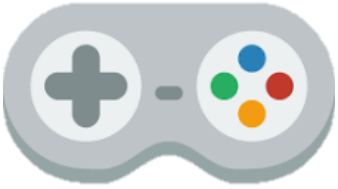
U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Bases de Dados

Mestrado Integrado em Engenharia Informática e Computação

GAME BOOK



The best gaming social experience!

Grupo 308
João Paulo Gomes Torres Abelha – **up201706412**
João Rafael Gomes Varela – **up201706072**
Vítor Hugo Pereira Barbosa – **up201703591**

Data de Entrega:
26 de maio, 2019

Índice

Descrição da Aplicação

Game Book é a aplicação que permite juntar o melhor dos mundos *Gaming* e *Social*, conferindo ao utilizador um *Social Gaming Experience* única. Através desta *app*, o utilizador poderá satisfazer-se ao comprar e jogar os seus títulos favoritos e ainda partilhar esta fabulosa experiência com os seus amigos.

O objetivo deste projeto será criar uma base de dados eficiente e intuitiva que permita gerir a informação dos utilizadores da aplicação, dos jogos disponíveis na loja, das empresas produtoras de jogos, de torneios que premiaram os utilizadores, entre outros, utilizando esses dados para criar recomendações aos utilizadores, às empresas e aos criadores de torneios.

Especificações da Aplicação

Na aplicação Game Book, cada **Utilizador** tem uma série de dados pessoais como um identificador, nome, e-mail, username, password, data de nascimento, morada e uma foto de perfil. Para estes dados são impostas algumas restrições, o e-mail é constituído obrigatoriamente pelos caracteres @ e . , tendo entre eles pelo menos um caratere, o username deve ter um comprimento entre 6 e 12 caracteres e a password um comprimento entre 4 e 15 caracteres.

Cada utilizador deve ter a ele associado um **Cartão de crédito** que apresenta um número que o identifica, um saldo e uma data de validade. É através deste cartão que serão feitas as compras de jogos e serão recebidos os prémios relacionados com a participação em torneios.

Como em qualquer rede social o utilizador pode seguir outros utilizadores, bem como ser seguido, podendo saber informação relativa aos jogos comprados pelos utilizadores que segue e ainda aos torneios e grupos em que estes se encontram inseridos.

Cada utilizador pode adquirir um jogo, comprando-o numa determinada data, podendo atribuir-lhe uma classificação (**Compra**). Do **Jogo** importa saber o seu nome, preço, data de lançamento e idade mínima. Para além disso, um utilizador caso não possua um determinado jogo (xor), pode adicioná-lo aos seus interesses (**Desejo de compra**), atribuindo-lhe um valor numérico de interesse de 1 a 10.

Cada jogo é lançado por uma **Empresa** e esta é detentora de vários jogos. Desta importa saber o seu nome, contato e NIF. Cada jogo pode estar disponível em diferentes **Plataformas** e pode ainda ter diferentes **Géneros**. Um jogo pode ser de apenas dois tipos distintos : **Online** e **Offline**.

Sempre que um utilizador joga online, é criada uma **Sessão**, onde se guarda a data inicial e a data final ambas no formato “YYYY-MM-DDThh:mm:ss” permitindo desta forma obter informação do número de horas jogadas de cada utilizador e ainda saber quais os utilizadores que jogam juntos um determinado jogo.

O sistema permite ainda fazer **Grupos** que são constituídos por vários utilizadores, podendo cada utilizador pertencer a vários grupos. Cada grupo tem um nome, uma data de criação e uma possível data de destruição.

A **Adesão** ao grupo é feita num determinado dia e o utilizador apresenta, logo à partida, um estatuto de membro comum, podendo ainda ser membro honorário ou administrador. Será ainda guardada, caso exista, a data de expulsão do utilizador de um determinado grupo.

Por último o sistema permite a realização de **Torneios** , com início numa determinada data (data inicial) , podendo ou não acabar no mesmo dia (data final), tendo cada torneio um nome associado. Cada utilizador obtém uma classificação, estando aos cinco primeiros participantes associado um **Prémio**. O prémio é dependente do torneio e da classificação, podendo a quantia correspondente ser entregue ou não, caso o número de participantes não seja o suficiente.

Diagrama UML

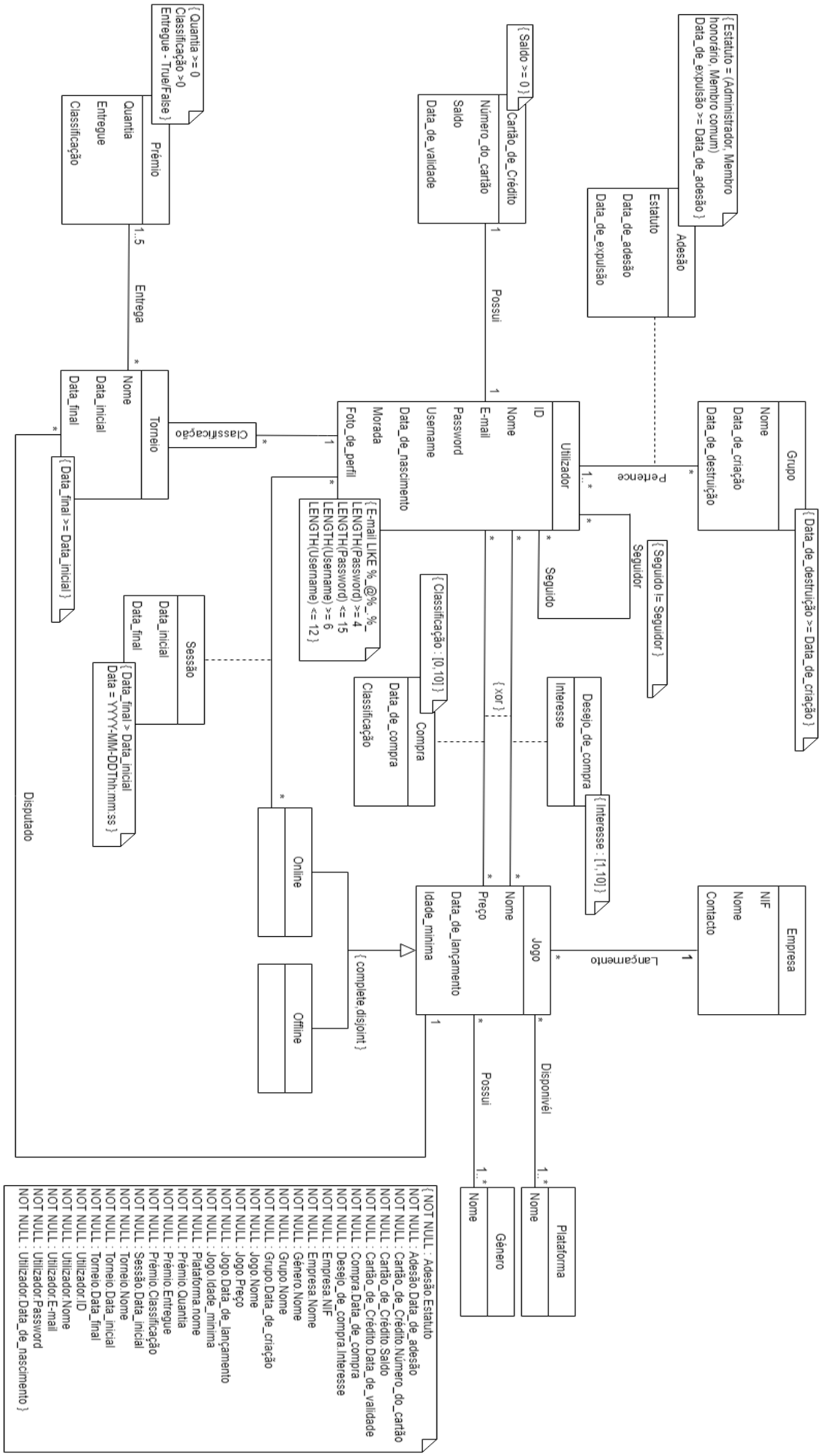
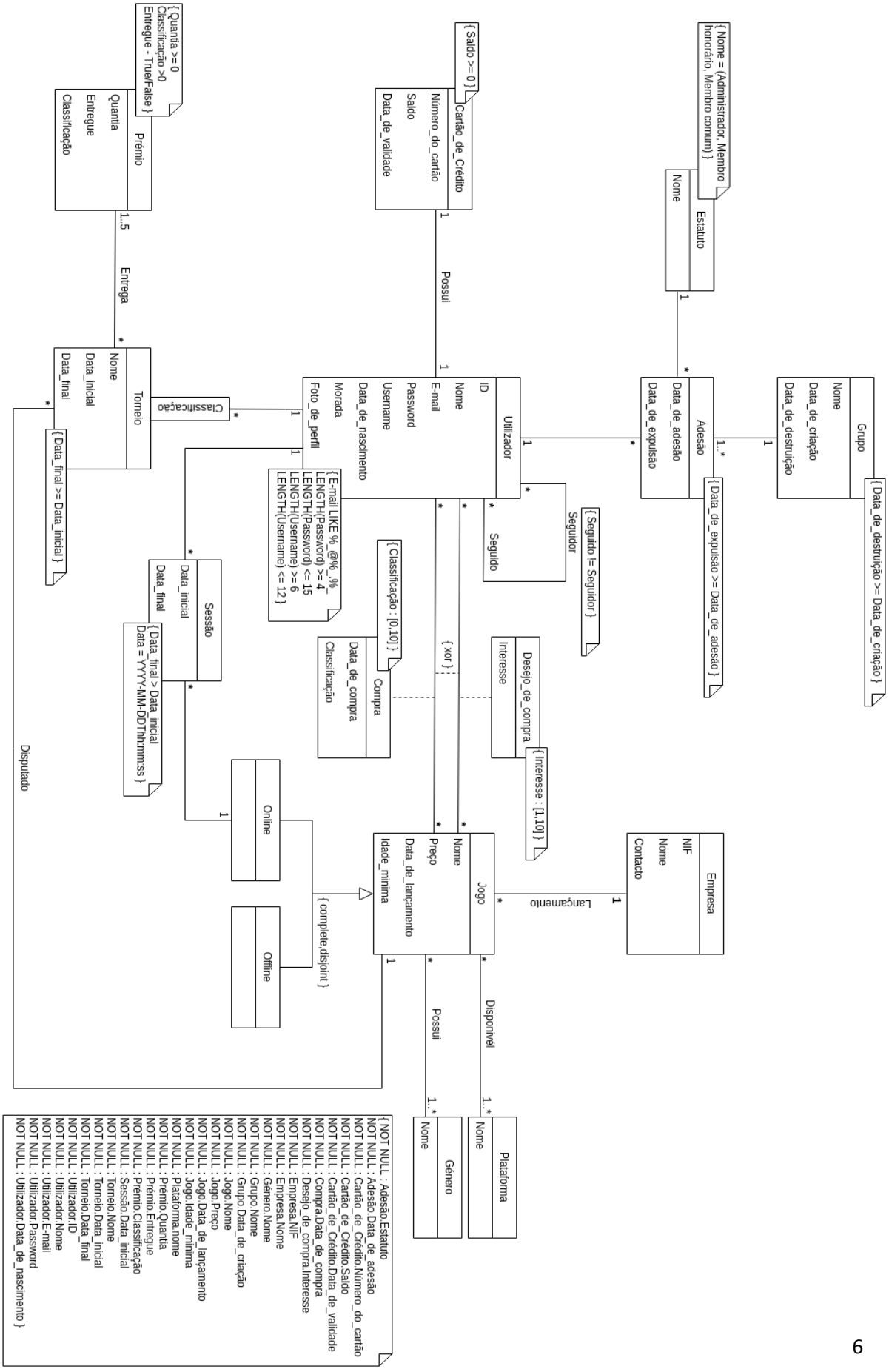


Diagrama UML Revisto



Modelo Relacional

Optou-se nas dependências funcionais por incluir em todas as relações ID's para permitir o uso do *alias* RowID e para que nas foreign keys o espaço ocupado no disco fosse reduzido. A escolha mais difícil foi nas generalizações, acabando por ser usado o E/R style, uma vez que a superclasse tem muitos atributos ao contrário das subclasses (generalização orientada a objetos eliminada) e a generalização *use nulls* não seria aplicável, uma vez que apenas os jogos online apresentam sessões, sendo necessário diferenciá-los(o que esta última generalização não garantia).

Utilizador (ID, nome, email, password, username, data_nascimento, morada, foto_perfil)

ID →nome, email, password, username, data_nascimento, morada, foto_perfil

email →ID, nome, password, username, data_nascimento, morada, foto_perfil

username →ID, nome, email, password, data_nascimento, morada, foto_perfil

ID é a *primary key*

SeguirUtilizador (IDseguidor → Utilizador, IDseguido → Utilizador)

IDseguidor e IDseguido são a *primary key* composta

IDseguidor e IDseguido são *foreign keys*

Cartao_de_credito (numero_de_cartao, saldo, data_de_validade, ID → Utilizador)

numero_de_cartao →saldo, data_de_validade, ID

ID →numero_de_cartao, saldo, data_de_validade

numero_de_cartao é uma *primary key*

ID é uma *foreign key*

Empresa (IDEmpresa, NIF, nome, contacto)

IDEmpresa → NIF, nome, contacto

NIF → IDEmpresa, nome,contacto

nome → IDEmpresa, NIF, contacto

contacto → IDEmpresa, NIF, nome

IDEmpresa é a *primary key*

Jogo (IDJogo, nome, preco, data_de_lancamento, idade_minima, IDEmpresa → Empresa)

IDJogo →nome, preco, data_de_lancamento, idade_minima, IDEmpresa

nome →IDJogo, preco, data_de_lancamento, idade_minima, IDEmpresa

IDJogo é a *primary key*

IDEmpresa é uma *foreign key*

Online (IDJogo → Jogo)

IDJogo é a *primary key*

IDJogo é uma *foreign key*

Offline (IDJogo → Jogo)

IDJogo é a *primary key*

IDJogo é uma *foreign key*

Plataforma (IDPlataforma, nome)

IDPlataforma →nome

nome →IDPlataforma

IDPlataforma é a *primary key*

PlataformaDisponível (IDPlataforma → Plataforma, IDJogo → Jogo)

IDPlataforma e IDJogo são a *primary key* composta

IDPlataforma e IDJogo são *foreign keys*

Genero (IDGenero, nome)

IDGenero → nome

nome → IDGenero

IDGenero é uma *primary key*

PossuiGenero (IDGenero → Genero, IDJogo → Jogo)

IDGenero e IDJogo são uma *primary key* composta

IDGenero e IDJogo são *foreign keys*

Compra (ID → Utilizador, IDJogo → Jogo, data_de_compra, classificacao)

ID, IDJogo →data_de_compra, classificacao

ID e IDJogo são uma *primary key* composta

ID e IDJogo são *foreign keys*

Desejo_de_Compra (ID → Utilizador, IDJogo → Jogo, interesse)

ID, IDJogo →interesse

ID e IDJogo são uma *primary key* composta

ID e IDJogo são *foreign keys*

Sessao (IDSessao, ID → Utilizador, IDJogo → Online, data_inicial, data_final)

IDSessao → ID, IDJogo, data_inicial, data_final

IDSessao é uma *primary key*

ID e IDJogo são *foreign keys*

Torneio (IDTorneio, nome, data_inicial, data_final, IDJogo → Jogo)

IDTorneio →nome, data_inicial, data_final, IDJogo

nome →IDTorneio, data_inicial, data_final, IDJogo

IDTorneio é uma *primary key*

IDJogo é uma *foreign key*

Premio (IDPremio, quantia, entregue, classificacao)

IDPremio → quantia, entregue, classificacao

IDPremio é uma *primary key*

EntregaPremio (IDTorneio → Torneio, IDPremio→ Premio)

IDTorneio e IDPremio são uma *primary key* composta

IDTorneio e IDPremio são *foreign keys*

Participacao (ID → Utilizador, IDTorneio → Torneio, classificacao)

ID, IDTorneio →classificacao

classificacao, IDTorneio →ID

ID e IDTorneio são uma *primary key* composta

ID e IDTorneio são *foreign keys*

Grupo (IDGrupo, nome, data_de_criacao, data_de_eliminacao)

IDGrupo →nome, data_de_criacao, data_de_eliminacao

IDGrupo é uma *primary key*

Estatuto (IDEstatuto, nome)

IDEstatuto →nome

nome → IDEstatuto

IDEstatuto é uma *primary key*

Adesao (IDAdesao, data_de_adesao, data_de_expulsao, IDEstatuto → Estatuto, IDGrupo → Grupo, ID → Utilizador)

IDAdesao →data_de_adesao, data_de_expulsao, IDEstatudo, IDGrupo, ID

IDAdesao é uma *primary key*

IDEstatuto, IDGrupo e ID são *foreign keys*

Análise de Dependências Funcionais e Formas Normais

Das dependências funcionais definidas na seção anterior, podemos confirmar que no lado esquerdo de qualquer uma, se encontra uma *key* da dependência. Esse facto pode ser demonstrado, visto que o fecho dos atributos do lado esquerdo de cada dependência permite chegar a qualquer um dos atributos da relação.

Utilizador:

$\{ID\}^+ = \{ID, nome, email, password, username, data\_nascimento, morada, foto\_perfil\}$

$\{username\}^+ = \{ID, nome, email, password, username, data\_nascimento, morada, foto\_perfil\}$

$\{email\}^+ = \{ID, nome, email, password, username, data\_nascimento, morada, foto\_perfil\}$

SeguirUtilizador:

$\{IDseguidor, IDseguido\}^+ = \{IDseguidor, IDseguido\}$

Cartao_de_credito:

$\{numero\_de\_cartao\}^+ = \{numero\_de\_cartao, saldo, data\_de\_validade, ID\}$

$\{ID\}^+ = \{numero\_de\_cartao, saldo, data\_de\_validade, ID\}$

Empresa:

$\{IDEmpresa\}^+ = \{IDEmpresa, NIF, nome, contacto\}$

$\{NIF\}^+ = \{IDEmpresa, NIF, nome, contacto\}$

$\{nome\}^+ = \{IDEmpresa, NIF, nome, contacto\}$

$\{contacto\}^+ = \{IDEmpresa, NIF, nome, contacto\}$

Jogo:

$\{IDJogo\}^+ = \{IDJogo, nome, preco, data\_de\_lancamento, idade\_minima, IDEmpresa\}$

$\{nome\}^+ = \{IDJogo, nome, preco, data\_de\_lancamento, idade\_minima, IDEmpresa\}$

Online:

$\{IDJogo\}^+ = \{IDJogo\}$

Offline:

$\{IDJogo\}^+ = \{IDJogo\}$

Plataforma:

$\{IDPlataforma\}^+ = \{IDPlataforma, nome\}$

$\{nome\}^+ = \{IDPlataforma, nome\}$

PlataformaDisponivel:

$\{IDPlataforma, IDJogo\}^+ = \{IDPlataforma, IDJogo\}$

Genero:

$\{IDGenero\}^+ = \{IDGenero, nome\}$

$\{nome\}^+ = \{IDGenero, nome\}$

PossuiGenero:

$\{IDGenero, IDJogo\}^+ = \{IDGenero, IDJogo\}$

Compra:

$\{ID, IDJogo\}^+ = \{ID, IDJogo, data\_de\_compra, classificacao\}$

Desejo_de_compra:

$\{ID, IDJogo\}^+ = \{ID, IDJogo, interesse\}$

Sessao:

$\{IDSessao\}^+ = \{IDSessao, ID, IDJogo, data\_inicial, data\_final\}$

Torneio:

$\{IDTorneio\}^+ = \{IDTorneio, nome, data\_inicial, data\_final, IDJogo\}$

$\{nome\}^+ = \{IDTorneio, nome, data\_inicial, data\_final, IDJogo\}$

Premio:

$\{IDPremio\}^+ = \{IDPremio, quantia, entregue, classificacao\}$

EntregaPremio:

$\{IDTorneio, IDPremio\}^+ = \{IDTorneio, IDPremio\}$

Participacao:

$\{ID, IDTorneio\}^+ = \{ID, IDTorneio, classificacao\}$

$\{classificacao, IDTorneio\}^+ = \{ID, IDTorneio, classificacao\}$

Grupo:

$\{IDGrupo\}^+ = \{IDGrupo, nome, data\_de\_criacao, data\_de\_eliminacao\}$

Estatuto:

$\{IDEstatuto\}^+ = \{IDEstatuto, nome\}$

$\{nome\}^+ = \{IDEstatuto, nome\}$

Adesao:

$\{IDAdesao\}^+ = \{IDAdesao, data\_de\_adesao, data\_de\_expulsao, IDEstatudo, IDGrupo, ID\}$

Pode-se observar que em qualquer dependência funcional, o seu lado esquerdo constitui uma (super)*key* da relação. Assim sendo, é possível confirmar que o modelo relacional se encontra na ***Boyce-Codd Normal Form*** e não há violações da forma normal.

Conclui-se também, imediatamente, que o modelo respeita a ***3rd Normal Form*** visto que esta forma representa um *super set* da BCNF.

Restrições

Para garantir a coerência e o bom funcionamento da base de dados, foram aplicadas várias restrições. As restrições impostas para além daquelas já realizadas pela estrutura e tipo de dados utilizados foram, nas várias relações, as seguintes:

Utilizador:

- **ID** é uma chave primária (PRIMARY KEY)
- **nome** não pode ser nulo (NOT NULL)
- **email** não pode ser nulo (NOT NULL), deve ser único (UNIQUE) e deve ter pelo menos uma letra antes do @ , pelo menos outra entre o @ e o . e por fim pelo menos outra depois do . isto é ser do tipo: %_@_.%_ (CHECK email LIKE '%_@_.%_')
- **password** não pode ser nulo (NOT NULL) e deve ter entre 4 a 15 caracteres (CHECK (LENGTH(password) >=4 AND LENGTH(password)<=15))
- **username** não pode ser nulo (NOT NULL), deve ser único (UNIQUE) e deve ter entre 6 a 12 caracteres (CHECK (LENGTH(username) >=6 AND LENGTH(username)<=12))
- **data_nascimento** não pode ser nulo (NOT NULL)
- **foto_perfil** deve estar no formato png (CHECK (foto_perfil LIKE "%_.png")), sendo uma referência para uma imagem(e não a imagem em si).

SeguirUtilizador:

- **IDseguidor** e ID**seguido** formam uma chave primária composta (PRIMARY KEY), são ambas chaves estrangeiras (FOREIGN KEY) e devem ser diferentes entre si (CHECK IDseguidor != IDseguido)

Cartao_de_credito:

- **numero_de_cartao** é uma chave primária (PRIMARY KEY)
- **saldo** deve ser um valor positivo ou igual a zero (CHECK saldo >= 0) e não pode ser nulo (NOT NULL)
- **data_de_validade** não pode ser nulo (NOT NULL)
- **ID** é uma chave estrangeira (FOREIGN KEY), deve ser único (UNIQUE) e não pode ser nulo (NOT NULL)

Empresa:

- **IDEmpresa** é uma chave primária (PRIMARY KEY)
- **nome** não pode ser nulo (NOT NULL) e deve ser único (UNIQUE)
- **NIF** não pode ser nulo (NOT NULL), deve ser único (UNIQUE) e deve ter exatamente 9 caracteres (CHECK (LENGTH(NIF) = 9)
- **contacto** deve ser único (UNIQUE) e deve ter exatamente 9 caracteres (CHECK (LENGTH(contacto) = 9)

Jogo:

- **IDJogo** é uma chave primária (PRIMARY KEY)
- **nome** não pode ser nulo (NOT NULL) e deve ser único (UNIQUE)
- **preco** não pode ser nulo (NOT NULL) e deve ser positivo ou zero (CHECK preco >=0)
- **data_de_lancamento** não pode ser nulo (NOT NULL)
- **idade_minima** não pode ser nulo (NOT NULL) e deve ser maior que zero (CHECK idade_minima > 0)
- **IDEmpresa** é uma chave estrangeira (FOREIGN KEY)

Online:

- **IDJogo** é uma chave primária (PRIMARY KEY) e também uma chave estrangeira (FOREIGN KEY)

Offline:

- **IDJogo** é uma chave primária (PRIMARY KEY) e também uma chave estrangeira (FOREIGN KEY)

Plataforma:

- **IDPlataforma** é uma chave primária (PRIMARY KEY)
- **nome** não pode ser nulo (NOT NULL) e deve ser único (UNIQUE)

PlataformaDisponivel:

- **IDPlataforma** e **IDJogo** formam uma chave primária composta (PRIMARY KEY) e são ambas chaves estrangeiras (FOREIGN KEY)

Genero:

- **IDGenero** é uma chave primária (PRIMARY KEY)
- **nome** não pode ser nulo (NOT NULL) e deve ser único (UNIQUE)

PossuiGenero:

- **IDJogo** e **IDGenero** formam uma chave primária composta (PRIMARY KEY) e são ambas chaves estrangeiras (FOREIGN KEY)

Compra:

- **ID** e **IDJogo** formam uma chave primária composta (PRIMARY KEY) e são ambas chaves estrangeiras (FOREIGN KEY)
- **data_de_compra** não pode ser nulo (NOT NULL)
- **classificacao** deve ter um valor entre 0 e 10 (CHECK (classificacao IS NULL OR (classificacao >= 0 AND classificacao <= 10)))

Desejo_de_compra:

- **ID** e **IDJogo** formam uma chave primária composta (PRIMARY KEY) e são ambas chaves estrangeiras (FOREIGN KEY)
- **interesse** não pode ser nulo (NOT NULL) e deve ser um valor entre 1 e 10 (CHECK (interesse > 0 AND interesse <= 10))

Sessao:

- **IDSessao** é uma chave primária (PRIMARY KEY)
- **ID** e **IDJogo** são ambas chaves estrangeiras (FOREIGN KEY) e nenhuma delas pode ser nula (NOT NULL)
- **data_inicial** não pode ser nulo (NOT NULL)
- **data_final** deve ser maior que data inicial (CHECK (data_inicial > data_final))

Torneio:

- **IDTorneio** é uma chave primária (PRIMARY KEY)
- **data_inicial** não pode ser nulo (NOT NULL)
- **data_final** não pode ser nulo (NOT NULL) e deve ser maior ou igual que a data inicial (CHECK (data_final >= data_inicial))
- **nome** não pode ser nulo (NOT NULL) e deve ser único (UNIQUE)
- **IDJogo** é uma chave estrangeira (FOREIGN KEY) e não pode ser nulo (NOT NULL)

Premio:

- **IDPremio** é uma chave primária (PRIMARY KEY)
- **quantia** deve ser positiva ou igual a zero (CHECK quantia >= 0) e não pode ser nulo - (NOT NULL)
- **entregue** representa um *booleano* (CHECK (entregue = 0 OR entregue =1)) e não pode ser nulo (NOT NULL)
- **classificacao** deve ser um valor entre 1 e 5 (CHECK (classificacao >= 1 AND classificacao <= 5)) e não pode ser nulo (NOT NULL)

EntregaPremio:

- **IDTorneio** e **IDPremio** formam uma chave primária composta (PRIMARY KEY) e são ambas chaves estrangeiras (FOREIGN KEY)

Participacao:

- **ID** e **IDTorneio** forma uma chave primária composta (PRIMARY KEY) e são ambas chaves estrangeiras (FOREIGN KEY)
- **classificacao** deve ser maior que zero (CHECK classificacao > 0), não poder ser nulo (NOT NULL) e deve ser único (UNIQUE)
- **classificacao** e **IDTorneio** formam uma UNIQUE KEY composta

Grupo:

- **IDGrupo** é uma chave primária (PRIMARY KEY)
- **nome** não pode ser nulo (NOT NULL)
- **data_de_criacao** não pode ser nulo (NOT NULL)
- **data_de_eliminacao,** caso exista, deve ser superior à **data_de_criação** (CHECK (data_de_eliminacao IS NULL OR data_de_eliminacao >= data_de_criacao))

Estatuto:

- **IDEstatuto** é uma chave primária (PRIMARY KEY)
- **nome** deve ser único (UNIQUE), não pode ser nulo (NOT NULL) e deve ser um de três diferentes tipos (CHECK (nome = 'Administrador' OR nome= 'Membro Honorario' OR nome = 'Membro Comum')) e por defeito deve ser um Membro Comum (Default('Membro Comum'))

Adesao:

- **IDAdesão** é uma chave primária (PRIMARY KEY)
- **data_de_adesao** não pode ser nulo (NOT NULL)
- **data_de_expulsao,** caso exista, deve ser maior que que a data_de_adesao (CHECK(data_de_expulsao IS NULL OR data_de_expulsao >= data_de_adesao))
- **IDEstatuto, IDGrupo** e **ID** são chaves estrangeiras (FOREIGN KEY) e nenhuma delas pode ser nula (NOT NULL)

Nota: Ao inserir uma data na tabela devem ser inseridas no formato: “YYYY-MM-DD” . Para além disso na tabela sessão a data_final e data_inicial devem ser inseridas com o formato “YYYY-MM-DDThh:mm:ss”.

Outras Considerações

Para maximizar a compatibilidade entre o SQL e outras *database engines* foi tido em conta o conceito de afinidade. Sabendo que a afinidade duma coluna é o tipo recomendado para essa coluna, maximizou-se o seu uso em detrimento de outros tipo de dados.

Foi imperativo analisar todas as restrições com cautela, uma vez que são elas que permitem manter a consistência e coerência da base de dados, garantindo que o (tipo de dados) e os seus valores são corretos e coerentes. Como PRIMARY KEYS foram usados inteiros (INTEGER) de forma a , não só a ocupar menos espaço, mas também a servir de ROWID . A primary key de uma tabela rowid, no caso de existir, geralmente não é a "verdadeira" chave primária, no sentido de não ser a chave exclusiva usada pelo mecanismo de armazenamento B-tree adjacente. A exceção a essa regra, e que foi aplicada, é quando a tabela rowid declara uma INTEGER PRIMARY KEY. Desta forma, esta torna-se um *alias* para o rowid. Geralmente, não se deve aceder ao rowid diretamente, mas em vez disso usar uma INTEGER PRIMARY KEY porque se o rowid não é um *alias* da INTEGER PRIMARY KEY não é persistente e pode mudar. Deste modo, o acesso aos registos, via rowid, é altamente otimizado e muito rápido.

A integridade referencial é um conceito importante no design de bases de dados. O termo refere-se a um estado quando todas as referências em uma base de dados são válidas e não existem links inválidos entre as várias tabelas que compõem o sistema. Quando existe integridade referencial, qualquer tentativa de vincular a um registro que ainda não existe falhará; isso ajuda a evitar erros do usuário, produzindo um banco de dados mais sólido, preciso e útil. A integridade referencial é geralmente implementada através do uso de chaves estrangeiras. Quando existe uma alteração destes valores quer por eliminação, quer por simples mudança do valor devemos ter especial atenção para não tornarmos a base de dados incoerente, devendo ser feitas as alterações necessárias à(s) relação(ões).

Por último, no ficheiro onde foi feita inserção de valores, para além de ter sido feito um extenso povoamento, garantiu-se que foram inseridos valores que cobrem as diferentes possibilidades/situações, sendo que todos os valores são coerentes, cobrindo mesmo aquelas situações que serão verificadas na terceira parte do trabalho com triggers.

Interrogação da Base de Dados

De seguida, estão dispostas 10 *Queries* , descritas em linguagem natural, que se consideraram relevantes para o problema em questão, apresentando informação importante e pertinente sobre a base de dados. É de notar que, durante a sua construção, se tiveram em conta fatores como eficiência e complexidade da interrogação.

Query 1: Obtém o nome e *username* dos utilizadores que seguem um número de pessoas igual ao número de seguidores. Esta *Query* é importante para efeitos estatísticos.

Query 2: Obtém o id, nome, *username* e número de seguidores do utilizador mais influente do *GameBook*, isto é, o utilizador com maior número de seguidores. Esta *Query* é importante para saber quem é o utilizador com mais influência e maior impacto sobre o resto dos utilizadores.

Query 3: Obtém o nome e os lucros totais (em vendas de jogos) de cada empresa. Esta *Query* é importante para efeitos estatísticos, dando-nos a conhecer o rendimento de qualquer empresa, mostrando qual é a empresa mais bem sucedida que faz jogos que agradam aos consumidores.

Query 4: Obtém o nome e número de seguidores dos grupos. Originalmente, a querie apresenta os três grupos mais influentes, mas esse número poderá ser modificado. A influência de cada grupo é medida pelo somatório de seguidores de cada membro do grupo. Esta *Query* é importante para saber qual o grupo mais popular e com maior influência sobre os utilizadores.

Query 5: Obtém, para cada empresa, o seu nome, dinheiro entregue em prémios e dinheiro por entregar e a soma de ambos (dinheiro que irá acabar por ser investido em torneios). Esta *Query* poderá ser importante por várias razões, como para se saber quais empresas entregam os prémios de valor mais elevado e quais ainda têm prémios por entregar.

Query 6: Obtém pares de id's de utilizadores que se seguem mutuamente. Esta *Query* é importante para efeitos estatísticos como por exemplo para saber que utilizadores são "compatíveis" entre si.

Query 7: Obtém o nome e número de géneros jogados, para cada utilizador que tem jogos cujo número de géneros diferentes é superior a metade dos géneros existentes (valor pode ser especificado na interrogação). Esta *Query* é relevante, na medida em que pode indicar que utilizadores jogam jogos de géneros mais variados.

Query 8: Obtém, para cada utilizador, a diferença de idade entre ele e a média de idades dos seus seguidores. Esta *Query* permite saber que faixa etária relativa é o público de cada utilizador.

Query 9: Obtém, para os 10 (número pode ser especificado na interrogação) jogos mais vendidos, o seu nome, número de sessões e vendas. Esta *Query* é importante para efeitos estatísticos, verificando-se entre os jogos mais vendidos qual é o que tem mais sessões.

Query 10: Obtém, para cada utilizador, o seu id e a percentagem de horas jogadas, relativamente a todas as horas jogadas na plataforma. Esta *Query* permite saber quais os jogadores mais ativos, ou seja, que investem mais tempo a jogar jogos da plataforma.

Adição de Gatilhos à Base de Dados

De seguida, constam alguns dos gatilhos, representados em linguagem natural, que são relevantes para o bom funcionamento da nossa base de dados.

Trigger 1: Xor_Desejo_Compra - o *trigger* é realizado sempre que um jogo é comprado e este pertencia à tabela Desejo_de_compra desse utilizador. Como um jogo adquirido não pode fazer parte dos desejos de compra de um utilizador, o tuplo respetivo da tabela Desejo_de_compra é eliminado.

Trigger 2: Compra_Idade_Minima - este *trigger* server para impedir que um utilizador faça a compra de um jogo, cuja idade mínima é superior à idade atual do utilizador.

Trigger 3: Entrega_Premio - este *trigger* atualiza o saldo do cartão de crédito de um determinado utilizador que atingiu uma classificação com direito a prémio, quando na tabela Premio, o atributo 'entregue' passa a ser igual a 1.

Existiam muitos outros *triggers* que poderíamos ter optado por fazer e que fazem sentido na base de dados, contudo, optamos pelos três acima pelo facto de se destacarem, não só pela sua importância, mas também por serem diferentes da maioria dos gatilhos que podiam ter sido implementados. A título exemplificativo, há muitos *triggers* que podiam ser realizados semelhantes ao *Trigger 2*, visto que há muitas datas que só fazem sentido numa determinada ordem cronológica. Para o *Trigger* 1, outro que poderia ser realizado semelhante seria a outra "metade" do *XOR*, garantindo que não é adicionado ao desejo de compra um jogo já obtido (mas seria muito semelhante).