se úteis ao serem usadas como um substituto de queries mais complexas, evitando desta forma o uso em demasia do Join. Para tal foram criadas as seguintes views:

Covid_App.Covid_App.ViewEnfermeiros: Obtém uma lista com a informação de todos os enfermeiros;

Covid_App.ViewMarcacoes: Mostra informação sobre todas as marcações;

Covid_App.viewGetPacientes: Retorna uma vista de todos pacientes;

Covid_App.viewGetStock: Mostra uma vista dos vários stocks de vacinas.

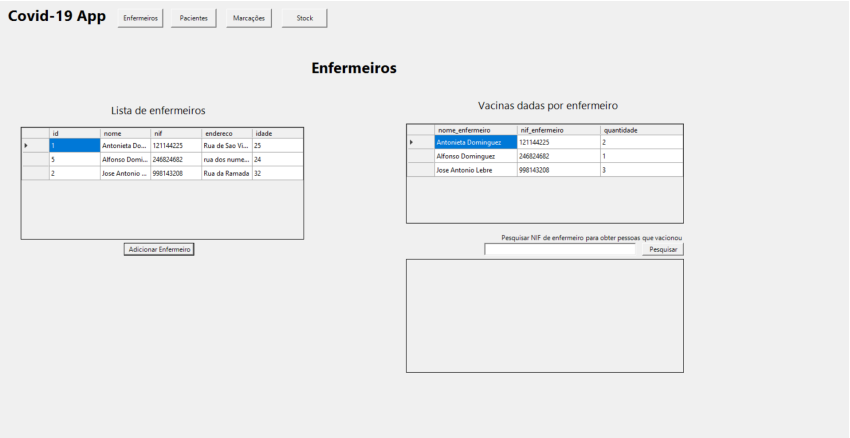
```
--VIEW que devolve o stock de vacinas
CREATE VIEW Covid_App.viewGetStock AS
    SELECT * FROM Covid_App.stock;
GO
--SELECT * FROM Covid_App.viewGetStock;
```

Visual Basic no Visual Studio

Para a criação da interface foi utilizado o Visual Basic e foram criados 10 forms:

- **AddEnfermeiro:** permite adicionar um novo enfermeiro à base de dados;
- **AddMarcacao:** permite adicionar uma nova marcação;
- **AddPaciente:** permite adicionar um novo paciente à base de dados;
- **AddStock:** permite incrementar o stock de uma certa vacina;
- **Enfermeiros:** são apresentados os enfermeiros da base de dados e o número de vacinas dadas por cada um. É possível pesquisar um enfermeiro e saber quem foram as pessoas que ele vacinou;
- **MainForm:** form onde estão apresentados os botões que levam aos restantes forms, assim como a database connection que aponta para a base de dados utilizada nas aulas;
- **Marcacoes:** é apresentada a lista de marcações e é possível pesquisar as marcações existentes num certo dia;
- **Pacientes:** é apresentada a lista de pacientes. Permite procurar várias informações sobre um paciente em concreto - vacina tomada/que vai tomar, reação à vacina tomada e eventos clínicos;
- **RemoveStock:** permite decrementar o stock de uma certa vacina;
- **Stock:** apresenta o stock de cada vacina e permite pesquisar o stock de uma certa vacina.

De seguida são apresentados dois forms, Enfermeiros e AddEnfermeiro (que abre após clicar no botão “adicionar enfermeiro”).



Conclusão

O resultado final foi bastante congratulante. Alguns dos conteúdos aprendidos ao longo do semestre não foram aplicados no contexto deste projeto pois as temáticas utilizadas revelaram-se bastante úteis e suficientes para o que era pretendido. Sendo esta a primeira vez que houve contacto com bases de dados e mysql foram sentidas algumas dificuldades no processo de planeamento que se refletiram numa fraca e pouco complexa apresentação intermédia. No entanto, após algumas correções feitas pelo professor Joaquim Sousa Pinto, iniciou-se o desenvolvimento do projeto e as coisas começaram a fluir de forma mais natural. Tal como dito anteriormente o resultado final foi deveras positivo e acabou por ir de encontro ao compromisso feito na fase inicial.