2020/21

BD – Relatório

PROJETO FINAL

BASE DE DADOS SOBRE FILMES

Filipe Gonçalves	98083
Tiago Matos	98134

ANÁLISE DE REQUISITOS

Para este projeto, pretendemos criar uma base de dados para filmes, assim como uma interface que permita a gestão, queremos poder adicionar e editar filmes e os seus componentes.

Para proceder ao desenho conceptual, ponderámos e pesquisámos um pouco sobre o tópico e percebemos algumas generalidades a ter em conta no nosso projeto.

Para começar, um **filme tem** informações que geralmente queremos guardar como:

- Título
- Ano
- Género

Para um utilizador também será interessante ter acesso a informação como:

- Duração
- Ranking
- Lucros
- Público Alvo
- Locais de Gravação

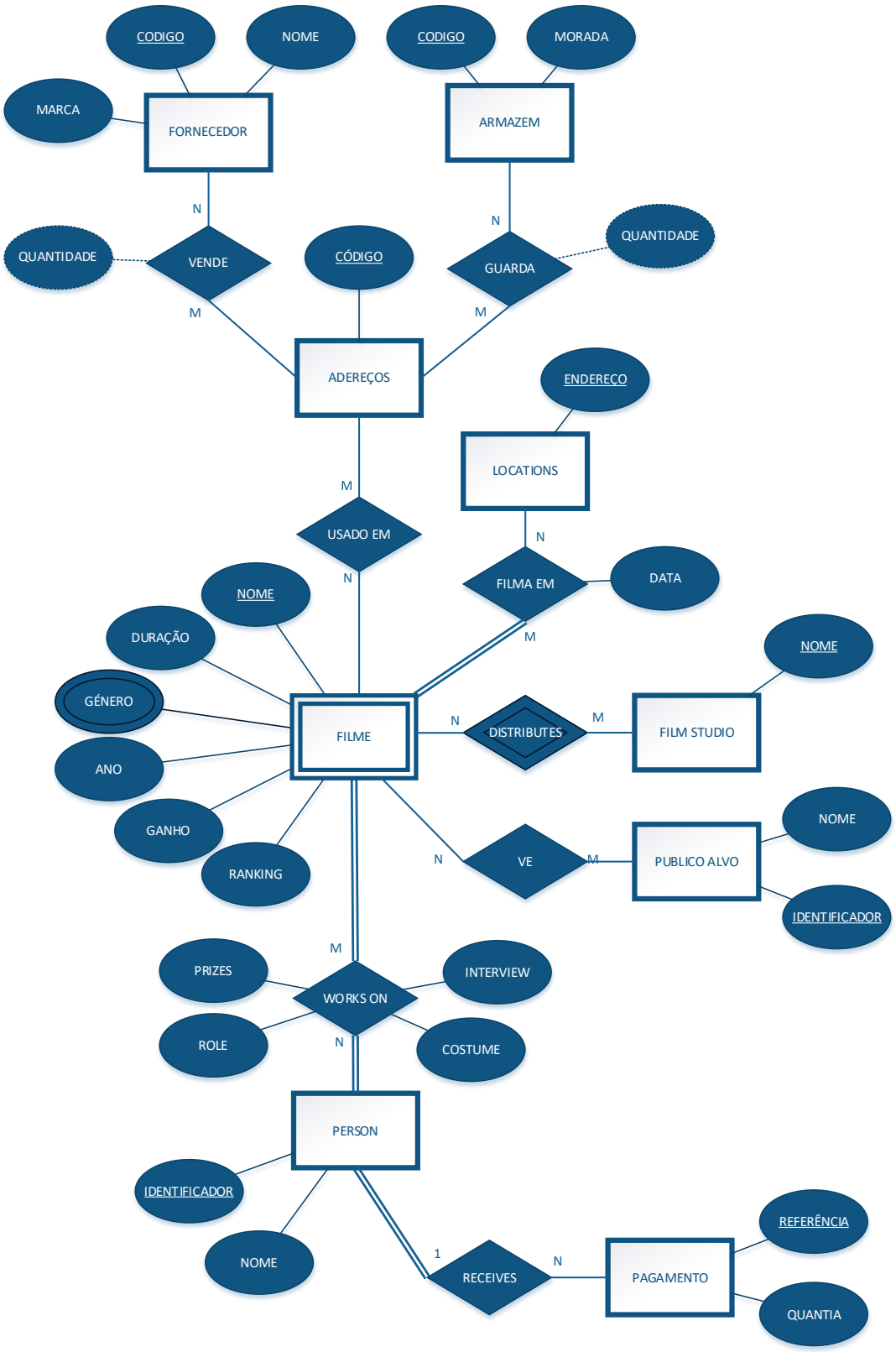
Além disso, para definir um filme não nos chega o Título, tendo em conta que poderão existir filmes com o mesmo nome. Assim, precisamos de definir também o **estúdio do filme**, que tem um nome (ex.: Marvel Studios).

Também queremos poder guardar um registo das **pessoas que trabalharam num filme**, com diversas *roles*, desde atores a *costume designers*. Estas pessoas também recebem pagamentos, que devem ser registados na base de dados com a quantia e a referência. Além disso, queremos ser capazes de guardar prémios que uma pessoa tenha recebido em determinado filme.

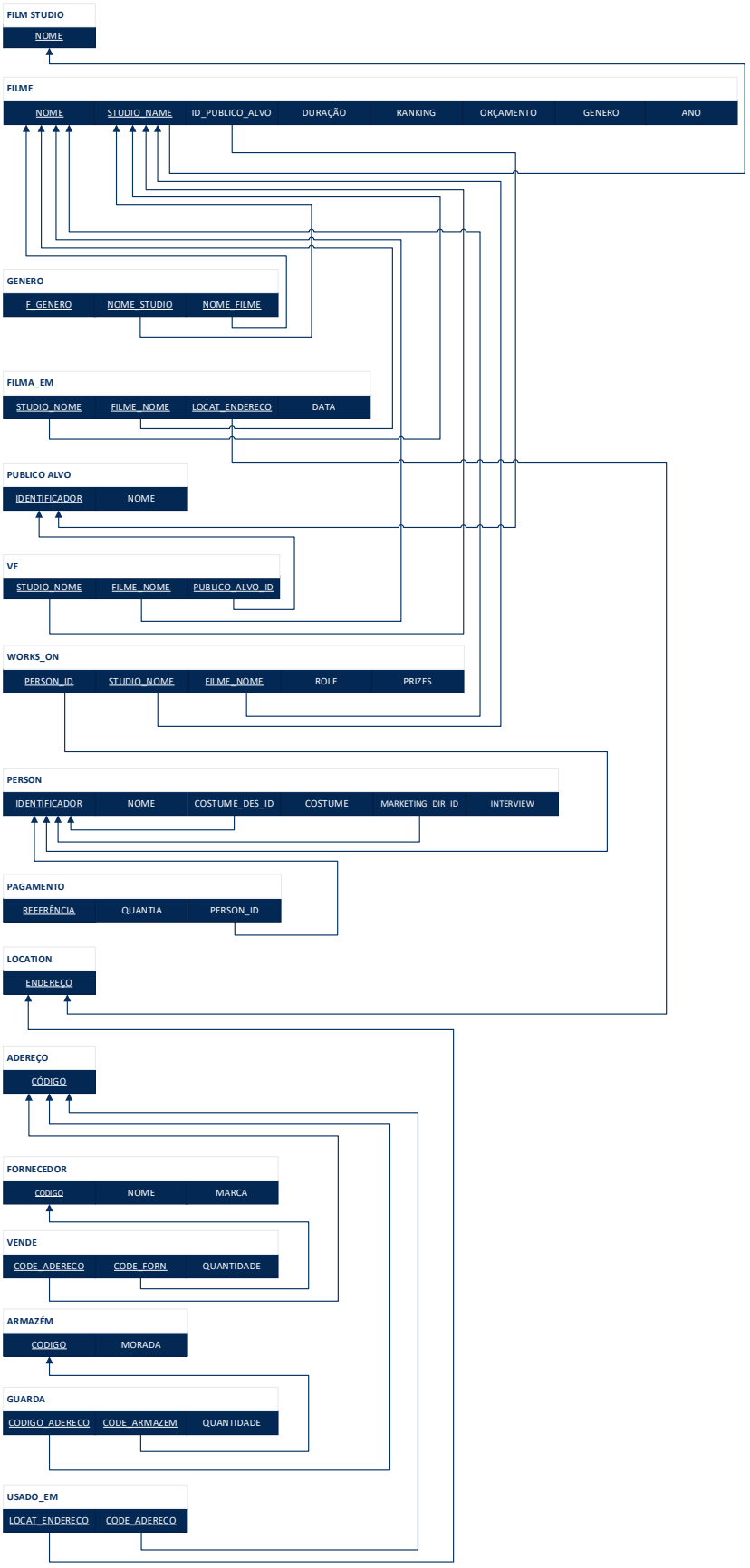
Para concluir, queremos guardar os **adereços** utilizados num filme, guardando informações como:

- Armazém onde estão guardados
- Quantidade que temos
- Fornecedor

DER – DIAGRAMA ENTIDADE-RELACIONAMENTO



MR – MODELO RELACIONAL



ÍNDICES

Para melhorar a performance da nossa base de dados, criámos quatro índices, dos quais:

1. idx_movies_year
   Para o *Ano* de um *Filme*

2. PessoaIdx
   Para o *Nome* de uma *Pessoa*

3. FilmeIdx
   Para o *FilmeNome* de um *Filme*

4. StudioFilmeIdx
   Para o *Nome* de um *FilmStudio*

TRIGGERS

Criámos os seguintes triggers:

1. role_prize
   Ao inserir ou atualizar um tuplo da tabela *WorksOn*, verifica se um *Prize* é compatível com a respetiva *Role*.

   Por exemplo, alguém com a role *Main Actor* não pode ter um prémio de *"Best Secondary Actor"*

STORED PROCEDURES

1. Para começar, desenvolvemos dois *stored procedures* para facilitar a **inserção de tuplos** em algumas tabelas. Temos:

   addActor
   Este *procedure* adiciona tuplos à tabela *WorksOn*, pelo *Role*, *FilmeNome, StudioNome* e *Nome* de uma *Pessoa*, que também é inserida caso não exista na base de dados.

addLocation

Este procedure adiciona tuplos à tabela *FilmaEm* pelo *FilmeNome*, *StudioNome*, *Data* e *Endereço* de uma *Localização*, que é também inserida caso ainda não exista na base de dados.

addProp

Este procedure adiciona tuplos à tabela *UsadoEm* pelo *FilmeNome, StudioNome* e *Código* de *Adereço*, que também é inserido caso o seu *Nome* não esteja presente na base de dados.

2. Para **obter todas as informações de um *Filme*** criámos também um *stored procedure* que, da junção do *Filme* e do *PublicoAlvo* devolve o *Ano, Duração, PublicoAlvo* (*Nome*)*, Ganho* e *Ranking.*

3. Para **verificação de login** criámos um stored procedure que verifica se o *Username* e a *password* correspondem a um utilizador válido na base de dados.

UDF – USER DEFINED FUNCTIONS

1. No que toca a *UDFs*, utilizámo-las principalmente para **implementar os filtros de pesquisas** da interface, tendo as seguintes funções:

   genre_movies
   Retorna uma tabela com todos os *Filmes* de um determinado *Género*

   year_movies
   Retorna uma tabela com todos os *Filmes* de um determinado *Ano*

   ator_filtro
   Retorna uma tabela com todos os *Filmes* em que determinada *Pessoa* trabalhou

   filme_filtro
   Retorna uma tabela com todos os *Filmes* com um determinado *Nome*

   filmestudio_filtro
   Retorna uma tabela com todos os *Filmes* de um determinado *Studio*

   nome_ator
   Retorna uma tabela com todas as *Pessoas* de um determinado *Nome*

   role_ator
   Retorna uma tabela com todas as *Pessoas* de uma determinada *Role (ex.: "Main Actor")*

   filme_ator
   Retorna uma tabela com todas as *Pessoas* que trabalharam num determinado *Filme*

   prizes_ator
   Retorna uma tabela com todas as *Pessoas* que receberam determinado *Prémio*

2. Utilizámo-las também para **retornar tabelas com múltiplos valores de um atributo**, tendo uma para retornar os *Géneros* de um *Filme*, as *Pessoas* que trabalharam num *Filme*, as *Localizações* em que um *Filme* foi gravado e os *Adereços* utilizados num *Filme*.

3. Finalmente, criámos a função money_movie que permite **TODO**

ADICIONADO:
TRANSACTION + ROLLBACK + TRY CATCH
NORMALIZACAO
AVG E COUNT
TYPE
RIGHT OUTER JOIN

NAO ADICIONA POR ERRO
GROUP BY
ORDER BY (não da em funcoes e procs e views :/)
ON DELETE SET NULL, ON UPDATE SET NULL
CLUSTERED INDEX

NAO ADICIONADO
ANY, ALL
IN, NOT IN
REUNION, UNION, PRODUTO
DOMAIN (mesmo que type)
CONSTRAINT (useless)

AINDA NAO ADICIONADO
DELETE
BETWEEN