Hospital Database (Parte 1)

Grupo 406

up201806538@fe.up.pt Henrique Manuel Ruivo Pereira
up201801011@fe.up.pt Iohan Xavier Sardinha Dutra Soares
up201806554@fe.up.pt Telmo Alexandre Espirito Santo Baptista

24 de Maio de 2020

Projeto BDAD - 2019/20 - MIEIC

Professora das Aulas Laboratoriais: Carla Alexandra Teixeira Lopes

Índice

1 Contexto

É pretendido modelar uma base de dados para um hospital com diversos tipos de serviços disponíveis.

O hospital é constituído por vários departamentos. Cada um destes tem nome, identificador, especializações e entidade responsável pelo departamento.

Staff e pacientes são pessoas, acerca das quais interessa saber o nome, o seu número de identificação único (cartão de cidadão ou equivalente), a sua morada, o seu contacto telefónico, o seu número de beneficiário, o seu sexo e a sua data de nascimento.

A entidade responsável por um departamento é um membro da staff do hospital, podendo este tratar-se de um médico, enfermeiro ou técnico no hospital. Cada membro do staff tem o seu código identificador no hospital.

Especificamente sobre os médicos, é necessário guardar o número do seu consultório no hospital (caso tenha) e a sua especialização (caso tenha). Sobre os enfermeiros e os técnicos, apenas é necessário guardar a sua especialização (caso tenha).

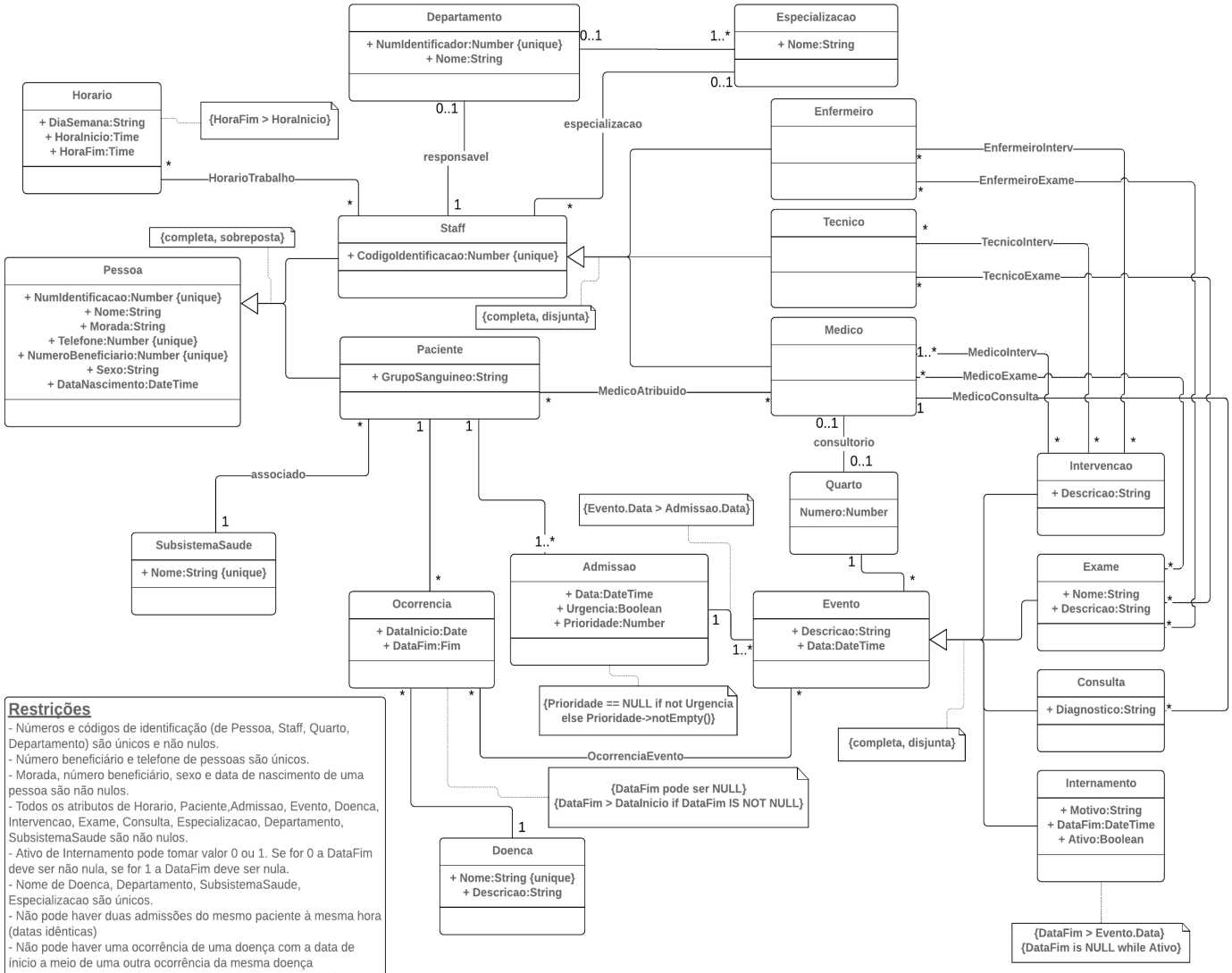
O hospital guarda informação sobre os seus pacientes como o seu grupo sanguíneo, o subsistema de saúde ao qual o paciente está associado, o histórico das ocorrências de doenças que o paciente teve, os médicos atribuídos naquele hospital, e as suas admissões no hospital.

Sobre cada doença interessa guardar o seu nome, uma descrição da doença.

Uma admissão no hospital tem uma data, se se trata de uma admissão de urgência e, caso seja uma urgência, a prioridade desta. Uma admissão pode desencadear vários tipos de eventos. Cada evento tem uma descrição sobre do que se trata, uma data, o quarto onde se realiza e outras informações dependendo do tipo de evento, que pode corresponder a: uma consulta, um exame, uma intervenção ou um internamento.

Numa consulta, interessa guardar o médico que a realizou e o diagnóstico da consulta. Num exame, guarda-se o nome do exame feito e uma descrição deste, bem como os médicos, enfermeiros e técnicos que dele participaram. Numa intervenção, guarda-se uma descrição da intervenção realizada, bem como os médicos, enfermeiros e técnicos envolvidos. Num internamento é necessário guardar o motivo deste, se ainda está ativo e, em caso negativo, a data na qual o paciente recebeu alta (saiu do internamento) caso tal já tenha acontecido.

2 UML - Modelo Conceptual



Restrições

- Números e códigos de identificação (de Pessoa, Staff, Quarto, Departamento) são únicos e não nulos.
- Número beneficiário e telefone de pessoas são únicos.
- Morada, número beneficiário, sexo e data de nascimento de uma pessoa são não nulos.
- Todos os atributos de Horario, Paciente,Admissao, Evento, Doenca, Intervencao, Exame, Consulta, Especializacao, Departamento, SubsistemaSaude são não nulos.
- Ativo de Internamento pode tomar valor 0 ou 1. Se for 0 a DataFim deve ser não nula, se for 1 a DataFim deve ser nula.
- Nome de Doenca, Departamento, SubsistemaSaude, Especializacao são únicos.
- Não pode haver duas admissões do mesmo paciente à mesma hora (datas idênticas)
- Não pode haver uma ocorrência de uma doença com a data de início a meio de uma outra ocorrência da mesma doença
- Um paciente não pode ter um novo internamento se ainda tiver outro ativo
- Um médico não pode ter duas consultas na mesma hora de início
- Uma admissão pode ser urgente ou não (1 ou 0), se for urgente deve ter uma prioridade maior que 0, se não deve ser 0
- Dado uma ocorrência que esteja relacionada a um evento, então o Paciente relacionado à Ocorrência e o Paciente relacionado à Admissão relacionada ao Evento devem ser iguais

3 Modelo Relacional

3.1 Classes

Classe Pessoa

- Pessoa(PessoaID, NumIdentificacao, Nome, Morada, Telefone, NumeroBeneficiario, Sexo, DataNascimento)

Classe Staff

- Staff(PessoaID→Pessoa, CodigoIdentificacao, Especializacao→Especializacao)
  - Enfermeiro(StaffID→Staff)
  - Tecnico(StaffID→Staff)
  - Medico(StaffID→Staff, Consultorio→Quarto)
- Especializacao(EspecializacaoID, Nome, Departamento→Departamento)
- Horario(HorarioID, DiaSemana, HoraInicio, HoraFim)
- Departamento(NumIdentificador, Nome, Responsavel→Staff)

Classe Paciente

- Paciente(PessoaID→Pessoa, GrupoSanguineo, SubsistemaSaude→SubsistemaSaude)
- SubsistemaSaude(SubsistemaSaudeID, Nome)

Classe Evento

- Admissao(AdmissaoID, Data, Urgencia, Prioridade, Paciente→Paciente)
- Doenca(DoencaID, Nome, Descricao)
- Ocorrencia(OcorrenciaID, DataInicio, DataFim, Paciente→Paciente, Doenca→Doenca)
- Quarto(Numero)
- Evento(EventoID, Descricao, Data, Admissao→Admissao, Quarto→Quarto)
  - Intervencao(EventoID→Evento, Descricao)
  - Exame(EventoID→Evento, Nome, Descricao)
  - Consulta(EventoID→Evento, Diagnostico, Medico→Medico)
  - Internamento(EventoID→Evento, Motivo, DataFim, Ativo)

3.2 Associações

Associações muitos-para-muitos

Associações relativas à classe Staff:

- HorarioTrabalho(StaffID→Staff, HorarioID→Horario)

Associações relativas à classe Paciente:

- MedicoAtribuido(PacienteID→Paciente, MedicoID→Medico)

Associações relativas à classe Evento:

- EnfermeiroInterv(EnfermeiroID→Enfermeiro, IntervID→Intervencao)
- EnfermeiroExame(EnfermeiroID→Enfermeiro, ExameID→Exame)
- TecnicoInterv(TecnicoID→Tecnico, IntervID→Intervencao)
- TecnicoExame(TecnicoID→Tecnico, ExameID→Exame)
- MedicoInterv(MedicoID→Medico, IntervID→Intervencao)
- MedicoExame(MedicoID→Medico, ExameID→Exame)
- OcorrenciaEvento(OcorrenciaID→Ocorrencia, EventoID→Evento)

Associações muitos-para-um

Para este tipo de associações foi adotado o método de adicionar uma chave estrangeira para a relação do lado "um"na relação do lado "muitos".

Associações um-para-um

Para este tipo de associações foi adicionada uma chave estrangeira à relação que possuiria o menor número de tuplos, exceto nos casos em que essa relação pudesse ter esse elemento como nulo, nos casos em que ambos podem ser nulos.

3.3 Generalizações

Quanto às generalizações, optou-se por usar o método E/R, criando uma relação para cada classe, e adicionando a chave da super-classe às subclasses, devido às restrições que necessitam ser implementadas nestas classes.

4 Análise de Dependências Funcionais e Formas Normais

Para cada relação será estudado se esta se encontra na forma normal Boyce-Codd e na 3ª forma normal.

Forma normal Boyce-Codd (BCNF)
Uma relação R com dependências funcionais FD está na forma normal de Boyce-Codd se:

- Para cada dependência funcional não trivial $\overline{A} \rightarrow \overline{B}$, $\overline{A}$ é uma (super)chave da relação R.

Terceira forma normal (3NF)
Uma relação R com dependências funcionais FD está na terceira forma normal se:

- Para cada dependência funcional não trivial $\overline{A} \rightarrow \overline{B}$, se se verificar pelo menos uma das seguintes propriedades:
  - $\overline{A}$ é uma (super)chave da relação R;
  - $\overline{B}$ consiste apenas de atributos primos (atributos que pertencem a pelo menos uma chave).

Vale notar que se uma relação estiver na forma normal Boyce-Codd (BCNF), esta também se encontra na terceira forma normal (3NF), já que as 3NF são um superconjunto das BCNF Se uma relação apenas consistir de dependências funcionais triviais, então a relação também está na forma normal de Boyce-Codd e por consequência na terceira forma normal.

Pessoa

- PessoaID→NumIdentificacao, Nome, Morada, Telefone, NumeroBeneficiario, Sexo, DataNascimento
- NumIdentificacao→PessoaID, Nome, Morada, Telefone, NumeroBeneficiario, Sexo, DataNascimento
- Telefone→PessoaID, NumIdentificacao, Nome, Morada, NumeroBeneficiario, Sexo, DataNascimento
- NumeroBeneficiario→PessoaID, NumIdentificacao, Nome, Morada, Telefone, Sexo, DataNascimento

Chaves candidatas:

$$\{PessoaID\}, \{NumIdentificacao\}, \{Telefone\}, \{NumeroBeneficiario\}$$

Como, para cada dependência não trivial, o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira forma normal (3NF)

Staff

- PessoaID→CodigoIdentificacao, Especializacao
- CodigoIdentificacao→PessoaID, Especializacao

Chaves candidatas:

$$\{PessoaID\}, \{CodigoIdentificacao\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Enfermeiro

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

Tecnico

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

Medico

- StaffID→Consultorio
- Consultorio→StaffID

Chaves candidatas:

$$\{StaffID\}, \{Consultorio\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Especializacao

- EspecializacaoID→Nome, Departamento
- Nome→EspecializacaoID, Departamento

Chaves candidatas:

$$\{EspecializacaoID\}, \{Nome\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Horario

- HorarioID→DiaSemana, HoraInicio, HoraFim
- DiaSemana, HoraInicio, HoraFim→HorarioID

Chaves candidatas:

$$\{HorarioID\}, \{DiaSemana, HoraInicio, HoraFim\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Departamento

- NumIdentificador→Nome, Responsavel
- Responsavel→NumIdentificador, Nome

Chaves candidatas:

$$\{NumIdentificador\}, \{Responsavel\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Paciente

- PessoaID→GrupoSanguineo, SubsistemaSaude

Chaves candidatas:

$$\{PessoaID\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

SubsistemaSaude

- SubsistemaSaudeID→Nome
- Nome→SubsistemaSaudeID

Chaves candidatas:

$$\{SubsistemaSaudeID\}, \{Nome\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Admissao

- AdmissaoID→Data, Urgencia, Prioridade, Paciente
- Data, Paciente→AdmissaoID, Urgencia, Prioridade

Chaves candidatas:

$$\{AdmissaoID\}, \{Data, Paciente\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Doenca

- DoencaID→Nome, Descricao
- Nome→DoencaID, Descricao

Chaves candidatas:

$$\{DoencaID\}, \{Nome\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Ocorrencia

- OcorrenciaID→DataInicio, DataFim, Paciente, Doenca
- DataInicio, Paciente, Doenca→OcorrenciaID, DataFim

Chaves candidatas:

$$\{OcorrenciaID\}, \{DataInicio, Paciente, Doenca\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Quarto

Não tem dependências funcionais.

Evento

- EventoID→Descricao, Data, Admissao, Quarto

Chaves candidatas:

$$\{EventoID\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Intervencao

- EventoID→Descricao

Chaves candidatas:

$$\{EventoID\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Exame

- EventoID→Nome, Descricao

Chaves candidatas:

$$\{EventoID\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Consulta

- EventoID→Diagnostico, Medico

Chaves candidatas:

$$\{EventoID\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

Internamento

- EventoID→Motivo, DataFim, Ativo

Chaves candidatas:

$$\{EventoID\}$$

Como para cada dependência não trivial o conjunto de atributos no lado esquerdo (A) é composto por uma key, então esta relação está na forma normal de Boyce-Codd (BCNF) e por consequência também se encontra na terceira form normal (3NF)

HorarioTrabalho

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

MedicoAtribuido

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

EnfermeiroInterv

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

EnfermeiroExame

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

TecnicoInterv

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

TecnicoExame

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

MedicoInterv

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

MedicoExame

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

OcorrenciaEvento

Não tem dependências funcionais não triviais, logo, esta relação encontra-se na forma normal de Boyce-Codd (BCNF) e na terceira forma normal (3NF)

5 Restrições

Pessoa

- cada pessoa tem um ID único (**PRIMARY KEY**)
- cada pessoa deve ter um número de identificação, nome, morada, número de beneficiário, telefone, sexo e data de nascimento (**NOT NULL**)
- o número de identificação e o número de beneficiário devem ser únicos (**UNIQUE**)

Staff

- cada membro da staff tem um ID único (**PRIMARY KEY**)
- além disso, PessoaID é chave estrangeira para Pessoa (**FOREIGN KEY**)
- não é possível eliminar uma pessoa enquanto houver staff mapeada para essa pessoa, mas pode-se alterar a pessoa que alterará na staff que a mapeia (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)
- Especializacao é chave estrangeira para Especializacao, e tal será posta a **NULL** caso a Especializacao seja eliminada, em caso de alterações, as alterações seram feitas em toda a staff que mapeia essa especialização (**ON DELETE SET NULL** e **ON UPDATE CASCADE**)

Enfermeiro

- cada enfermeiro tem o seu ID (**StaffID**) que é chave estrangeira para Staff (**FOREIGN KEY**)
- além disso, este ID deve ser único (**PRIMARY KEY**)
- não é possível eliminar um membro da staff enquanto houver um enfermeiro mapeado para esse membro da staff, mas pode-se alterar a staff que alterará também no enfermeiro que a mapeia (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)

Tecnico

- cada técnico tem o seu ID (**StaffID**) que é chave estrangeira para Staff (**FOREIGN KEY**)
- além disso, este ID deve ser único (**PRIMARY KEY**)
- não é possível eliminar um membro da staff enquanto houver um técnico mapeado para esse membro da staff, mas pode-se alterar a staff que alterará também no técnico que a mapeia (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)

Medico

- cada médico tem o seu ID (**StaffID**) que é chave estrangeira para Staff (**FOREIGN KEY**)
- além disso, este ID deve ser único (**PRIMARY KEY**)
- não é possível eliminar um membro da staff enquanto houver um médico mapeado para esse membro da staff, mas pode-se alterar a staff que alterará também no médico que a mapeia (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)
- Consultorio é chave estrangeira para Quarto (**FOREIGN KEY**)
- o consultório deve ser único (**UNIQUE**)
- caso se elimine um quarto que é mapeado como consultório, o valor do consultório é colocado a nulo, em caso de alteração, o consultório também é alterado (**ON DELETE SET NULL** e **ON UPDATE CASCADE**)

Especializacao

- cada especialização deve ter um ID único (**PRIMARY KEY**)
- nome da especialização deve ser único e não nulo (**NOT NULL** e **UNIQUE**)
- Departamento é chave estrangeira para Departamento
- caso se elimine um departamento, será colocado a nulo o valor do departamento em todas as especializações que o mapeiem, em caso de alteração, será alterado também (**ON UPDATE CASCADE** e **ON DELETE SET NULL**)

Horario

- cada horário tem um ID único (**PRIMARY KEY**)
- cada horário deve ter um dia da semana correspondente, uma hora de início e uma hora de fim (**NOT NULL**)
- não deve haver dois horários coincidentes (mesmo dia, mesma hora de ínicio e mesma hora de fim) (**UNIQUE**)
- a hora de início deve ser antes da hora de fim (**CHECK**)

Departamento

- cada departamento tem o seu número identificador único (**PRIMARY KEY**)
- cada departamento tem um nome e um responsável (**NOT NULL**)
- Responsavel é chave estrangeira para staff (**FOREIGN KEY**)
- não é possível remover um membro da staff caso esse esteja mapeado como um Responsavel de um departamento, mas é possível alterá-lo, tais alterações serão aplicadas no departamento que mapeia o membro da staff (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)
- não pode haver dois departamentos com um mesmo membro da staff responsável (**UNIQUE**)

Paciente

- cada paciente tem um número de identificação único (**PRIMARY KEY**)
- além disso, número de identificação (**PessoaID**) é chave estrangeira para Pessoa (**FOREIGN KEY**)
- não é possível eliminar uma pessoa enquanto houver um paciente mapeada para essa pessoa, mas pode-se alterar a pessoa que alterará no paciente que a mapeia (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)
- cada paciente tem um grupo sanguineo (**NOT NULL**)
- SubsistemaSaude é chave estrangeira para SubsistemaSaude (**FOREIGN KEY**)
- ao eliminar um subsistema de saúde colocará em todos os pacientes que o mapeavam o valor **NULL**, em caso de alteração, alterará também no paciente que o mapeia (**ON DELETE SET NULL** e **ON UPDATE CASCADE**)

SubsistemaSaude

- cada subsistema de saúde tem um ID único (**PRIMARY KEY**)
- nome do subsistema de saúde deve ser único e não nulo (**NOT NULL** e **UNIQUE**)

Admissao

- cada admissão tem um ID único (**PRIMARY KEY**)
- cada admissão tem uma data e um paciente (**NOT NULL**)
- uma admissão pode ser urgência ou não (isto é, tem o valor 1 ou 0, respetivamente), por omissão trata-se de uma admissão não urgente (**NOT NULL**, **CHECK** e **DEFAULT**)
- caso seja uma admissão de urgência a prioridade deve ter um valor maior que 0, em caso não urgente, a prioridade é 0 (**NOT NULL**, **CHECK** e **DEFAULT**)
- Paciente é chave estrangeira para Paciente (**FOREIGN KEY**)
- não é possível eliminar um paciente caso este esteja mapeado em Admissao, e alterar um paciente altera também em Admissao (**ON UPDATE CASCADE** e **ON DELETE RESTRICT**)
- um paciente não pode ter duas admissões na mesma hora do mesmo dia (datas idênticas) (**UNIQUE**)

Doenca

- cada doença tem um ID único (**PRIMARY KEY**)
- cada doença tem um nome, uma descrição e os seus sintomas (**NOT NULL**)
- o nome da doença deve ser único (**UNIQUE**)

Ocorrencia

- cada ocorrência tem um ID único (**PRIMARY KEY**)
- cada ocorrência tem uma data ínicio, um paciente e uma doença associada à ocorrência (**NOT NULL**)
- Paciente é chave estrangeira para Paciente, Doenca é chave estrangeira para Doenca (**REFERENCES**)
- não é possível eliminar uma doença ou um paciente se houver ocorrências a mapeá-los, mas ao alterar os mesmos, a informação também será alterada nas ocorrências que os mapeiam (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)
- se a data de fim for não nula, então deve ser depois da data de ínicio (**CHECK**)
- não pode haver ínicio de uma ocorrência de uma doença a meio de uma ocorrência dessa mesma doença (**TRIGGER**) (esta restrição foi implementada, nesta entrega, apenas considerando datas de ínicio iguais)

Quarto

- cada quarto tem um número único (**PRIMARY KEY**)

Evento

- cada evento tem um ID único (**PRIMARY KEY**)
- cada admissão tem uma descrição e uma data, a admissão ao qual está relacionada e o quarto onde foi procedido esse evento (**NOT NULL**)
- Admissao é chave estrangeira para Admissao, Quarto é chave estrangeira para Quarto (**FOREIGN KEY**)
- não é possível eliminar uma admissão nem um quarto enquanto houver ocorrências a mapeá-los, em caso de alteração, estas também são aplicadas nas ocorrências que os mapeiam (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)
- a data do evento deve ser maior que a data de admissão (**TRIGGER**)

Invervencao

- cada intervenção tem um ID (do evento) único (**PRIMARY KEY**)
- cada intervenção tem uma descrição (**NOT NULL**)
- o ID da intervenção (**EventoID**) é chave estrangeira para Evento (**FOREIGN KEY**)
- não é possível eliminar um evento se houver uma intervenção que o mapeia, em caso de alterações, estas também são aplicadas nas intervenções que os mapeiam (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)

Exame

- cada exame deve ter um ID (do evento) único (**PRIMARY KEY**)
- cada exame tem um nome e uma descrição (**NOT NULL**)
- o ID do exame (**EventoID**) é chave estrangeira para Evento (**FOREIGN KEY**)
- não é possível eliminar um evento se houver um exame que o mapeia, em caso de alterações, estas também são aplicadas nos exames que os mapeiam (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)

Consulta

- cada consulta deve ter um ID (do evento) único (**PRIMARY KEY**)
- cada consulta tem um médico associado e um diagnóstico (**NOT NULL**)
- o ID da consulta (**EventoID**) é chave estrangeira para Evento, Medico é chave estrangeira para Medico (**FOREIGN KEY**)
- não é possível eliminar um evento se houver uma consulta que o mapeia, em caso de alterações, estas também são aplicadas nas consulta que os mapeiam. O mesmo se aplica para a chave estrangeira Medico (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)
- um médico não pode ter duas consultas na mesma hora de ínicio (**TRIGGER**)

Internamento

- cada internamento deve ter um ID (do evento) único (**PRIMARY KEY**)
- cada internamento tem um motivo e se o internamento está ativo ou não (**NOT NULL**)
- o ID do internamento (**EventoID**) é chave estrangeira para Evento (**FOREIGN KEY**)
- não é possível eliminar um evento se houver um internamento que o mapeia, em caso de alterações, estas também são aplicadas nos internamento que os mapeiam (**ON DELETE RESTRICT** e **ON UPDATE CASCADE**)
- Ativo é um valor booleano, tomando o valor de 0 ou 1 (**CHECK**)
- se Ativo estiver a 1 então DataFim deve ser não nula, se tiver a 0 então DataFim deve ser nula (**CHECK**)
- DataFim deve ser maior que a Data do evento mapeado na chave estrangeira EventoID (**TRIGGER**)
- um paciente não pode ter um novo internamento se tiver um ativo (**TRIGGER**)

HorarioTrabalho

- cada intermante deve ter um ID de staff e um ID de horário único (**PRIMARY KEY**)
- StaffID é chave estrangeira para Staff e HorarioID chave estrangeira para Horario (**REFERENCES**)
- ao eliminar uma staff que esteja mapeada num HorarioTrabalho, eliminará também o HorarioTrabalho, em caso de alteração, também alterará em HorarioTrabalho. O mesmo aplica-se à chave estrangeira Horario (**ON DELETE CASCADE** e **ON UPDATE CASCADE**)

MedicoAtribuido

- cada intermante deve ter um ID de um paciente e um ID de um médico único (**PRIMARY KEY**)
- PacienteID é chave estrangeira para Paciente e MedicoID chave estrangeira para Medico (**FOREIGN KEY**)
- ao eliminar um paciente que esteja mapeada num MedicoAtribuido, eliminará também o MedicoAtribuido, em caso de alteração, também alterará em MedicoAtribuido. O mesmo aplica-se à chave estrangeira MedicoID (**ON DELETE CASCADE** e **ON UPDATE CASCADE**)

EnfermeiroInterv

- deve ter uma intervenção ao qual foi atribuído um enfermeiro (**PRIMARY KEY**)
- EnfermeiroID é uma chave estrangeira para Enfermeiro e IntervID é chave estrangeira para Intervencao
- ao eliminar um enfermeiro que esteja mapeado num EnfermeiroInterv, eliminará também o EnfermeiroInterv, em caso de alteração, também alterará em EnfermeiroInterv. O mesmo aplica-se à chave estrangeira IntervID (**ON DELETE CASCADE** e **ON UPDATE CASCADE**)

EnfermeiroExame

- deve ter um exame ao qual foi atribuído um enfermeiro (**PRIMARY KEY**)
- EnfermeiroID é uma chave estrangeira para Enfermeiro e ExameID é chave estrangeira para Exame
- ao eliminar um enfermeiro que esteja mapeado num EnfermeiroExame, eliminará também o EnfermeiroExame, em caso de alteração, também alterará em EnfermeiroExame. O mesmo aplica-se à chave estrangeira ExameID (**ON DELETE CASCADE** e **ON UPDATE CASCADE**)

TecnicoInterv

- deve ter uma intervenção ao qual foi atribuído um técnico (**PRIMARY KEY**)
- TecnicoID é uma chave estrangeira para Tecnico e IntervID é chave estrangeira para Intervencao
- ao eliminar um técnico que esteja mapeado num TecnicoInterv, eliminará também o TecnicoInterv, em caso de alteração, também alterará em TecnicoInterv. O mesmo aplica-se à chave estrangeira IntervID (**ON DELETE CASCADE** e **ON UPDATE CASCADE**)

TecnicoExame

- deve ter um exame ao qual foi atribuído um técnico (**PRIMARY KEY**)
- TecnicoID é uma chave estrangeira para Tecnico e ExameID é chave estrangeira para Exame
- ao eliminar um técnico que esteja mapeado num TecnicoExame, eliminará também o TecnicoExame, em caso de alteração, também alterará em TecnicoExame. O mesmo aplica-se à chave estrangeira ExameID (**ON DELETE CASCADE** e **ON UPDATE CASCADE**)

MedicoInterv

- deve ter uma intervenção ao qual foi atribuído um médico (**PRIMARY KEY**)
- MedicoID é uma chave estrangeira para Medico e IntervID é chave estrangeira para Intervencao
- ao eliminar um médico que esteja mapeado num MedicoInterv, eliminará também o MedicoInterv, em caso de alteração, também alterará em MedicoInterv. O mesmo aplica-se à chave estrangeira IntervID (**ON DELETE CASCADE** e **ON UPDATE CASCADE**)

MedicoExame

- deve ter um exame ao qual foi atribuído um médico (**PRIMARY KEY**)
- MedicoID é uma chave estrangeira para Medico e ExameID é chave estrangeira para Exame
- ao eliminar um médico que esteja mapeado num MedicoExame, eliminará também o MedicoExame, em caso de alteração, também alterará em MedicoExame. O mesmo aplica-se à chave estrangeira ExameID (**ON DELETE CASCADE** e **ON UPDATE CASCADE**)

OcorrenciaEvento

- deve ter um evento que esteja ligada a uma ocorrência (**PRIMARY KEY**)
- OcorrenciaID é uma chave estrangeira para Ocorrencia e EventoID é chave estrangeira para Evento
- ao eliminar uma ocorrência que esteja mapeado num OcorrenciaEvento, eliminará também o OcorrenciaEvento, em caso de alteração, também alterará em OcorrenciaEvento. O mesmo aplica-se à chave estrangeira EventoID (**ON DELETE CASCADE** e **ON UPDATE CASCADE**)
- o paciente relacionado com a ocorrência deve ser o mesmo relacionado com a admissão à qual o evento está relacionado (**TRIGGER**)

6 Interrogações à Base de Dados

6.1 Distribuição dos Grupos Sanguíneos dos Pacientes

A primeira interrogação tem como objetivo listar todos os grupos sanguíneos na base de dados e a distribuição destes sobre os pacientes, isto é, a percentagem de pacientes que possuem aquele tipo sanguíneo.

É apresentada então uma tabela com o grupo sanguíneo e a respetiva percentagem.

Operadores usados: *SELECT*, *FROM*, *GROUP BY*, *COUNT*, *SUM*, *OVER*, *AS*.

6.2 Média de horas de trabalho dos médicos ao longo da semana

A segunda interrogação lista todos os dia da semana e o número de horas de trabalho médio exercidos pelos médicos.

Se houver algum dia que não possui médicos a trabalhar, o dia será mostrado da mesma com o número de horas média a zero.

Operadores usados: *SELECT*, *FROM*, *GROUP BY*, *UNION*, *SUM*, *COUNT*, *AS*, *NATURAL JOIN*, *DISTINCT*, *WHERE*, *NOT IN*.

6.3 Tabela de Estatísticas das Doenças

A terceira interrogação lista uma tabela com estatísticas de doenças, contendo o nome da doença, a descrição, número médio de dias em que os pacientes tiveram a doença, o grupo sanguíneo mais afetado, percentagem de mulheres afetadas, a percentagem de homens afetados, a idade média das mulheres afetadas e a idade média dos homens afetados.

Se não houver registos de pacientes com uma certa doença, essa doença não é apresentada na tabela.

Operadores usados: *SELECT*, *FROM*, *GROUP BY*, *CREATE VIEW*, *AS*, *COUNT*, *SUM*, *ON*, *NATURAL JOIN*, *INNER JOIN*, *WHERE*, *DISTINCT*, *UNION*, *NOT IN*, *MAX*, *IS NOT*, *DROP VIEW*.

6.4 Distribuição do número de pacientes em cada mês

A quarta interrogação lista os meses do ano e o número de admissões de pacientes naquele mês, contabilizando todos os anos que se tem registo.

Operadores usados: *SELECT*, *COUNT*, *DISTINCT*, *AS*, *SUBSTR*, *FROM*, *GROUP BY*.

6.5 Médico atribuído a cada consultório

A quinta interrogação lista os médicos atribuídos a cada consultório.

Operadores usados: *SELECT*, *FROM*, *ON*, *JOIN*, *ORDER BY*.

6.6 Meses mais frequentemente correspondentes ao mês com mais urgências do ano

A sexta interrogação lista os meses que mais vezes corresponderam ao mês com mais urgências de um determinado ano.

Operadores usados: *SELECT*, *FROM*, *WHERE*, *CREATE VIEW*, *AS*, *COUNT*, *MAX*, *GROUP BY*, *DROP VIEW*.

6.7 Pacientes que tiveram consulta com o responsável de cada departamento

A sétima interrogação lista os pacientes que tiveram consulta com o médico responsável por cada departamento, ordenados por ordem alfabética de nome e, em caso de empate, por ordem alfabética de departamento.

Operadores usados: *SELECT*, *DISTINCT*, *FROM*, *AS*, *NATURAL JOIN*, *JOIN*, *WHERE*, *AND*, *ORDER BY*.

6.8 Médicos mais velhos de cada departamento

A oitava interrogação lista os médicos mais velhos de cada departamento ordenados por ordem alfabética.

Operadores usados: *SELECT*, *MIN*, *AS*, *FROM*, *INNER JOIN*, *ON*, *GROUP BY*, *ORDER BY*.

6.9 Departamento mais ativo em intervenções em mulheres

A nona interrogação mostra qual o departamento que mais realizou intervenções em mulheres.

Operadores usados: *SELECT*, *AS*, *COUNT*, *FROM*, *INNER JOIN*, *ON*, *WHERE*, *IN*, *UNION*, *GROUP BY*, *ORDER BY DESC*, *LIMIT*.

6.10 Pacientes possivelmente em risco numa epidemia

A décima interrogação lista os pacientes que tiveram uma certa doença e que tiveram uma consulta com um certo médico num certo dia, isto pode ser útil para identificar possíveis casos de contágio de uma doença numa epidemia.

Operadores usados: *SELECT*, *FROM*, *WHERE*, *IN*, *INNER JOIN*, *ON*, *AND*.

7 Triggers

7.1 Validar datas de ocorrências

Antes de uma inserção ou antes de uma atualização de uma ocorrência na base de dados deve ser verificado se a data de início da ocorrência é após o nascimento da pessoa em causa.

Assim este trigger envolve a criação de dois triggers, um que age antes de uma inserção na tabela *Ocorrencia* e outro que age antes de uma atualização na tabela já citada.

No ficheiro para verificar o gatilho (*gatilho1_verifica.sql*) é inserido um paciente novo para ser a cobaia, além de uma doença nova experimental.

Após isso é mostrado a tabela de ocorrências antes de ser feita qualquer inserção ou atualização (para simplicidade é só mostrado ocorrências com ID maior que 450000000, mostrando assim uma tabela vazia). De seguida, é tentada uma inserção na tabela de ocorrências com uma data de início antes do nascimento da cobaia, de modo a verificar que o trigger bloqueia essa ação, é então mostrada a tabela de ocorrências após a tentativa de inserção, podendo verificar-se que mantém-se igual à anterior.

Agora, será feita uma outra tentativa de inserção da mesma ocorrência mas agora com uma data de início após o nascimento da cobaia, verificando-se assim que o trigger não bloqueia a operação e os valores são inseridos, podendo estes ser verificados na tabela que será mostrada após esta segunda tentativa de inserção.

Por último, assumindo que a segunda tentativa de inserção terminou em sucesso ocorre uma tentativa de atualização da data de início da ocorrência inserida, de modo a que a data seja antes do nascimento da cobaia, deve verificar-se que o trigger bloqueará tal operação e não ocorrerá nenhuma modificação, podendo o resultado ser verificado na tabela que será apresentada após esta última tentativa.

7.2 Validar data de fim de internamento a partir de data de evento

Antes de uma inserção ou atualização de um Internamento, deve ser verificado se a data de fim do Internamento é superior (posterior) à data do Evento correspondente.

Esta restrição corresponde portanto à criação de dois gatilhos, um que age antes de uma inserção e outro que age antes de uma atualização, ambos sobre a tabela *Internamento*.

No ficheiro para verificar o gatilho (*gatilho2_verifica.sql*) são inseridos dois Eventos e respetivos Internamentos (EventoID's 2750 e 2751), um para servir de "controlo", cujo Internamento tem data de fim superior à do Evento, e outro para ativar o gatilho, cujo Internamento tem data de fim inferior à do Evento.

Após as inserções, é mostrada a tabela correspondente ao *Natural Join* de Evento e Internamento a partir do EventoID 2741 (na qual apenas foi inserido o primeiro Internamento - 2750 -, enquanto o segundo - 2751 - foi bloqueado).

Para o segundo gatilho, tenta alterar-se a data de fim do Internamento previamente introduzido, cujo valor original era "1989-12-10 00:00:02", para "1989-12-10 00:00:03", a modo de controlo, sendo esta ação permitida. Posteriormente, uma vez que a data do Evento correspondente é "1989-12-10 00:00:01", tenta alterar-se a data de Fim do Internamento para "1989-12-10 00:00:00", ação que é bloqueada pelo gatilho.

Para confirmar estas alterações, é mostrada a tabela a partir do EventoID 2741, na qual aparece a data de fim do Internamento correspondente ao controlo ("1989-12-10 00:00:03"), permitindo a verificação do funcionamento do segundo gatilho.

Nota: Para todas as tabelas apresentadas nestes dois gatilhos, são apresentadas apenas as colunas EventoID, Data e DataFim, para mais fácil visualização dos dados.

7.3 Validar datas de consultas

Antes da inserção ou atualização de uma Consulta é necessário verificar se neste horário o Medico já possui outra Consulta.

Para isso são criados dois gatilhos, um antes da inserção de Consulta e outro na atualização de Evento. Já que Consulta não possui a informação da data, mas sim evento.

Logo, quando se cria uma Consulta nova, verifica-se a data do Evento e o Medico associado à Consulta. Se este Medico já possuir consultas no horário deste evento, a inserção é invalidada.

Já quanto à atualização, verifica-se no Evento a data, e na Consulta que está associada ao Evento o Medico, impedindo da mesma maneira a alteração.

No ficheiro *gatilho3_verifica.sql* primeiro cria-se um Evento com uma data inválida, e ao criar uma Consulta associada a este Evento o erro ocorre. Depois cria-se um Evento com data válida, e não há erro.

Posteriormente, faz-se uma tentativa de atualização de um Evento já associado a uma Consulta, e o valor inserido na data corresponde a uma data que já possui uma outra Consulta com o mesmo Medico. Por fim faz-se uma atualização para uma data disponível.

Entre todas as etapas acima enumeradas há *SELECT statements* mostrando o funcionamento de cada uma.