

Miguel Moura  **up201609149**@fe.up.pt

Antero Santos **up201604602**@fe.up.pt

Simão Santos **up201505695**@fe.up.pt

## Bases de Dados

ASM- Empresa de Videojogos

MIEIC06- Grupo 605

(6 de Março de 2018)

Descrição

Este projeto baseia-se na gestão de uma empresa de Videojogos semelhante à EA com as suas várias divisões.

A companhia possui várias divisões, jogos de desporto, jogos de ação, jogos de carros, etc. Sendo que todas elas obedecem à mesma estrutura.

Cada empresa tem uma **Divisão** encarregue da criação de um jogo e gestão dos seus **Departamentos** e **Funcionários**. Uma **Divisão** precisa de um nome.

Cada **Divisão** possui também uma **Morada**, portanto o nome da rua, a cidade em que está baseada, o número (da porta) e ultimamente o país.

Sobre os **Funcionários** é guardado na base de dados o seu nome, a sua idade e o tempo (em meses) em que trabalham no seu respetivo **Departamento**. Está também guardada a sua **Morada** com os mesmos atributos da **Divisão**.

O produto final, o **Jogo**, necessita de um nome, e um **Género de Jogo**. Cada **Jogo** tem apenas um **Género de Jogo**, mas um **Género de Jogo** pode ter muitos jogos. Por isso, **Género de Jogo** tem um nome, por exemplo: terror, ação, Role-Playing, e uma idade recomendada.

Cada **Jogo** é jogado numa **Plataforma**, esta que tem um nome, e uma marca responsável pela sua conceção.

Um **Jogador** para se registar e jogar necessita de inserir um username e a sua idade. Um **Jogador** pode ser um tester (alguém que joga o jogo em fases BETA ou em épocas antes de se lançar um update, para certificar que não há “bugs” ou erros nas atualizações) e ter também pacote premium (acesso exclusivo a conteúdo não disponível a jogadores que jogam de graça).

Todos estes jogadores constituem uma **Playerbase**, que corresponde a apenas um **Jogo**. **Playerbase** esta que contém número total de jogadores que jogam o dito **Jogo** e a média de idades dos jogadores, para fins estatísticos.

Atributos

DIVISÃO

* Nome

MORADA

* Nome da Rua
* Cidade
* Número
* País

DEPARTAMENTO

* Número de Funcionários

FUNCIONÁRIO

* Nome
* Idade
* Tempo (meses)

JOGO

* Nome

GÉNERO DO JOGO

* Nome
* Idade Recomendada

JOGADOR

* Idade
* Premium
* Tester
* Username

PLAYERBASE

* Nº de jogadores
* Idade Média

PLATAFORMA

* Nome
* Marca

Esquema Relacional e Dependências Funcionais

* Divisão(id, nome,moradaID->Morada)
  + {Id} - >{nome,moradaID }
* Morada(id, nomeRua, cidade, numeroPorta,país)
  + {Id}-> {nomeRua, cidade, numeroPorta,país}
* Funcionário(id, nome, idade, tempoServiço, depID->Departamento)
  + {Id}->{nome, idade, tempoServiço, depID }
* Departamento(id, nome, tamanhoEquipa)
  + {Id}->{ nome, tamanhoEquipa }
* Jogo(id, nome, playerbaseId->Playerbase)
  + {Id}->{ nome, playerbaseId}
* Playerbase(id, numeroTotal, idadeMédia)
  + {Id}->{ numeroTotal, idadeMédia }
* Jogador(id, username, idade, tester, premium)
  + {Id}->{ username, idade, tester, premium }
* Género(id, nome, idadeRecomendada)
  + {Id}->{ nome, idadeRecomendada }
* Plataforma(id,nome, marca)
  + {Id}->{ nome, marca }
* DepartamentoDivisão(depID->Departamento, divID->Divisão)
* FuncionárioDepartamento(funcID->Funcionário, depID->Departamento)
* JogadorPlayerbase(jogadorID->Jogador, playerbaseID->Playerbase)
* jogoDivisão(jogoId->Jogo, divID->Divisão)
* jogoGénero(jogoID->Jogo, generoID->Genero)
* jogoJogador(jogoId->Jogo, jogadorID->Jogador)
* jogoPlataforma(jogoID->Jogo, plataformaID->Plataforma)

Formas Normais

Relativamente às restantes relações enumeradas na página anterior, não existirá quebra da 3ª Forma Normal, nem da Forma Normal de Boyce-Codd na medida em que o lado esquerdo de cada dependência é uma super-key do esquema relacional – condição suficiente para cumprir ambas.

Restrições

Para assegurar uma boa manutenção da base de dados recorreu-se ao uso de restrições na produção de várias classes.

Nas ocasiões onde a restrição NOT NULL é associada a um atributo, manifesta-se a obrigatoriedade da existência deste mesmo atributo para a formação mínima da classe. Ficam como exemplo:

* Os atributos *username*, *idade, tester e premium* da classe Álbum são declarados como NOT NULL, pois o conceito de jogador não poderia subsistir sem a existência destas propriedades.
* Os atributos *nome*, *idade e tempoServico* da classe Funcionário são declarados como NOT NULL, pois o conceito de funcionário não poderia subsistir sem a existência destas propriedades.

A restrição UNIQUE foi especialmente aplicada a atributos identificadores de uma certa classe, embora estes não operem como chaves. Fica como exemplo:

* Atributo *nome* da classe Género – não existem dois géneros de jogo com a mesma denominação.
* Atributo *nome* da classe Jogo – a base de dados não permite o registo de jogos com nomes repetidos.
* Atributo *username* da classe Jogador – a base de dados não permite o registo de jogadores com usernames repetidos.

Por outro lado, a restrição CHECK haverá sido praticada com a mentalidade de restringir certos aspetos de atributos, aliando uma maior segurança a um melhor controlo dos dados. A listagem de usos inclui:

* O atributo *idade* na classe Funcionário não deverá ser inferior a 18 anos.
* O atributo *numeroTotal* na classe Playerbase tem um limite mínimo de 1 jogador, uma vez que sem jogadores não há uma playerbase.
* O atributo *tamanhoEquipa* na classe Playerbase tem um limite mínimo de 1 funcionário.