BDAD 17/18

Entrega 3

Trabalho realizado por:

João Alves, up201605236
Mariana Costa, up201604414
Tiago Castro, up201606186

Índice

1. Descrição do projeto

O presente trabalho pretende esquematizar a implementação de uma base de dados numa editora musical ("record label").

Em primeiro lugar, consideramos a classe **Pessoa**, da qual derivam as classes **Artista** e **Produtor**. Sobre cada **pessoa** é necessário preservar o nome, a data de nascimento e a nacionalidade.

De cada **artista**, é necessário preservar o "stage name" e o conjunto de **álbuns** que produziu até ao momento.

Relativamente aos **álbuns**, pretende-se guardar o título, data de lançamento, estilo (que pode estar associado ou não, visto que há albuns que não se enquadram em nenhum), **campanhas publicitárias** às quais esteve associado, o **número de vendas mensais** e as **sessões de gravação** efetuadas. A um mesmo álbum pode estar associado um ou vários **artistas**, mas apenas um **produtor**.

Cada **sessão de gravação** tem uma data, horas de início e fim e um estúdio no qual é efetuada. Do **estúdio de gravação**, é preciso saber a renda e morada. De notar que, numa **sessão de gravação**, não é necessário que todos os **artistas** responsáveis pelo **álbum** estejam presentes. É também relevante mencionar que consideramos pertinente manter um registo das **sessões de gravação** em que um **artista** participou, pelo que foi implementada a associação entre as duas classes.

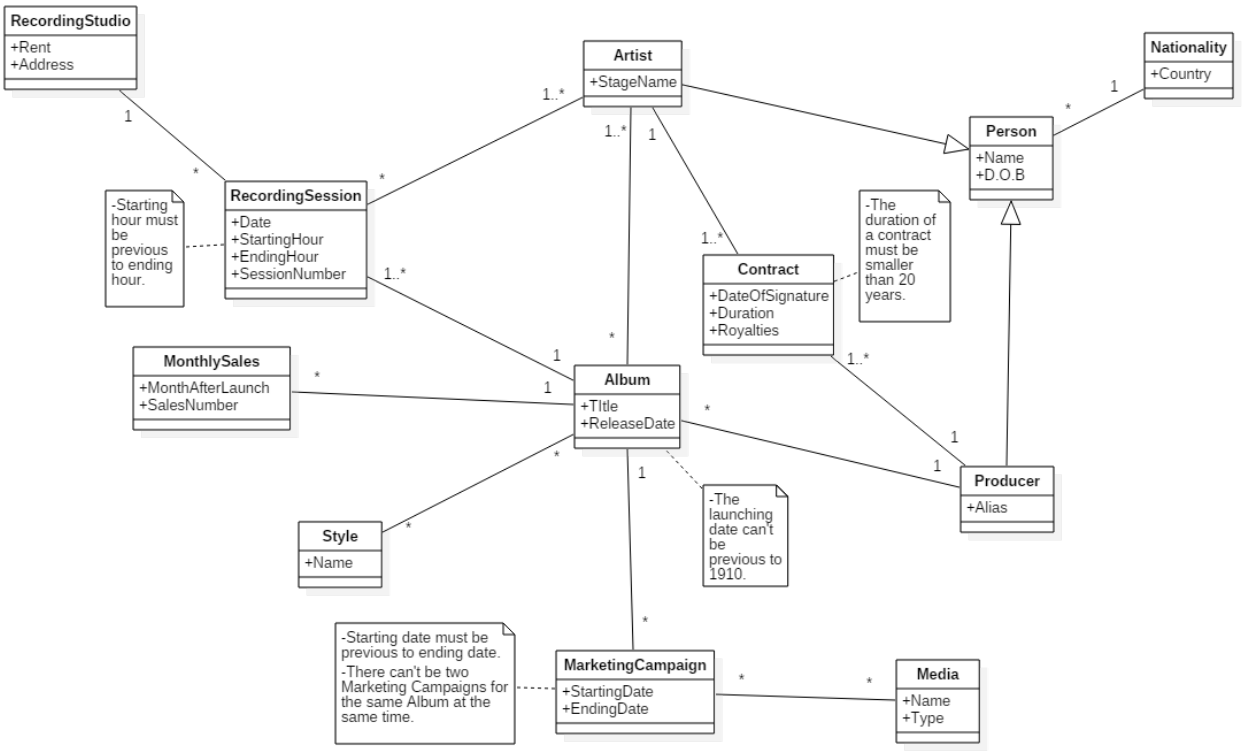
Cada **registo de vendas mensal** contém o mês (após o lançamento) do álbum a que se refere e o número efetivo de vendas. Não há uma ligação direta ao **artista**, uma vez que consideramos que o seu vencimento é determinado pela editora e não provém diretamente das vendas dos álbuns pelos quais é responsável.

Uma **campanha publicitária** é marcada pelas datas de início e fim, bem como as **plataformas** a que recorre. Essas **plataformas** têm um nome e um tipo (rádio, tv, digital, ...) a elas associado. A campanha pode não utilizar um meio em particular (por exemplo, publicidade através de posters).

De um **produtor**, procura-se guardar o seu nome, data de nascimento, **nacionalidade** e **álbuns** aos quais está associado.

Por fim, tanto os **artistas** como os **produtores** mantêm **contratos** com a editora em questão. Destes contratos, são guardados a data de assinatura, a duração e percentagem de lucro à qual o **artista** ou **produtor** em questão tem direito. Consideramos que um **contrato** reflete o desejo de cooperação entre um **artista**/**produtor** e a empresa, que se estende para lá do escopo de um projeto musical, pelo que não existe nenhuma associação direta a um **álbum**.

2. Diagrama UML



3. Diagrama relacional e dependências funcionais

Person(<u>id</u>, dateOfBirth, name, nationality→Nationality)

- {id} → {dateOfBirth, name, nationality};

Artist(<u>id</u>→Person, stageName);

- {id→Person} → {stageName};

Producer(<u>id</u>→Person, alias);

- {id→Person } → {alias};

Nationality(<u>id</u>, countryName);

- {id} → {countryName};
- {countryName} → {id};

Album(<u>id</u>, title, releaseDate, producer→Producer);

- {id} → {title, releaseDate, producer};

Contract(<u>id</u>, dateOfSignature, duration, royalties, producer→Producer, artist→Artist);

- {id} → {dateOfSignature, duration, royalties, producer, artist};

RecordingSession(<u>id</u>, date, startingHour, endingHour, studio→Studio, album→Album);

- {id} → {date, startingHour, endingHour, studio, album};

RecordingStudio(<u>id</u>, address, rent);

- {id} → {address, rent};
- {address} → {id};

Style(<u>id</u>, name);

- {id} → {name};
- {name} → {id};

MonthlySales(<u>id</u>, monthsAfterLaunch, salesNumber, album→Album);

- {id} → {monthsAfterLaunch, salesNumber, album};
- {monthsAfterLaunch, album} → {salesNumber};

MarketingCampaign(<u>id</u>, startingDate, endingDate, album→Album);

- {id} → {startingDate, endingDate, album};
- {startingDate, endingDate, album} → {id};

Media(id, name, type);

- {id} → {name, type};

ArtistInSession(artist → Artist, session → Session);

AlbumStyle(style → Style, album → Album);

MediaInCampaign(campaign → MarketingCampaign, media → Media);

ArtistInAlbum(artist → Artist, album → Album);

4. Formas normais

De forma a que as presentes relações se encontrem na Forma Normal *Boyce-Codd*, é necessário que estas cumpram o seguinte critério:

- Para todas as dependências funcionais não-triviais $\overline{A} \rightarrow \overline{B}$, $\overline{A}$ é uma (*super*)*key;*

Sendo que a Terceira Forma Normal contêm a Forma Normal Boyce-Codd, as relações estarão na Terceira Forna Normal caso cumpram o critério acima descrito, ou se:

- Para todas as dependências funcionais não-triviais $\overline{A} \rightarrow \overline{B}$, $\overline{B}$ consiste apenas em atributos primos*;*

Assim, o modelo proposto cumpre, na sua maioria, estas normas, uma vez que o lado esquerdo de cada dependência é uma *super-key* do esquema relacional – condição suficiente para satisfazer ambas as normas. Caso o lado esquerdo não seja uma *super-key*, se o lado direito for composto por atributos primos, encontra-se apenas na Terceira Forma Normal. Contudo, é necessário mencionar as exceções a estas regras, como por exemplo na relação *MonthlySales*, na qual, a partir do mês após o lançamento e o álbum, é possível chegar ao número de vendas.

5. Restrições

Para preservar a integridade dos dados armazenados e a conformidade do projeto com as especificidades do enunciado, recorreu-se ao uso de restrições aquando do planeamento da transição do modelo relacional para a concretização de uma base de dados. Desta forma, passamos a listar as principais restrições utilizadas e os casos mais relevantes da sua utilização.

5.1 NOT NULL

A restrição *NOT NULL* garante que a existência de um determinado atributo é uma condição necessária para a criação da respetiva classe. Dada a sua vasta utilização, optamos por apenas exemplificar alguns exemplos mais relevantes:

- Os atributos *royalties* e *duration* são o cerne de um contrato realizado neste contexto, pelo que não poderão, de forma alguma, ser ignorados; pelo contrário, o atributo *dateOfSignature* não possui um peso tão significativo na sua interpretação;
- O atributo *address* da classe *RecordingStudio* é obrigatório, uma vez que se trata de um espaço físico fixo; contudo, o atributo *rent* não o é, uma vez que, por exemplo, o estúdio poderá pertencer à *record label*.

5.2 UNIQUE

A restrição *UNIQUE* é necessária para atributos que, embora não sejam chaves, são identificadores da classe. A título de exemplo, listamos alguns desses casos:

- Não podem existir dois género musicais com o mesmo nome, pelo que o atributo *name* da classe é único;
- Não podem existir dois países com o mesmo nome, pelo que o atributo *name* da classe é único.
- A partir de um álbum e do mês após o seu lançamento, é possível chegar ao número de vendas.

5.3 CHECK

A restrição *CHECK* trata-se, possivelmente, da restrição cuja implementação suscita explicações mais interessantes que as anteriores, uma vez que procura dar respostas a problemas específicos ao tema em estudo. Como tal, passamos a descrever algumas implementações mais relevantes:

- A data de lançamento de um álbum não pode ser anterior a 1910;
- A duração de um contrato não pode ser superior a 20 anos;
- Um contrato é estabelecido entre a empresa e um artista, produtor ou ambos.

5.4 Integridade referencial

Relativamente a restrições de integridade referencial, optamos por aplicar chaves estrangeiras a classes que partilham uma ligação direta. Tomemos como exemplo a classe *MarketingCampaign* e *MonthlySales*: ambas estão necessariamente associadas a um álbum, uma vez que não existiriam sem ele. Assim, este mesmo princípio foi aplicado às restantes classes, consoante a necessidade.

6. Interrogações

Segue-se a lista de interrogações consideradas relevantes no contexto do problema analisado, que comprovam o correto funcionamento da base de dados.

- Todos os álbuns de um artista;
- Contratos realizados entre duas datas;
- Artistas que realizaram gravações em todos os estúdios;
- Segundo artista com o maior número de vendas;
- Número de vendas totais de um mês para um determinado estilo musical;
- Álbum mais vendido de um determinado artista;
- Número de horas de gravação de todos os álbuns;
- Artistas sem gravações há mais de dois anos: verificação das associações entre as Artist, ArtistInSession e RecordingSession;
- Número de álbuns gravados por artistas de uma determinada nacionalidade: verificação das associações entre Nationality, Artist, Person e ArtistInAlbum;
- Duração restante de todos os contratos de um artista.

7. Triggers

Foram definidos 3 gatilhos com o propósito de monitorizar e garantir a manutenção da base de dados aquando da realização de certas operações:

- AdicionaContract: Antes de ser introduzido um novo contrato, o sistema verifica se este é realizado apenas entre a empresa e um artista ou um produtor, garantido que os dois não coexistem num mesmo contrato;
- VerificaMinArtists: Quando um artista é removido da base de dados, o sistema verifica se algum álbum ou sessão de gravação tem como único participante esse mesmo artista; se isso se verificar, os respetivos álbuns ou sessões de gravação são removidos;
- ValidateRecordingSession: Antes de atualizar a informação sobre uma sessão de gravação, o sistema verifica se a data na qual se realizou é anterior ao lançamento desse mesmo álbum.