

Base de Dados - Serralves



Manuel Anselmo Gomes Moreira
Miguel Azevedo Lopes
Sofia Ariana Moutinho Coimbra Germer

up201402718@fe.up.pt
up201704590@fe.up.pt
up201907461@fe.up.pt

Introdução

Para o desenvolvimento do projeto da unidade curricular “Base de Dados”, foi nos permitido escolher um tema sobre o qual deveríamos desenvolver uma base de dados. Desta forma, escolhemos modelar Serralves, por ser um símbolo da cultura do Porto e devido ao nosso interesse em aproveitar a liberdade que nos foi dada para construirmos um projeto com o qual nos identificamos.

DEFINIÇÃO DO MODELO CONCEPTUAL

→ Exposição

A classe Exposição é caracterizada pelo seu ID, nome , descrição, data de início, data de fim e duração (calculada pela diferença entre a data de fim e a data de início).
Data de início, data de fim e duração podem ser NULL caso a exposição seja permanente.
Foi implementada também uma restrição que obriga a data de fim a ser posterior à data de início.

→ Obra

Todas as obras têm características comuns : o seu ID, Nome, Ano e Descrição (apenas o ID é obrigatório dado que existem obras cujo contexto é desconhecido).
A classe Obra é uma generalização completa disjunta de Filmes, Fotografias, Pintura e Escultura, uma vez que uma obra tem de ser obrigatoriamente uma dessas 4 possibilidades.

◆ Filme

Um filme para além das características que possui por ser uma obra é caracterizado pela sua duração, gênero e idioma.
O atributo duração de filme não pode ser NULL visto que este fator é fundamental para a organização e gestão das suas exibições (duração tem de ser superior a 0).

◆ Fotografia

Uma fotografia é caracterizada pela sua altura e largura (valores obrigatoriamente diferentes de 0), bem como um booleano Digital, que é true caso a fotografia seja digital e false caso seja analógica.

◆ Pintura

Uma pintura é caracterizada pela sua altura, largura, estilo artístico e cuidados a ter na sua preservação.

◆ Escultura

Uma escultura é caracterizada pelo seu peso, altura, largura e material em que foi construída.

O peso da escultura é um atributo que não pode ser NULL uma vez que influencia substancialmente as condições de transporte (peso tem de ser superior a 0).

NOTA: Os atributos altura e largura são obrigatórios em fotografia, pintura e escultura, por uma questão de transporte e disposição das peças. Uma vez que os valores de altura e largura são bastante dispareys não consideramos haver uma justificação forte suficiente para criarmos uma classe "Tamanho" (que contemple estes dois atributos).

→ **Artista**

Um artista é caracterizado pelo seu ID, nome , data de nascimento, data de falecimento (que tem valor nulo caso o artista ainda esteja vivo), idade (calculada pela diferença entre a data de falecimento e a data de nascimento, caso o artista tenha falecido, ou entre a data atual e a data de nascimento, caso este seja vivo). Contém ainda os parâmetros nacionalidade e biografia. Todos os parâmetros podem ser NULL com a excepção do ID e do nome (consideramos esta ser a opção mais sensata dado que em obras mais antigas, os detalhes dos artistas nem sempre são conhecidos).

→ **Bilhete**

O bilhete é caracterizado pelo ID, preço, desconto e valor final que é calculado pela multiplicação (Valor Final=Desconto*Preço). Apenas o desconto e valor finais podem ser NULL.

→ **Fatura**

A fatura é caracterizada pelo ID e valor. Nenhum valor pode ser NULL.

→ **Pessoa**

Uma pessoa é constituída por: um NIF, um nome, uma morada, um código postal uma data de nascimento e (idade calculada pela diferença entre a data atual e a data de nascimento). A classe Pessoa é uma generalização incompleta e sobreposta de um Amigo de Serralves e do Staff.

Por um lado, é incompleta uma vez que existem objetos da classe pessoa que nem são Amigos de Serralves nem membros do Staff, no caso da compra ser realizada com fatura de contribuinte e de ser necessário guardar as informações relativas ao cliente.

Por outro lado, é sobreposta uma vez que decidimos admitir a possibilidade de um membro do Staff desejar ser também amigo de Serralves.

É aceitada a possibilidade de alguém visitar Serralves e adquirir um bilhete normal, sem qualquer tipo de desconto nem vontade de incluir dados na fatura e portanto, sem necessidade de revelar qualquer tipo de informação pessoal. Deste modo, essa pessoa não vai ser adicionada à nossa classe Pessoa.

NOTA - É na sequência desta relações entre Bilhete, Fatura , Pessoa (e Staff, o vendedor) que surge a relação enária representada no UML.

- **Amigo**

O amigo de Serralves é um cliente que escolhe pagar uma anuidade para poder visitar o museu sempre que quiser, sem ter de pagar bilhete. Assim, de forma a ser feito um débito direto da conta é adicionado o atributo NIB. O tipo de amigo indica o tipo de benefícios bem como a anualidade associada. Existem as seguintes opções:

- 1 - Normal 50€
- 2 - Família 75€ (permite acesso gratuito à família do amigo)
- 3 - Estudante ou Senior 25€

- **Staff**

As características globais do staff são o NISS (número de Segurança Social) e o salário.

A classe Staff estabelece uma relação de associação com a classe Exposição (Curador do Museu), com os Espaços do Museu (Seguranças) , com as obras (Vigilantes) e com os bilhetes (Vendedor).

Nota: Procurando uma maior flexibilidade na nossa base de dados, consideramos o caso da compra do bilhete ser feita online e portanto não ter um staff a ele associado.

- **HorárioStaff**

Caracterizado pela hora de entrada e saída, sendo que por razões óbvias adicionamos a restrição da hora de saída ter de ser posterior à hora de entrada.

Uma vez que o museu de serralves se encontra aberto durante muitas horas torna- se evidente a existência de vários turnos de trabalho. O horário de um funcionário é determinado ,então, pelo turno que lhe é atribuído, sendo este a access key desta associação qualificada. Sabendo o turno do funcionário, sabemos o seu horário.

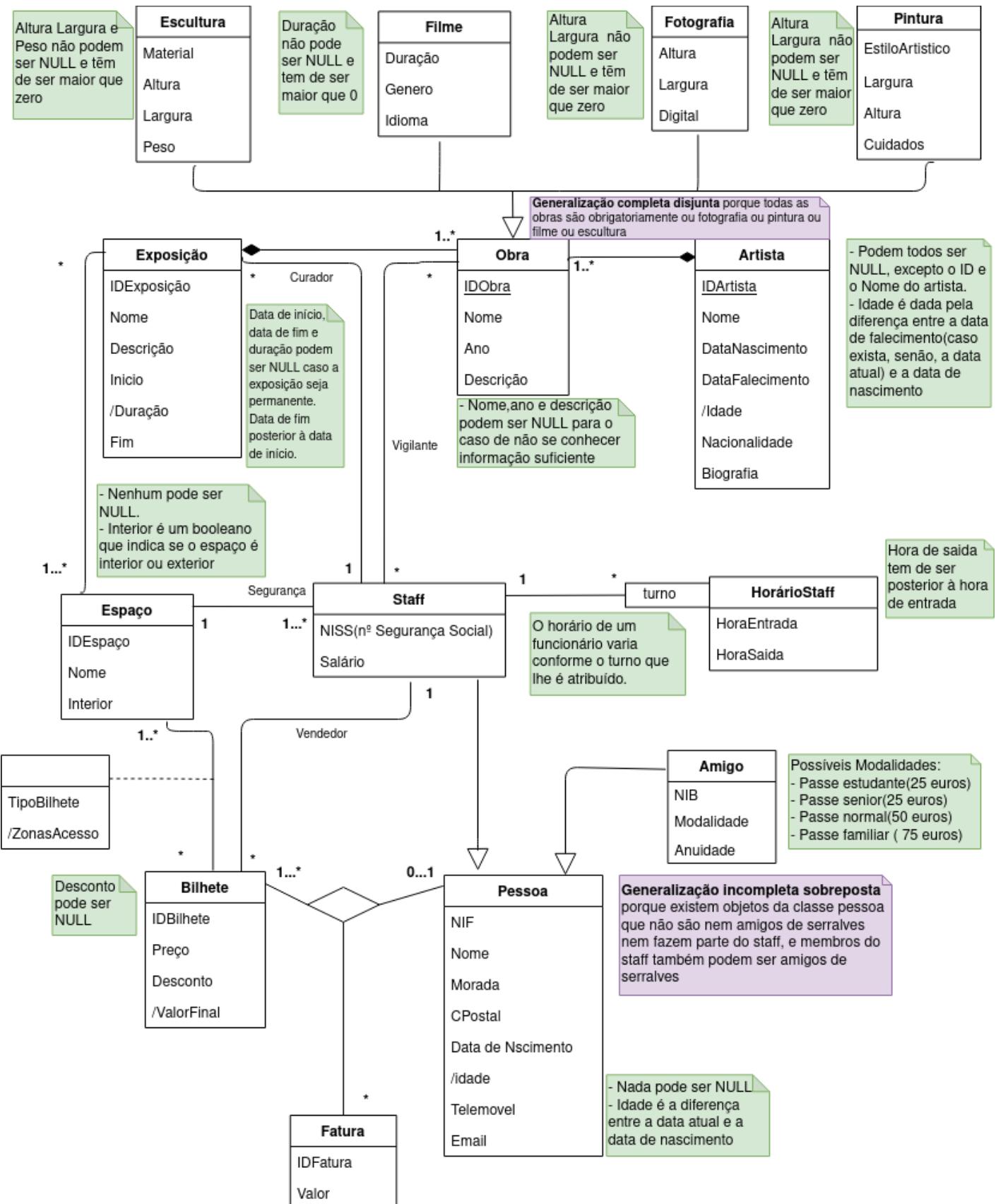
→ **Espaço**

Um espaço é caracterizado pelo seu ID, nome e um booleano "Interior" que nos permite saber se o espaço é interior (true) ou exterior (false).

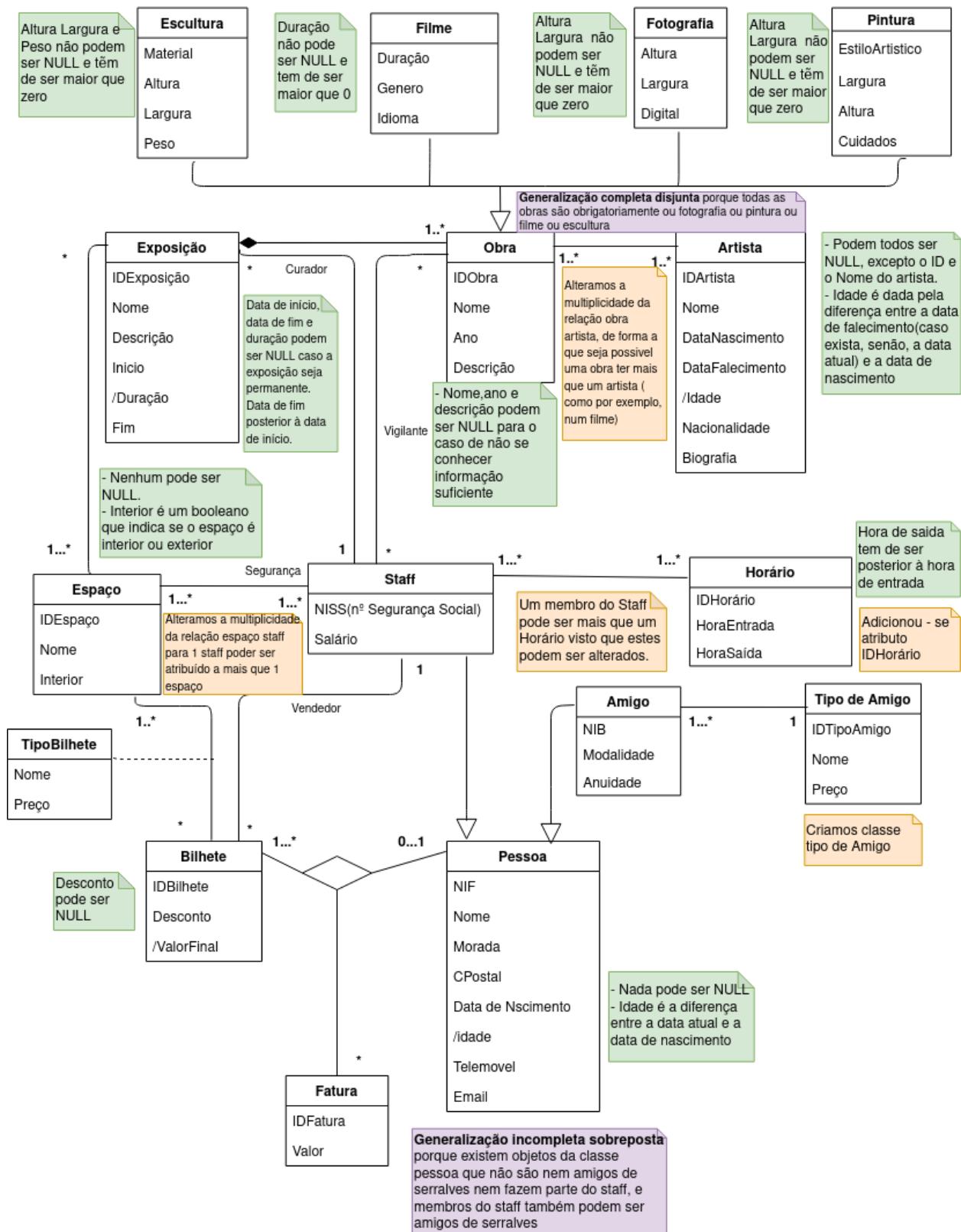
A associação Espaço-exposição permite saber em que Espaço(s) está a decorrer uma determinada exposição (uma exposição tem que estar alocada a pelo menos um espaço, no entanto um espaço não tem de ter uma exposição).

Existe ainda uma classe de associação entre Bilhete e Espaço, com o parâmetro Tipo, que indica o tipo de bilhete e um parâmetro zonas de acesso que mediante o Tipo determina as zonas que se podem aceder, já que um bilhete pode dar acesso a diferentes zonas do Museu (Exemplo: Tipo 1 -> Casa principal + Parque , Tipo 2 -> Casa principal + Casa Manoel Oliveira+ Parque, etc).

UML DA 1ª ENTREGA



UML REVISTO



UML REVISTO - Análise das alterações efetuadas

De forma a facilitar a compreensão das alterações efetuadas no UML, decidimos comentar o UML com umas caixas de comentário laranja com o intuito de evidenciar e explicar as mudanças efetuadas:

1. Alteramos a multiplicidade da relação obra artista para que uma obra possa ser atribuída a mais do que um artista
2. Alteramos a multiplicidade da relação espaço staff, uma vez que nos foi indicado que seria mais sensato possibilitar a atribuição de um Staff a mais do que um espaço
3. Alteramos a relação de Staff e Horário de forma a facilitar a construção da base de dados. Adicionamos , também, um atributo de IDHorário que será a primary key de horário. Assim, agora um membro do staff pode ter mais do que um horário.
4. Perante a sugestão do professor, decidimos criar a classe chamada "Tipo de amigo" com o atributo preço e uma chave que permita identificar o tipo de amigo (IDTipoAmigo). Deste modo, torna - se mais simples a possível situação futura do museu desejar criar novos tipos de amigos.
5. Perante o conhecimento de SQL, concluímos que faria mais sentido na nossa classe de associação entre Bilhete e Espaço armazenar os atributos nome e preço.

B. DEFINIÇÃO DO ESQUEMA RELACIONAL

Os atributos a sublinhado são as **chaves primárias** e os atributos identificados com “->” são as **chaves estrangeiras**.

Pessoa (NIF, Nome, Morada, CPostal, Data de Nascimento, Idade , Telemóvel, Email)

→ Idade é um atributo derivado

Staff (NIF -> Pessoa , NISS , Salário)

Exposição (IDExposição, Nome, Descrição, Início, Duração, Fim, NIF -> Staff)

→ Duração é um atributo derivado (diferença entre a data de fim e a data de início)

Espaço (IDEspaço, Nome, Interior)

Artista (IDArtista, Nome, Data de Nascimento, Data de Falecimento, Idade, Nacionalidade, Biografia)

→ Idade é um atributo derivado

Obra (IDObra, Nome, Ano, Descrição , IDArtista -> Artista, IDExposição -> Exposição)

Filme (IDObra -> Obra , Duração, Género, Idioma)

Fotografia (IDObra -> Obra, Altura, Largura, Digital)

Pintura (IDObra -> Obra, Estilo Artístico, Largura, Altura, Cuidados)

Escultura (IDObra -> Obra, Material, Altura, Largura, Peso)

Bilhete (IDBilhete, Desconto, Valor Final, NIF -> Staff)

→ Valor Flnal é um atributo derivado (diferença entre Preço e Desconto)

Fatura (IDFatura, Valor, NIF->Staff)

Compra (IDFatura -> Fatura, IDBilhete -> Bilhete, NIF -> Pessoa)

Amigo (NIF -> Pessoa, NIB, Modalidade, Anuidade, IDTipoAmigo -> TipoAmigo)

TipoAmigo(IDTipoAmigo , Nome, Preço)

Vigilante (NIF -> Staff, IDObra -> Obra)

Segurança (NIF -> Staff, IDEspaço -> Espaço)

Horário (IDHorário , HoraEntrada, HoraSaída)

TipoBilhete (IDEspaço -> Espaço, IDBilhete -> Bilhete , Nome, Preço)

ExposiçãoEspaço (IDExposição -> Exposição , IDEspaço -> Espaço)

ObraArtista (IDObra -> Obra, IDArtista -> Artista)

HorárioStaff (IDHorário -> Horário, NIF -> Staff)

C. ANÁLISE DE DEPENDÊNCIAS FUNCIONAIS E FORMAS NORMAIS

Para uma tabela estar na forma BCNF tem de estar obrigatoriamente na 3^aForma Normal, na 2^aForma Normal e , consequentemente, também na 1^a Forma Normal.

Assim, sempre que na determinação da forma referimos que se encontra na forma BCNF, optamos por omitir que se encontra na 3^a Forma Normal e consequentemente na 2^a e 1^a forma também.

Pessoa (NIF, Nome, Morada, CPostal, Data de Nascimento, Idade , Telemóvel, Email)

→ FDs

NIF -> Nome, Morada, CPostal, Data de Nascimento, Idade , Telemóvel, Email

Telemóvel -> NIF, Nome, Morada, CPostal, Data de Nascimento, Idade, Email

Email -> NIF, Nome, Morada, Postal , Data de Nascimento, Idade, Telemóvel

→ Forma: BCNF

Analizando as closures:

"{Telemóvel}⁺ = {Telemóvel, NIF, Morada, CPostal, Data de Nascimento, Idade, Email}"

"{Email}⁺ = {Email, NIF, Nome, Morada, CPostal, Data de Nascimento, Idade, Telemóvel}

Tanto na segunda como na terceira dependência funcional estão incluídos todos os atributos da tabela, pelo que concluímos que a BCNF não é violada.

Staff (NIF -> Pessoa , NISS , Salário)

→ FDs

NIF -> NISS , Salário

NISS -> NIF, Salário

→ Forma: BCNF

Analizando a closure da segunda dependência funcional:

"{NISS}⁺ = {NIF, Saláriol}"

Estão incluídos todos os atributos da tabela, pelo que concluímos que a BCNF não é violada.

Exposição (IDExpo, Nome, Descrição, Início, Duração, Fim, NIF -> Staff)

→ FDs

IDExpo -> Nome, Descrição, Início, Duração, Fim, NIF

→ Forma: BCNF

Espaço (IDEspaço, Nome, Interior)

→ FDs

IDEspaço -> Nome, Interior

Nome -> IDEspaço, Interior

→ Forma: BCNF

Analizando a closure da segunda dependência funcional:

$\{\text{Nome}\}^+ = \{\text{IDEspaço}, \text{Interior}\}$

Estão incluídos todos os atributos da tabela, pelo que concluímos que a BCNF não é violada.

Artista (IDArtista, Nome, Data de Nascimento, Data de Falecimento, Idade, Nacionalidade, Biografia)

→ FDs

IDArtista → Nome, Data de Nascimento, Data de Falecimento, Idade, Nacionalidade, Biografia

→ Forma: BCNF

Obra (IDObra, Nome, Ano, Descrição, IDArtista → Artista, IDExposição → Exposição)

→ FDs

IDObra → Nome, Ano, Descrição, IDArtista, IDExposição

→ Forma: BCNF

Filme (IDObra → Obra, Duração, Género, Idioma)

→ FDs

IDObra → Duração, Género, Idioma

→ Forma: BCNF

Fotografia (IDObra → Obra, Altura, Largura, Digital)

→ FDs

IDObra → Altura, Largura, Digital

→ Forma: BCNF

Pintura (IDObra → Obra, Estilo Artístico, Largura, Altura, Cuidados)

→ FDs

IDObra → Estilo Artístico, Largura, Altura, Cuidados

→ Forma: BCNF

Escultura (IDObra → Obra, Material, Altura, Largura, Peso)

→ FDs

IDObra → Material, Altura, Largura, Peso

→ Forma: BCNF

Bilhete (IDBilhete, Preço, Desconto, Valor Final, NIF → Staff)

→ FDs

IDBilhete → Preço, Desconto, Valor Final, NIF

→ Forma: BCNF

Fatura (IDFatura, Valor)

- FDs
 - IDFatura -> Valor

→ Forma: BCNF

Compra (IDFatura -> Fatura, IDBilhete -> Bilhete, NIF -> Pessoa)

- FDs
 - Sem dependências funcionais não triviais

→ Forma: BCNF

Amigo (NIF -> Pessoa, NIB, IDTipoAmigo -> TipoAmigo)

- FDs
 - NIF -> NIB, IDTipoAmigo

→ Forma: BCNF

TipoAmigo (IDTipoAmigo , Nome, Preço)

- FDs
 - IDTipoAmigo -> Nome, Preço
 - Nome -> IDTipoAmigo, Preço

→ Forma: BCNF

Analizando a closure da segunda dependência funcional:

" $\{Nome\}^+ = \{IDTipoAmigo, Preço\}$ "

Estão incluídos todos os atributos da tabela, pelo que concluímos que a BCNF não é violada.

Vigilante (NIF -> Staff, IDObra -> Obra)

- FDs
 - Sem dependências funcionais não triviais

→ Forma: BCNF

Segurança (NIF -> Staff, IDEspaço -> Espaço)

- FDs
 - Sem dependências funcionais não triviais

→ Forma: BCNF

Horário (IDHorário , HoraEntrada, HoraSaída)

- FDs
 - IDHorário -> HoraEntrada, HoraSaída

→ Forma: BCNF

TipoBilhete (IDEspaço -> Espaço, IDBilhete -> Bilhete , Nome, Preço)

- FDs
 - IDEspaço, IDBilhete -> Tipo, Nome, Preço

→ Forma: BCNF

ExposiçãoEspaço (IDExposição -> Exposição , IDEspaço -> Espaço)

- FDs
Sem dependências funcionais não triviais
- Forma: BCNF

ObraArtista (IDObra -> Obra, IDArtista -> Artista)

- FDs
Sem dependências funcionais não triviais
- Forma: BCNF

HorárioStaff (IDHorário -> Horário, NIF -> Staff)

- FDs
Sem dependências funcionais não triviais
- Forma: BCNF

NOTA (Atributos Derivados) :

Um caso particular de violação à 3NF/BCNF com que nos deparamos advém de dependências funcionais que contêm atributos derivados. Uma vez que, os atributos derivados mais tarde serão implementados com uma VIEW, estes atributos deixarão de estar na sua tabela original e passarão a estar somente na VIEW. Como consequência disto, as dependências funcionais da tabela original que incluíam este atributo, vão deixar de existir. Assim, concluímos que não faz sentido estar a decompor uma relação com base numa dependência funcional que vai deixar de existir.

CONCLUSÃO:

O critério para que uma relação R esteja conforme Boyce-Codd Normal Form (BCNF) é que para todas as suas dependências funcionais não triviais $A \rightarrow B$, A seja uma superkey, ou seja que A^+ contenha todos os atributos de R.

Para todas as relações e todas as FDs da nossa base de dados, a partir de A de cada FD conseguimos conhecer todos os atributos da relação, o que implica que A é a (super) key.

Assim podemos concluir que todas as relações do nosso modelo relacional estão em BCNF e 3NF.

E. ADIÇÃO DE RESTRIÇÕES À BASE DE DADOS

Nota : Sempre que um atributo é uma Primary Key, sabemos que esse atributo não pode ser NULL.

Pessoa

- Não podem haver duas pessoas com o mesmo NIF e este tem que ter 9 dígitos
 - ◆ NIF PRIMARY KEY
 - ◆ CONSTRAINT CHECK (length(NIF) == 9)
- Todas as pessoas devem ter um nome, morada, código postal, data de nascimento, telemóvel e email
 - ◆ Nome NOT NULL
 - ◆ Morada NOT NULL
 - ◆ CPostal NOT NULL
 - ◆ Data de Nascimento NOT NULL
 - ◆ Telemóvel NOT NULL
 - ◆ Email NOT NULL
- Telemóvel, Email e Código Postal tem restrições no formato que podem assumir (Telemovel tem 9 digitos, Email tem o formato caracteres@caracteres.extensão e Código Postal tem de ser um conjunto de quatro caracteres, seguido por um hífen e por outros três caracteres)
 - ◆ CONSTRAINT CHECK (length(Telemovel) == 9)
 - ◆ CONSTRAINT CHECK (Email LIKE '%@%.%_')
 - ◆ CONSTRAINT CHECK (CPostal LIKE '___-__')
- Não podem haver duas pessoas com o mesmo Telemóvel, nem com o mesmo Email
 - ◆ Telemóvel UNIQUE
 - ◆ Email UNIQUE

Staff

- Não podem haver dois membros do Staff com o mesmo ID. Este ID deverá corresponder a um ID de uma Pessoa na tabela Pessoa
 - ◆ NIF PRIMARY KEY REFERENCES Pessoa (NIF)
- Não podem existir membros do Staff com o mesmo NISS e tem que ter 11 dígitos
 - ◆ NISS UNIQUE
 - ◆ CONSTRAINT CHECK (length(NISS) == 11)
- Todos os membros do Staff devem ter salário e NISS definidos
 - ◆ Salário NOT NULL
 - ◆ NISS NOT NULL

Exposição

- Não podem haver duas exposições com o mesmo ID

- ◆ IDExposição PRIMARY KEY
- Todas as exposições devem ter nome e descrição
 - ◆ Nome NOT NULL
 - ◆ Descrição NOT NULL
 - ◆ Início NOT NULL
- A exposição tem duas opções: ou é permanente (não tem data de fim) ou é finita (data de fim superior à data de início).
 - ◆ ExposicaoCheckDatas CHECK (Fim > Início OR Fim IS NULL)

Espaço

- Não podem haver dois espaços com o mesmo IDEspaço
 - ◆ IDEspaço PRIMARY KEY
- Todos os espaços devem ter nome
 - ◆ Nome NOT NULL
- Não podem haver dois espaços com o mesmo Nome
 - ◆ Nome UNIQUE

Artista

- Não podem haver dois artistas com o mesmo ID
 - ◆ IDArtista PRIMARY KEY
- Cada artista tem de ter um nome
 - ◆ Nome NOT NULL
- Caso o artista tenha falecido, a data de falecimento tem de ser posterior à data de nascimento.
 - ◆ DataFalecimento DATE CONSTRAINT CHECK (DataFalecimento > DataNascimento)
OR DataFalecimento IS NULL)

Obra

- Não podem existir duas obras com o mesmo ID
 - ◆ IDObra PRIMARY KEY
- Contém uma foreign key que faz referência à tabela exposição, dado que cada obra pertence a uma exposição
 - ◆ IDExposição REFERENCES Exposicao(IDExposicao)

Filme

- Não podem existir dois filmes com o mesmo ID. Este ID deverá corresponder a um ID de uma obra na tabela Obra.
 - ◆ IDObra PRIMARY KEY REFERENCES Obra (IDObra)
- Cada filme deve ter obrigatoriamente uma duração e tem que ser maior que 0
 - ◆ Duração NOT NULL
 - ◆ CONSTRAINT CHECK (Duracao > 0)

Fotografia

- Não podem existir duas fotografias com o mesmo ID. Este ID deverá corresponder a um ID de uma obra na tabela Obra.
 - ◆ IDObra PRIMARY KEY REFERENCES Obra (IDObra)
- Cada fotografia deve ter obrigatoriamente altura e largura
 - ◆ Altura NOT NULL
 - ◆ Largura NOT NULL
- Altura tem de ser maior que 0
 - ◆ Altura CHECK (Altura > 0)
- Largura tem de ser maior que 0
 - ◆ Largura CHECK (Largura > 0)

Pintura

- Não podem existir duas pinturas com o mesmo ID. Este ID deverá corresponder a um ID de uma obra na tabela Obra.
 - ◆ IDObra PRIMARY KEY REFERENCES Obra (IDObra)
- Cada pintura deve ter obrigatoriamente altura e largura
 - ◆ Altura NOT NULL
 - ◆ Largura NOT NULL
- Altura tem de ser maior que 0
 - ◆ Altura CHECK (Altura > 0)
- Largura tem de ser maior que 0
 - ◆ Largura CHECK (Largura > 0)

Escultura

- Não podem existir duas esculturas com o mesmo ID. Este ID deverá corresponder a um ID de uma obra na tabela Obra.
 - ◆ IDObra PRIMARY KEY REFERENCES Obra (IDObra)
- Cada escultura deve ter obrigatoriamente altura, largura e peso
 - ◆ Altura NOT NULL
 - ◆ Largura NOT NULL
 - ◆ Peso NOT NULL
- Altura tem de ser maior que 0
 - ◆ Altura CHECK (Altura > 0)
- Largura tem de ser maior que 0
 - ◆ Largura CHECK (Largura > 0)
- Peso tem de ser maior que 0
 - ◆ Peso CHECK (Peso > 0)

Bilhete

- Não podem haver dois bilhetes com o mesmo ID
 - ◆ IDBilhete PRIMARY KEY
- Desconto tem que ser maior que 0 (0%) e menor ou igual a 1 (100%) ou NULL
 - ◆ Desconto CONSTRAINT CHECK ((Desconto>0.0 AND Desconto<=1.0) OR Desconto IS NULL)

Fatura

- Não podem haver duas faturas com o mesmo ID
 - ◆ IDFatura PRIMARY KEY
- Valor não pode ser nulo
 - ◆ Valor NOT NULL
- NIF é foreign key faz referência à tabela Staff, representa o vendedor que emitiu a fatura
 - ◆ IDStaff REFERENCES Staff (NIF)

Compra

- Não podem haver duas instâncias com o mesmo par (IDFatura, IDBilhete) : chave primária composta
 - ◆ PRIMARY KEY (IDFatura, IDBilhete)
- NIF é foreign key e faz referência à tabela Pessoa, dado que numa compra o cliente pode querer incluir os seus dados, como o Nome, NIF, Morada, etc.
 - ◆ NIF REFERENCES Pessoa(NIF)

Amigo

- Não podem haver dois amigos com o mesmo NIF; este NIF deve corresponder a um NIF de uma pessoa na tabela Pessoa
 - ◆ NIF PRIMARY KEY REFERENCES Pessoa (NIF)
- Não podem haver dois amigos com o mesmo NIB
 - ◆ NIB UNIQUE
- IDTipoAmigo faz referência à tabela TipoAmigo, dado que o Amigo pode ser do tipo Senior, Estudante, Normal, Família ou Professor.
 - ◆ IDTipoAmigo REFERENCES TipoAmigo(IDTipoAmigo)

TipoAmigo

- Não podem haver dois tipos de amigos com o mesmo ID
 - ◆ IDTipoAmigo PRIMARY KEY
- Todas os tipos de amigos devem ter a sua anuidade especificada, bem como o seu nome
 - ◆ Nome NOT NULL
 - ◆ Preço NOT NULL
- Não podem haver dois tipos de amigo com o mesmo nome
 - ◆ Nome UNIQUE

- Preço tem que ser maior que zero
 - ◆ Preço CHECK (Preço > 0)

Vigilante

- Não podem haver duas instâncias com o mesmo par (NIF, IDObra) : chave primária composta
 - ◆ PRIMARY KEY (NIF, IDObra)
- O NIF deve corresponder a um NIF da tabela Staff e o ID de Obra deve corresponder a um ID da tabela Obra
 - ◆ NIF REFERENCES Staff (NIF)
 - ◆ IDObra REFERENCES Obra (IDObra)

Segurança

- Não podem haver duas instâncias com o mesmo par (NIF, IDEspaço) : chave primária composta
 - ◆ PRIMARY KEY (NIF, IDEspaço)
- O NIF deve corresponder a um NIF da tabela Staff e o ID de Espaço deve corresponder a um ID da tabela Espaço
 - ◆ NIF REFERENCES Staff (NIF)
 - ◆ IDEspaço REFERENCES Espaço (IDEspaço)

Horário

- Não podem existir dois horários com o mesmo ID
 - ◆ PRIMARY KEY (IDHorário)
- Todos os Horários devem ter hora de entrada e hora de saída
 - ◆ HoraEntrada NOT NULL
 - ◆ HoraSaída NOT NULL
- HoraSaída tem de ser posterior à HoraEntrada
 - ◆ CHECK (HoraSaída > HoraEntrada)

TipoBilhete

- Não podem haver duas instâncias com o mesmo par (IDEspaco, IDBilhete) : chave primária composta
 - ◆ PRIMARY KEY (IDEspaco, IDBilhete)
- O IDEspaco deve corresponder a um ID da tabela Espaco e o IDBilhete deve corresponder a um ID da tabela Bilhete
 - ◆ IDEspaco REFERENCES Espaco (IDEspaco)
 - ◆ IDBilhete REFERENCES Bilhete (IDBilhete)
- Cada tipo de bilhete tem de ter um preço
 - ◆ Preço NOT NULL
- Nome do tipo de bilhete tem de existir
 - ◆ Nome NOT NULL

ExposicaoEspaco

- Não podem haver duas instâncias com o mesmo par (IDExposicao, IDEspaco) : chave primária composta
 - ◆ PRIMARY KEY (IDExposicao, IDEspaco)
- O ID de Exposição deve corresponder a um ID da tabela exposição e o ID de Espaço deve corresponder a um ID da tabela Espaco
 - ◆ IDEspaco REFERENCES Espaco (IDEspaco)
 - ◆ IDE exposição REFERENCES Exposição (IDExposição)

ObraArtista

- Não podem haver duas instâncias com o mesmo par (IDObra, IDArtista) : chave primária composta
 - ◆ PRIMARY KEY (IDObra, IDArtista)
- O ID de Obra deve corresponder a um ID da Obra e o ID de Artista deve corresponder a um ID da tabela Artista
 - ◆ IDObra REFERENCES Obra (IDObra)
 - ◆ IDArtista REFERENCES Artista (IDArtista)

HorarioStaff

- Não podem haver duas instâncias com o mesmo par (NIF, IDHorário) : chave primária composta
 - ◆ PRIMARY KEY (NIF, IDHorário)
- O NIF deve corresponder a um NIF da tabela Staff e o ID de Horário deve corresponder a um ID da tabela Horário
 - ◆ NIF REFERENCES Staff (NIF)
 - ◆ IDHorário REFERENCES Horário (IDHorário)

Adicionalmente, a todas as chaves estrangeiras foi adicionada a constraint ON DELETE CASCADE ON UPDATE CASCADE, pois em caso de remoção ou alteração da chave, esta deve -se refletir nos sítios onde é referida.

É também relevante mencionar que demos nome a todas as restrições para facilitar na leitura/interpretação e para ajudar na detecção de possíveis erros que surgissem na povoação.

F. CARREGAMENTO DE DADOS

Para criar o ficheiro povoar.sql recorremos à povoação manual para inserir dados como os das Pessoas, Amigos, Staff, Horários. Adicionalmente, para tentar povoar a base de dados com dados realistas, desenvolvemos scripts para a povoação dos Artistas e Obras e as relações entre estes, recorrendo aos dados fornecidos pelo site do próprio [museu de Serralves](#). O mesmo foi feito para a geração de dados de Bilhetes e Faturas.

G. INTERROGAÇÃO DA BASE DE DADOS

1. Número de clientes normais e número clientes "Amigo", discriminados por "tipo de Amigo"
2. Quais os espaços mais visitados (com mais bilhetes vendidos)?
3. Qual o salário e horas semanais trabalhadas por cada membro do Staff?
4. Top 5 Artistas com obras mais expostas em Serralves
5. Quais são os horários de trabalho? E quantos trabalhadores estão associados a cada um desses horários?
6. Número de obras de cada tipo (Fotografia /Pintura/ Escultura / Filme)
7. Qual a percentagem de obras vigiadas ?
8. Quais as exposições que estão a decorrer atualmente bem como o nome do Curador dessa exposição
9. Listagem de todos os trabalhadores de Serralves incluindo o valor idade (calculado a partir da data de nascimento) e se são maiores de idade.
10. Quantos bilhetes é que cada Funcionário de Serralves vendeu

Notas:

1. Foram efetuadas umas ligeiras alterações aos dados que inserimos na entrega anterior (no ficheiro povoar.sql), de forma a procurar tornar mais explícita a funcionalidade de cada querie. Adicionamos, por exemplo, uma maior variedade de horários, de forma a ser visível o cálculo de horas semanais.
2. Tal como foi mencionado previamente na seção C (Análise de dependências funcionais e formas normais) recorremos a uma VIEW na interrogação 9 para efetuar o cálculo da idade.

H. ADIÇÃO DE GATILHOS À BASE DE DADOS

1. Ao adicionar um bilhete a uma compra, alterar o valor final da Fatura usando o valor do desconto
2. Se o horário de um trabalhador for aumentado, o seu salário também é aumentado proporcionalmente.
3. Assegura que nenhum funcionário pode ser em simultâneo Segurança e Vigilante.

Instruções de teste:

Para testar os gatilhos, o utilizador deve ler o ficheiro de criação da base de dados, "criar.sql", de seguida povoar a base de dados com o ficheiro "povoar.sql" e só depois ler os ficheiros para adicionar, "trigger1_add.sql", e testar o trigger, "trigger1_check.sql".

Conclusão

Desde o início do projeto esforçamo-nos para assegurar que a nossa base de dados se aproximava o máximo possível a uma usada no mundo real.

Para isso, aprofundamos bastante alguns conceitos teóricos sendo que o nosso projeto inclui a realização de um UML, a definição de um esquema relacional, a análise respetiva das suas dependências funcionais e formas normais e a indicação das restrições impostas à base de dados. Por último, numa vertente mais prática, recorrendo ao SQLITE criamos a base de dados e efetuamos a sua povoação, sendo que para a última entrega adicionamos a criação de interrogações e de gatilhos.