Universidade do Minho
Mestrado Integrado em Engenharia Informática
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Lectivo de 2016/2017

Gestão de um jogo online

Vitor Brito (A73960), Daniel Soares (A73740), Duarte Freitas (63129), Diogo Brandão (76534)

Novembro de 2016

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

Gestão de um jogo online

Vitor Brito (A73960), Daniel Soares (A73740),

Duarte Freitas (63129), Diogo Brandão (76534)

Novembro de 2016

Resumo

O que seria de um jogo online se de cada vez que alguém quisesse jogar, tivesse de esperar 1,2 minutos só para que todos os conteúdos da sua conta estivessem disponíveis? E para iniciar um jogo, esperaria minutos só para conseguir saber qual o servidor disponível para se poder conectar?
Nós sabemos que não.
Deste modo, iniciamos o nosso trabalho com um tema do nosso quotidiano e do nosso agrado, jogos online. Não foi difícil levantar os requisitos que, pensamos nós, sustentam uma base de dados de um jogo online, sendo que a nossa inclinação foi para um jogo de tiros em primeira pessoa (FPS).
Após 2 reuniões entre todos, chegamos a um modelo conceptual base, que de seguida mostramos ao professor Orlando Belo de forma a validar o mesmo.
Depois disso, tudo ficou esclarecido na reunião seguinte que tivemos quanto ao que precisava de ser feito de seguida, sendo que limamos arestas quanto aos requisitos que achamos necessários.
Após o modelo conceptual referido acima, o modelo lógico e físico seguiram de imediato sem grandes dificuldades.
Cremos que o nosso esquema de base de dados, no fim, fica bem implementado quanto ao tema em questão, sendo que cobre as necessidades básicas que, aos nossos olhos, servem de funcionamento ao mesmo.

Área de Aplicação: Desenho, Produção, Implementação de uma base de dados
Palavras-Chave: Administração de base de dados relacional, SQL, Diagramas Entidade-Relacionamento, Normalização de modelos relacionais, MySQL

Índice

Índice de Figuras

Índice de Tabelas

1. Introdução

Este trabalho visa a implementação de uma base de dados que sustente o funcionamento e o registo de dados sobre um jogo online.

1.1. Contextualização

A nossa inclinação é mais para uma base de dados de um jogo FPS. Mas com quase nenhuma alteração (ou mesmo nenhuma), os dados de qualquer tipo de jogo online podem ser registados e sustentados por esta base de dados.

Suponhamos que a empresa GameOn (fictícia) nos contactou, de modo apresentar um projeto de um revolucionário jogo do género FPS. Esta empresa trabalha no ramo dos jogos online à apenas 5 anos sendo que pensam eles que este é o momento de dar o salto para maiores palcos, maiores vendas e sobretudo um lugar nos clássicos FPS.

Para isso decidem lançar o jogo referido acima.

1.2. Apresentação do Caso de Estudo

A GameOn pretende que lhe forneçamos uma base de dados que permita o registo e o alicerce do jogo que querem lançar, sendo que repararam eles que a melhor e mais rápida forma de ligar a informação acerca dos vários componentes fora de jogo é através de SQL.

Eles pretendem um sistema que seja capaz de guardar informação acerca dos usuários do jogo, de todos os itens que os mesmos possuem e em que servidor estão a jogar.

Deve haver ainda um lugar onde se guarde as estatísticas de jogo de cada usuários, em que toda a essa informação é dada pela aplicação no fim de cada jogo, sendo que a base de dados, no final, apenas precisa de ter um procedimento que calcule o ranking geral de todos os usuários. Com isto a GameOn pretende atribuir um prémio anual aos 5 usuários com melhor performance no jogo, logo este procedimento apenas vai ser usado uma vez por ano.

Querem ainda saber qual o número de jogadores que estão a jogar e em que servidores, de modo a aperfeiçoarem os mapas e ver quais as tendências mais apetecíveis de jogo.

1.3. Motivação e Objectivos

A GameOn pretende que não se espere minutos só para algum jogador conseguir visualizar o seu inventário. Deste modo, o SQL fornece um alicerce no que toca à rapidez de fornecimento de informação.

A empresa olha também para grandes palcos, e com grandes palcos vêm grandes responsabilidades, e sobretudo, enormes bases de dados. Após um estudo do que havia disponível no mercado, repararam que um motor de base de dados baseado em SQL seria o que de melhor poderiam ter para reunir e acessar informação de forma organizada e rapidamente.

1.4. Estrutura do Relatório

Em suma, em termos de conteúdo real, os próximos 4 capítulos irão ter o sumo do trabalho.

O segundo capitulo faz referência ao levantamento de requisitos que possibilitaram o avanço do trabalho.

Já no terceiro, pegando nestes requisitos, foi criado o modelo conceptual de dados, sendo que o objetivo foi o de cobrir ao máximo todos os requisitos levantados de forma a criar uma base de dados consistente e duradoura.

Para o quarto capitulo, de forma muito simples e direta, foram aplicadas as regras gerais de passagem do modelo conceptual para o modelo lógico, sendo que este capitulo cobre todo esse processo.

Por ultimo, temos no quinto capitulo o modelo físico de dados. Aqui sim está tudo operacional e pronto a trabalhar.

Comecemos então pelo inicio, o levantamento de requisitos que vem já na próxima página.

2. Levantamento de requisitos

Como a empresa GameOn foi inventada por nós, o levantamento de requisitos não foi difícil, sendo que somos nós que somos entrevistados e entrevistadores. Logo não há grande espaço para ambiguidades.
Após uma primeira reunião entre nós, foi discutido tudo aquilo que ao nosso ver é o cerne de um jogo online. Sendo que no fim da mesma cada um ficou com o dever de duvidar do que foi discutido e com isso descobrir potenciais falhas e formas de as resolver.
Na segunda reunião, depois de todos terem pensado bem no tema chegamos às seguintes conclusões sobre os requisitos pedidos:

- A GameOn pretende uma base de dados que é "embebida" no jogo que querem criar, de modo a funcionar como suporte às funcionalidades fora de partida.
- Quando um usuário se quer registar, a aplicação do jogo reúne todas as informações necessárias e a base de dados guarda um novo jogador com essas informações
- Cada usuário tem um conjunto de itens que pode comprar ao longo do tempo, sendo que estes ficam guardados num inventário.
- Cada usuário tem informações estatísticas sobre os jogos feitos, informações estas fornecidas pela aplicação do jogo, que são guardadas de modo a que, quando pedido, haja um procedimento que calcule o ranking de jogadores com essas informações.

3. Modelo conceptual de dados

3.1. Identificar entidades

- **Jogador**

  Um jogador é um usuário do jogo. Este é caracterizado pelo seu nome, idade, n+úmero de telemóvel, correio eletrónico, uma referência de banimento, ou seja, ver se o usuário foi banido por algum motivo do jogo e a palavra-chave da sua conta.

- **Rank**

  Guarda informação que é utilizada para calcular a classificação de cada jogador. Esta informação provém de cada jogo feito, sendo que a aplicação do jogo, no fim do mesmo, atualiza os atributos referentes a cada caso e assim, quando se invoca o procedimento que dá a classificação geral de todos os jogadores obter esses dados a partir de tudo o que está aqui.

- **Item**

  Itens são todos aqueles objetos de jogo que podem ser escolhidos antes do jogo começar. Aqui estão todas as armas, DLC (Downloadable Content), granadas, etc, disponíveis dentro do jogo. Cada utilizador pode adquirir estes itens.

- **Main Server**

  O Main Server é, como o nome indica, o servidor principal da GameOn. Esta entidade vem aqui por facilidade, visto que através dela podemos retirar informação direta sobre o número de jogadores online e os que estão em jogo, uma vez que para poder jogar, cada usuário tem de fazer login no servidor principal.

- **Game Server**

  Aqui é onde a magia acontece. Cada jogo tem um servidor associado de modo a que vários usuários possam jogar simultaneamente, ou seja, aqui é onde o jogo realmente acontece, ou melhor, a base onde o jogo realmente acontece. Este servidor tem um limite de jogadores em jogo ao mesmo tempo.

- **Mapa**

  Nesta entidade procuramos especificar quais os mapas de jogo disponíveis, que podem estar a ser utilizados nos vários jogos a decorrer.

- **Tipo**

  Aqui é especificado o tipo de mapa que esta a ser usado, sendo que a GameOn tem os seus próprios mapas, que são os oficiais, mas podem haver usuários mais criativos que queiram criar o seu próprio mapa de jogo e pô-lo online de modo a que outros usuários possam jogar nele. Assim sendo estes últimos são mapas não oficiais.

3.2. Identificar Relacionamentos

Na seguinte tabela apresentamos os relacionamentos entre todas as entidades que decidimos que o nosso trabalho deve ter. Apresentamos apenas uma tabela visto que só temos relacionamentos binários.

Entidade	**Multiplicidade**	**Relacionamento**	**Multiplicidade**	**Entidade**
Jogador	N	Inventário	N	Item
Jogador	N	Login	1	Main Server
Jogador	N	Joga em	1	Game Server
Main Server	1	Tem	N	Game Server
Game Server	N	Tem	1	Mapa
Mapa	N	Tem	1	Tipo

Tabela 1: Relacionamentos binários do modelo conceptual

3.3. Identificar e associar atributos com entidades

De seguida são apresentados os diferentes atributos que cada entidade possui.

- **Jogador**

  Aquando do registo no jogo é pedido a cada utilizador que forneça alguns dados pessoais.

Atributo	**Descrição**	**Tipo de dados**	**Possibilidade Nulo**	**Composto**	**Derivado**
Id	Número único de Jogador	Integer	Não	Não	Não
Nome	Nome completo do jogador	100 caracteres	Não	Não	Não
Idade	Idade do jogador	Integer	Não	Não	Não
Password	Password da conta do jogador	25 caracteres	Não	Não	Não
BAN Check	Verificação de conta banida	BOOLEAN	Não	Não	Não
Telemovel	Número de telemóvel to jogador	Integer	Sim	Não	Não
Email	Email do jogador	30 caracteres	Não	Não	Não

Tabela 2: Caracterização da entidade Jogador

- **Item**

  Todos os itens disponíveis no jogo são caracterizados pelos atributos desta tabela, sendo que funcionam como uma espécie de extras.

Atributo	**Descrição**	**Tipo de dados**	**Possibilidade de nulo**	**Composto**	**Derivado**
Id	Número de identificação do item	Integer	Não	Não	Não
Nome	Nome do item	30 caracteres	Não	Não	Não
Descrição	Descrição do	150	Sim	Não	Não

	item	caracteres			

Tabela 3: Caracterização da entidade Item

- **Main Server**

Atributo	**Descrição**	**Tipo de dados**	**Possibilidade de nulo**	**Composto**	**Derivado**
Id	Número identificador do servidor	Integer	Não	Não	Não
Jogadores online	Número total de jogadores que estão online	Integer	Não	Não	Sim
Jogadores em partida	Número total de jogadores a jogar	Integer	Não	Não	Sim

Tabela 4: Caracterização da entidade Main Server

- **Game Server**

Atributos	Descrição	Tipo de dados	Possibilidade de nulo	Composto	Derivado
Id	Número identificador do servidor de jogo	Integer	Não	Não	Não
Nome	Nome do servidor de jogo	20 caracteres	Não	Não	Não
Capacidade	Capacidade de usuários conectados total do servidor	Integer	Não	Não	Não
Média Ping	Velocidade de rede média no servidor	Float	Não	Não	Não

Tabela 5: Caracterização da entidade Game Server

- **Mapa**

Atributo	**Descrição**	**Tipo de dados**	**Possibilidade de nulo**	**Composto**	**Derivado**
Id	Número de identificação do mapa	Integer	Não	Não	Não
Nome	Nome do mapa	25 caracteres	Não	Não	Não

Tabela 6: Caracterização da entidade Mapa

- **Rank**

Atributo	**Descrição**	**Tipo de dados**	**Possibilidade de nulo**	**Composto**	**Derivado**
Id	Identificação única	Integer	Não	Não	Não
Disparos	Número de disparos total	Integer	Não	Não	Não
Headshot	Número total de headshots	Integer	Não	Não	Não
Tiros certos	Número de tiros que atingiram o adversário	Integer	Não	Não	Não
Mortes	Número total de vezes morto	Integer	Não	Não	Não
Adversários Mortos	Número total de adversários mortos	Integer	Não	Não	Não

Tabela 7: Caracterização da entidade Rank

- **Tipo**

Atributo	**Descrição**	**Tipo de dados**	**Possibilidade de nulo**	**Composto**	**Derivado**
Id	Identificador único de tipo	Integer	Não	Não	Não
Descrição	Descrição do tipo de mapa (Oficial ou Não Oficial)	11 caracteres	Não	Não	Não

Tabela 8: Caracterização da entidade Tipo

3.4. Determinar atributos candidatos a chave e chaves primárias

De modo a escolher corretamente quais os atributos que podem ser considerados chave, recorremos às dependências funcionais, que são apresentadas nos próximos diagramas de dependencia

- **Jogador**

Dependência	**Tipo**
Id->(Nome, Email, Telemóvel, Password, Idade, BAN Check)	Total
Email->(Id, Nome, Telemóvel, Password, Idade, BAN Check)	Total
Telemóvel->(Id, Nome, Email, Password, Idade, BAN Check)	Total
(Id,Password)->(Nome, Email, Telemóvel, Idade, BAN Check)	Parcial
(Email,Password)->(Id, Nome, Telemóvel, Idade, BAN Check)	Parcial
(Telemóvel, Password)-> (Id, Nome, Email, Idade, BAN Check)	Parcial
(Id,Email)->(Nome, Telemóvel, Password, Idade, BAN Check)	Parcial
(Id,Telemovel)-> (Nome, Email, Password, Idade, BAN Check)	Parcial
(Email,Telemóvel)-> (Id, Nome, Password, Idade, BAN Check)	Parcial
(Id,Email,Telemóvel,Pasword)-> (Idade,BAN Check)	Parcial

Tabela 9: Tabela de dependências da entidade Jogador

- **Rank**

Dependência	**Tipo**
Id->(Disparos,Headshots,Tiros certos,Mortes,Adversários mortos)	Total

Tabela 10: Tabela de dependências da entidade Rank

- **Item**

Dependência	**Tipo**
Id->(Nome,Descrição)	Total
Nome->(Id,Descrição)	Parcial
(Id,Nome)->(Descrição)	Parcial

Tabela 11: Tabela de dependências da entidade Item

- **Game Server**

Dependência	**Tipo**
Id->(Capacidade,Media Ping)	Total

Tabela 12: Tabela de dependências da entidade Game Server

- **Main Server**

Dependência	**Tipo**
Id->(Jogadores Online,Jogadores em Jogo)	Total

Tabela 13: Tabela de dependências da entidade Main Server

- **Mapa**

Dependência	**Tipo**
Id->(Nome)	Total
Nome->(Id)	Total

Tabela 14: Tabela de dependências da entidade Mapa

- **Tipo**

Dependência	**Tipo**
Id->(Descrição)	Total

Tabela 15: Tabela de dependências da entidade Tipo

Com isto é possível de forma fácil identificar as chaves candidatas e dessa forma escolher a que melhor se adequa ao nosso interesse. Com isto a tabela seguinte mostra essa escolha

Entidade	Chaves candidatas	Chave primária	Chaves alternativas
Jogador	Id, Email, Telemóvel, (Id, Password), (Email, Password), (Telemóvel, Password), (Id, Email), (Id, Telemóvel), (Email, Telemóvel), (Id, Email, Telemóvel, Password)	Id	Email, Telemovel
Rank	Id	Id	-
Item	Id, Nome, (Id, Nome)	Id	Nome
Game Server	Id	Id	-
Main Server	Id	Id	-
Mapa	Id,Nome	Id	Nome
Tipo	Id	Id	-

Tabela 16: Descrição da escolha de chaves primárias

3.5. Validação do modelo conceptual com perguntas ao modelo

De modo a verificar se o nosso modelo corresponde ao que é pedido e se é possível obter a informação que necessita de ser guardada elaboramos um conjunto de questões que justificam a validade do nosso modelo conceptual.

- Quero saber que jogadores estão a jogar neste momento.

  Ora para isto basta olhar para o relacionamento entre Jogador e Game Server. Este relacionamento dá informação sobre os jogadores que estão a jogar.

- Quero saber os dados de um certo jogador

  A entidade Jogador guarda toda a informação sobre os dados do jogador

- Quais os mapas que estão a ser utilizados? Qual é o que está a ser mais utilizado?

Basta olhar para o relacionamento entre Game Server e Mapa. Com este relacionamento é possível identificar na entidade Game Server quais os Mapas que estão a ser utilizados e é possível fazer uma contagem de modo a ver qual o mais usado.

- A GameOn quer saber a informação de cada jogador de modo a produzir o ranking geral de jogadores.

  A entidade Rank guarda toda a informação acerca de todos os jogadores, sendo que o relacionamento entre Rank e Jogador faz distinção entre a informação presente na entidade Rank em relação a cada Jogador.

- De modo a aumentar os lucros a GameOn pretende criar promoções de modo a que os itens que são menos comprados possam levar os jogadores a comprá-los.

  Na entidade Item podem ver-se todos os itens disponíveis no jogo, sendo que de modo a verificar quais os que estão a ser menos comprados basta olhar para o relacionamento entre Jogador e Item de modo a fazer uma contagem que identifique os itens que a maior parte dos jogadores não tem.

- De modo a evoluir o jogo e dar oportunidade aos jogadores de mostrar o seu trabalho extra jogo mas em favor do jogo, a GameOn pretende avaliar junto dos seus técnicos quais dos mapas não oficiais têm potencial para se tornarem oficiais. Para isso precisam de saber quais são os mapas não oficiais.

  Entre Mapa e Tipo existe um relacionamento. Na entidade Tipo estão os valores Oficial e Não oficial, logo basta selecionar os registos da entidade Mapa que têm tipo Não oficial.

3.6. Desenho do modelo conceptual

Usando a ferramenta TerraER, apresentamos na figura abaixo o diagrama entidade-relacionamento. Desta forma materializamos de forma gráfica tudo o que foi dito até aqui sobre o nosso projeto, para que desta forma seja demais fácil compreensão.

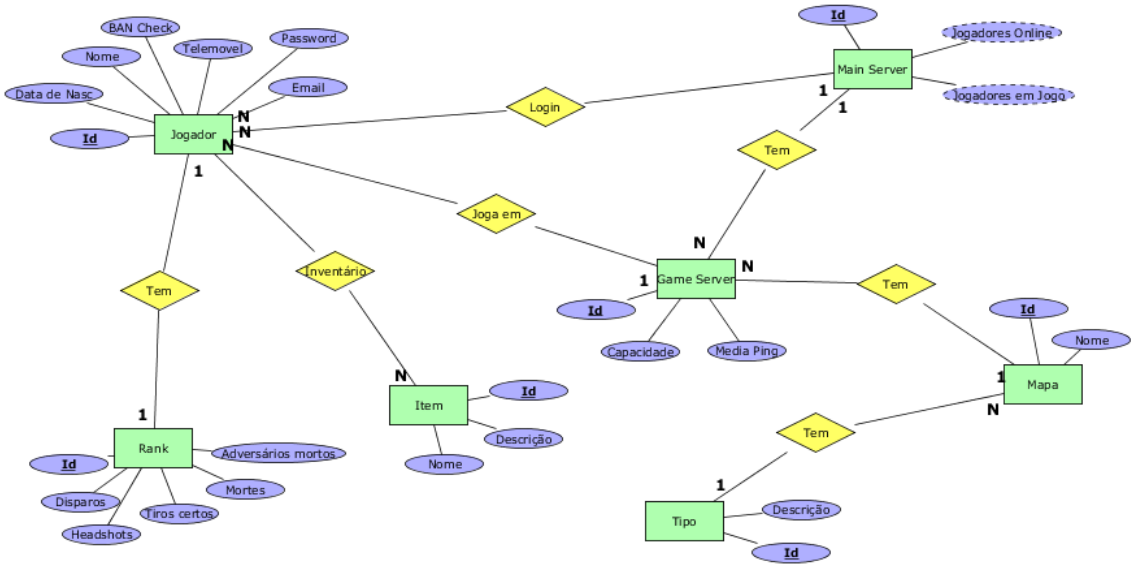

Figura 1: Diagrama Entidade-Relacionamento do modelo conceptual

4. Construção e validação do modelo lógico de dados

4.1. Passagem do modelo conceptual ao modelo lógico

De modo a produzir o nosso modelo lógico de dados, é preciso transformar todos os conceitos do modelo conceptual para o modelo lógico. Assim sendo existem um conjunto de regras que se devem seguir de modo a fazer este trabalho. Tendo o modelo conceptual feito, é quase automático produzir o modelo lógico, seguindo as regras claro.

Assim sendo, criamos a nossa própria convenção quanto ao nome e à forma de escrita de cada relacionamento, entidade e atributo. Cada entidade é escrita da mesma forma que no modelo conceptual, sendo que se a entidade tiver mais do que uma palavra, faz-se a concatenação das palavras constituintes (i.e. retiram-se os espaços) e põe-se a inicial de cada palavra a maiúscula. Claro está, se a entidade só for constituída por uma palavra, esta põe-se com a primeira letra maiúscula. Para os atributos optamos por seguir a mesma regra das entidades, sendo que a única alteração é que a primeira letra é sempre minúscula.

Quanto aos relacionamentos, sempre que existe um relacionamento de N para N, foi criada outra tabela com o nome criado a partir da concatenação das duas entidades, sendo que a primeira letra de cada entidade fica em maiúscula nesta concatenação. Esta nova tabela possui dois campos, um é a chave estrangeira que referencia uma das tabelas, e o outro é a chave estrangeira que referencia a outra tabela. Nos relacionamentos 1 para N, foi criada uma cópia da chave primária da entidade do lado 1, e essa cópia foi colocada na entidade do lado N como chave estrangeira. Por ultimo nos relacionamentos 1 para 1 foi criada uma cópia da chave primária do lado que se considera ser o lado pai, e essa cópia foi colocada no lado considerado filho como chave estrangeira. No nosso modelo existe um relacionamento desta natureza que é entre Jogador e Rank. Aqui considera-se Jogador como o lado pai e Rank como lado filho uma vez que Rank só precisa de estar relacionado com Jogador.

Não tendo mais nenhum tipo de relacionamento apresentamos de seguida o modelo lógico desnormalizado criado no MySQKWorkbench.

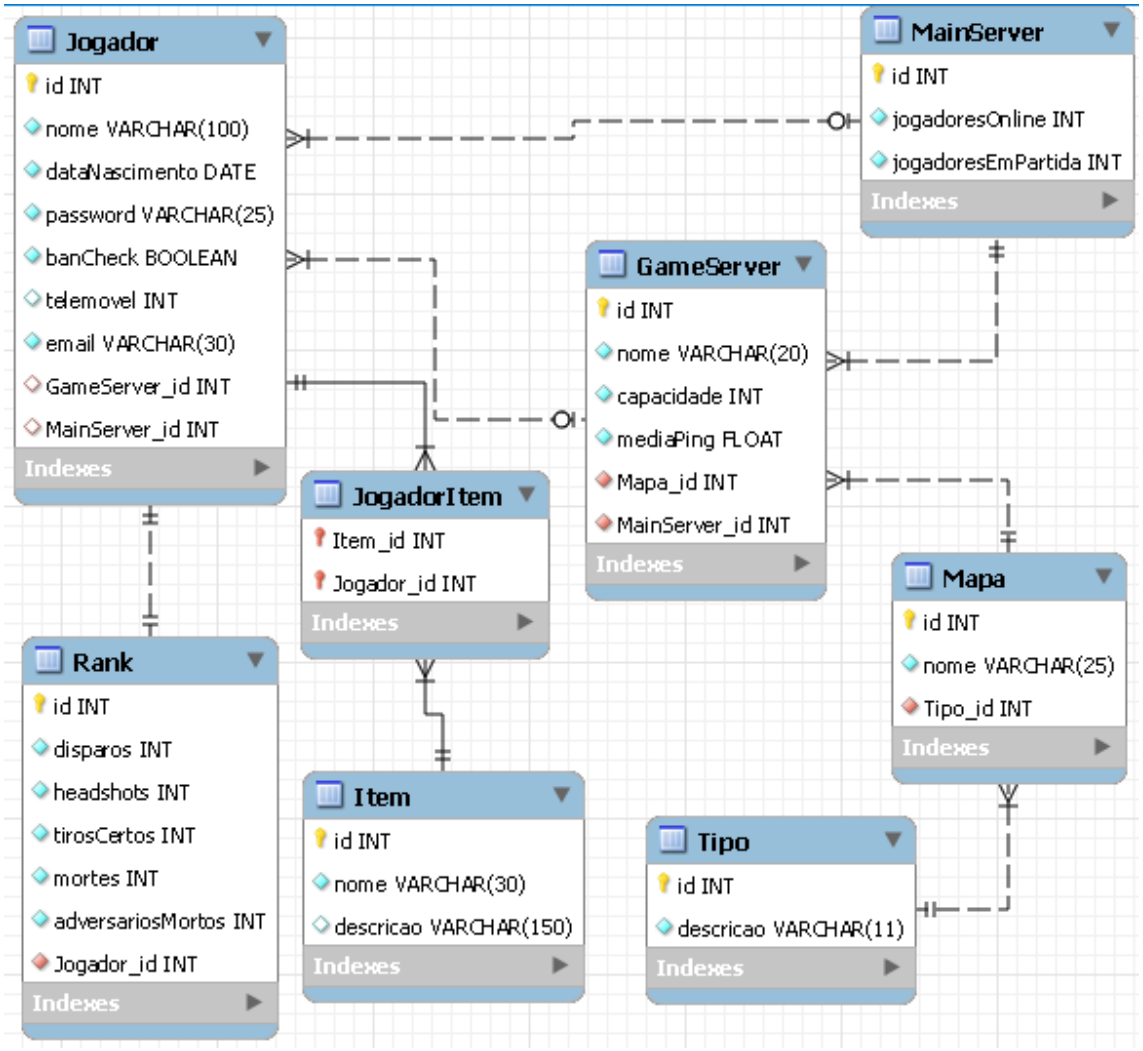

Figura 2: Diagrama do modelo lógico de base de dados

4.2. Normalização

Quanto à normalização do nosso modelo lógico, reparamos que ele está normalizado segundo certas convenções. Para efeitos de normalização não estão a ser considerados valores numéricos e valores booleanos. Isto deve-se ao facto de, sabendo que um booleano em SQL é um TINYINT(1), então estamos sempre a falar de valores numéricos. A razão pela qual "ignoramos" a normalização nestes caso é que, criando uma nova tabela, por exemplo pegando no atributo idade da entidade Jogador, iria gastar-se mais memória. Isto porque esta nova tabela iria ter pelo menos a chave primária (considerando que a

chave primária possa já significar a idade), ou seja iria ser um valor do tipo INT, e na entidade Jogador iriamos ter uma chave estrangeira com referência à tabela Idade. Ora por isto já se viu que no mínimo, desta forma se iriam ter tantos inteiros quanto os registos de Jogador mais tantos inteiros quanto os registos de Idade, o que é claramente maior do que apenas ter tantos inteiros quanto os registos de Jogador.
Há ainda a referir que todos os atributos chamados nome e descrição de todas as entidades que os possuem são sempre considerados diferentes. Isto porque, por exemplo na entidade GameServer, a existirem dois nomes iguais, então está-se a falar do mesmo servidor de jogo, o que não faz sentido pois se um servidor de jogo existe e está criado, então igual a esse não existe mais nenhum.
Deste modo é de fácil verificação que não existem grupos de atributos que se repetem nas entidades que estamos a considerar, e como tal o nosso esquema está na 1ª forma normal.

Quanto à 2ª forma normal, atendendo às tabelas presentes no capitulo 3.4 verifica-se que, para todas as entidades, todos os atributos não chave primária são totalmente dependentes da mesma. Posto isto o esquema está então na 2ª forma normal.

Por fim, falando da 3ª forma normal, verifica-se que o esquema também está na 3ª forma normal, uma vez que em cada tabela não existe nenhum atributo que seja dependente de mais algum atributo além da chave primária da mesma tabela.
Chegado aqui, como o esquema lógico até este ponto esteve sempre normalizado, então consideramos que não vale a pena testar as seguintes formas normais, uma vez que estamos satisfeitos com a implementação atual e não são vislumbradas vantagens e melhoramentos com as próximas formas normais.
Concluímos com tudo isto que o nosso esquema se encontra normalizado.

4.3. Validação do modelo lógico com perguntas ao modelo

Concluindo a implementação do modelo lógico, voltamos a fazer as questões do ponto 3.5 ao modelo produzido. Como a passagem do modelo conceptual ao modelo lógico foi pacífica e seguindo as regras, estas questões naturalmente continuam com a mesma resposta, sendo que não vale a pena reescrever o mesmo que foi escrito no ponto 3.5.

5. Construção do modelo físico de dados

O motor de base de dados utilizado é o MySQL, que aliás é o motor que estamos comummente habituados a trabalhar nas aulas.

5.1. Construção do modelo físico

Neste ponto, para cada tabela considerada no modelo lógico, vamos proceder à especificação da constituição dos seus atributos e possíveis observações quanto a essa tabela. Deste modo, para caracterizar cada atributo iremos dizer o nome do atributo e de seguida todas as características do mesmo.

- **Jogador**

  Jogador(

id	INT, chave primária, não nulo, auto incremento
idade	TINYINT(3), não nulo, sem sinal
nome	VARCHAR(100), não nulo
password	VARCHAR(25), não nulo
ban check	BOOLEAN, não nulo, default=False
telemóvel	INT, sem sinal
email	VARCHAR(30), não nulo
GameServer_id	INT, chave estrangeira, tabela de referência: GameServer, On Update no action, On Delete no action
MainServer_id	INT, chave estrangeira, tabela de referência: MainServer, On Update no action, On Delete no action

)

  Nesta tabela é de notar que se o jogador não estiver online então a chave estrangeira MainServer_id é NULL.

  Do mesmo modo, se o jogador não estiver nalgum servidor a jogar então a chave estrangeira GameServer_id é NULL.

- **Rank**

  Rank(

id	INT, chave primária, não nulo, sem sinal, auto incremento
disparos	INT, não nulo, sem sinal, default=0
headshots	INT, não nulo, sem sinal, default=0
tirosCertos	INT, não nulo, sem sinal, default=0
mortes	INT, não nulo, sem sinal, default=0
adversariosMortos	INT, não nulo, sem sinal, default=0
Jogador_id	INT, chave estrangeira, tabela de referência: Jogador, On Update no action, On Delete no action

)

- **Item**

  Item(

id	INT, chave primária, não nulo, sem sinal, auto incremento
nome	VARCHAR(30), não nulo
descrição	VARCHAR(150)

)

- **JogadorItem**

  JogadorItem(

Item_id	INT, chave estrangeira, tabela de referência: Item, On Update no action, On Delete no action
Jogador_id	INT, chave estrangeira, tabela de referência: Jogador, On Update no action, On Delete no action

)

- **MainServer**

  MainServer(

id	INT, chave primária, não nulo, sem sinal, auto incremento
jogadoresOnline	INT, não nulo, sem sinal, default=0
jogadoresEmPartida	INT, não nulo, sem sinal, default=0

)

- **GameServer**

  GameServer(

id	INT, chave primária, não nulo, sem sinal, auto incremento
nome	VARCHAR(20), não nulo

capacidade INT, não nulo, sem sinal
mediaPing FLOAT, não nulo
Mapa_id INT, chave estrangeira, tabela de referência: Mapa, On Update no action, On Delete no action
MainServer_id INT, chave estrangeira, tabela de referência: MainServer, On Update no action, On Delete no action
)

- **Mapa**

Mapa(
id INT, chave primária, não nulo, sem sinal, auto incremento
nome VARCHAR(25), não nulo
Tipo_id INT, chave estrangeira, tabela de referência: Tipo, On Update no action, On Delete no action
)

- **Tipo**

Tipo(
id INT, chave primária, não nulo, sem sinal, auto incremento
descricao VARCHAR(11), não nulo
)

Note-se que nesta tabela só vão existir 2 registos, um que tem como descrição "Oficial" e outro que tem como descrição "Não oficial". Como tal pôs-se um limite no número esperado de linhas da tabela de 2.

5.2. Construção dos atributos derivados.

No nosso esquema temos, na tabela MainServer, dois atributos derivados. São eles jogadoresOnline e jogadoresEmPartida.

De modo a calcular estes dois valores foram criados dois triggers. Um deles calcula o número de jogadores online, ou seja, é o somatório, na tabela Jogador, de todos os registos com o campo MainServer_id diferente de NULL. Este trigger atualiza o valor de jogadoresOnline.

O segundo calcula os jogadores que estão em jogo, ou seja, é o somatório, na tabela Jogador, de todos os registos com o campo GameServer_id diferente de NULL. Este trigger atualiza o valor de jogadoresEmPartida.

5.3. Povoamento da base de dados

- **Main Server**

```
insert into MainServer (id) values (1);


delimiter $$
create trigger updateJogadoresOnline
        after update on Jogador
  for each row
begin
update MainServer m
        set m.jogadoresOnline =(
                select count(MainServer_id) from Jogador
  );
end$$


delimiter $$
create trigger updateJogadoresEmPartida
        after update on Jogador
  for each row
begin
update MainServer m
  set m.jogadoresEmPartida=(
                select count(GameServer_id) from Jogador
    );
end$$


delimiter $$
create trigger updateJogadoresOnlineOnInsert
        after insert on Jogador
  for each row
begin
update MainServer m
        set m.jogadoresOnline =(
                select count(MainServer_id) from Jogador
  );
end$$
```

```
delimiter $$
create trigger updateJogadoresEmPartidaOnInsert
        after insert on Jogador
  for each row
begin
update MainServer m
        set m.jogadoresEmPartida=(
                select count(GameServer_id) from Jogador
    );
end$$
```

Note-se que aqui foram criados os triggers que vão calcular os atributos derivados da tabela MainServer. Sendo que existem 4 triggers, um que atualiza o valor de jogadoresOnline quando se faz um update na tabela Jogador, que se chama updateJogadoresOnline, e outro que faz o mesmo, mas na inserção de um novo registo na tabela Jogador, que se chama updateJogadoresOnlineOnInsert.
Por ultimo, temos os triggers que atualizam o valor de jogadoresEmPartida, partindo da mesma ideia dos triggers referidos acima e que se chamam updateJogadoresEmPartida e updateJogadoresEmPartidaOnInsert.

- **GameServer**

```
insert into GameServer
        (nome,capacidade,mediaPing,Mapa_id,MainServer_id)
  values
  ("Tea Party",8,20.4,2,1),
  ("Join AXE team",12,15.3,5,1),
  ("Portugal",8,21.4,6,1),
  ("Spain Terror",6,10.8,2,1),
  ("Sniper Association",24,30.2,4,1),
  ("Forces United",8,20.4,4,1),
  ("Desert Operations",8,17.9,7,1),
  ("No man team",12,12.4,5,1),
  ("All round",16,14.3,7,1);
```

- **Tipo**

```
insert into Tipo
        (descricao)
        values
        ("Oficial"),("Não Oficial");
```

- **Mapa**

```
insert into Mapa
        (nome,Tipo_id)
  values
  ("Forest",1),
  ("Futurism",1),
  ("Space Ship",1),
  ("Meteor Crash Site",1),
  ("Clouds",2),
  ("Green field",2),
  ("Desert",1),
  ("City",2),
  ("Dust",1),
  ("Paks",2);
```

- **Jogador**

```
insert into Jogador
        (nome,dataNascimento,Password,banCheck,telemovel,email,GameServer_id,MainSer
ver_id)
  value
  ("DiogoBrandão','2010-0-01',"gdsnvdaewafr",False,912345656,"dbrand@mail.pt",1,1),
  ("John','1980-04-03',"324daewafr",False,95232565,"john@mail.pt",1,1),
  ("Cactus','1998-04-24',"gds443lkj",False,912988756,"vyb@mail.pt",2,1),
  ("leetawp','1997-01-27',"noskill",False,912988756,"leet@mail.pt",2,1),
  ("Duk','1997-01-27',"awpt",False,912988756,"duk@mail.pt",2,1),
  ("Fidalgo','1985-08-12',"3243gjmmvsdan",False,942345560,"fid@mail.pt",3,1),
  ("Asdrúbal','1995-01-18',"09uitora",False,932789560,"asimi@mail.pt",3,1),
  ("Trigo','2000-05-23',"32rtt55mvsdan",False,92388880,"trigo@mail.pt",3,1),
  ("Camper','2000-12-03',"qawsed",False,9258270,"camp@mail.pt",3,1),
  ("Algoritmo','1985-08-14',"mnhyui545",False,987678000,"algo@mail.pt",3,1),
  ("Vyb','1985-08-14',"noskillvyb",False,987634032,"vyb@mail.pt",2,1),
```

("Caloteiro",'1993-01-30',"calotas",False,940000000,"calot@mail.pt",3,1),

("Facadas",'1998-07-25',"knifeh",False,940360015,"knifes@mail.pt",2,1),

("Pato",'1992-10-30',"cctech",False,940000001,"ccpato@mail.pt",3,1);

- **Item**

insert into Item

(nome,descricao)

values

("AK 47","Arma automática de medio alcance"),

("Granada","Granada de mão destruidora"),

("MP7","Submetralhadora"),

("Granada de fumo","Granada de mão que provoca uma nuvem de fumo"),

("Colete de kevlar","Proteção contra balas e pequenos projecteis"),

("RPG","Lança misseis de grande alcance"),

("Espada japonesa","Espada clássica japonesa, efetiva a curto alcançe"),

("AWP","Arma de longo alcance de alto calibre"),

("Desert Eagle","pistola semi-automática de alto calibre"),

("Revolver","Morte certa num curto alcance"),

("Espingarda canos cerrados","Destruição total do adversário"),

("Glock","Pistola semi-automática com modo de rajada"),

("Colt","Arma automática de medio alcance"),

("Taser","Frita-o bem frito com este taser"),

("P90","Submetralhadora"),

("Scout","Arma de longo alcance");

- **Rank**

insert into Rank

(disparos,headshots,tirosCertos, mortes, adversariosMortos, Jogador_id)

value

(139186,9473,64703,27082,32017,1),

(39186,1473,14703,2389,6017,2),

(28912,928,12001,3910,7291,3),

(982317,98073,521703,327582,453491,4),

(739186,89473,624703,241782,473017,5),

(8311,373,5416,1377,2392,6),

(273,27,98,19,46,7),

(82038,1003,52390,12325,24703,8),

(8312,374,5417,1377,2393,9),

```
(8312,373,5418,1379,2392,10),
(982321,98088,521891,347582,423491,11),
(139486,8473,71703,33082,37017,12),
(787186,49473,524703,211782,443017,13),
(102712,763,5218,979,1562,14);
```

- **JogadorItem**

```
insert into JogadorItem
        (Item_id,Jogador_id)
  values
  (1,1),(1,3),(1,6),(1,7), (1,8),
  (2,4), (2,2), (2,3),(2,6),
  (3,1),(3,6),(3,7),(3,16),(3,15),
  (4,1), (4,5),(4,11),(4,13),(4,12),
  (5,14), (5,13), (5,1),(5,16),(5,4),
  (6,1),(6,4), (6,5),(6,6),(6,7),(6,10),
  (7,13), (7,14), (7,15),(7,16),(7,10),
  (8,13),(8,10),(8,12), (8,9),
  (9,13), (9,4),(9,7),(9,8), (9,9),
  (10,16),(10,11),(10,6), (10,8), (10,7),
  (11,14),(11,12),(11,13), (11,6), (11,7), (11,16),
  (12,4), (12,6), (12,9),(12,2),
  (13,16),(13,14),(13,12), (13,11),
  (14,16), (14,1), (14,15),(14,2), (14,3);
```

- **Procedimentos auxiliares**

  De modo a calcular o ranking geral de jogadores foram criados dois procedimentos, um para calcular o ranking de todos os jogadores e outro para calcular o ranking dos N primeiros jogadores, sendo que RankingGeral() calcula o primeiro e RankingGeral(N) calcula o segundo.

```
delimiter $$
create procedure RankingGeral()
begin
select
Jogador.nome,(((r.adversariosMortos/r.mortes)+(r.tirosCertos/r.disparos))+(r.headshots*0.3))
mod 100 as Pontuacao from Jogador
        inner join Rank as r
```

```
   on Jogador.id=r.Jogador_id
   order by Pontuacao desc;
end $$

delimiter $$
create procedure RankingGeralLimitado(in n INT)
begin
select
Jogador.nome,(((r.adversariosMortos/r.mortes)+(r.tirosCertos/r.disparos))+(r.headshots*0.3))
mod 100 as Pontuacao from Jogador
            inner join Rank as r
   on Jogador.id=r.Jogador_id
   order by Pontuacao desc
   limit n;
end $$
```

6. Conclusão

Consideramos que a nossa implementação do tema a que nos propusemos é satisfatória e cobre grande parte da matéria dada.

Ao longo do período de trabalho fomos gerindo entre todos o trabalho necessário à passagem para as fases seguintes, de modo que como grupo, funcionou tudo bem.

A principal dificuldade talvez tenha sido criar o modelo conceptual, pois como foi um tema à nossa escolha houve muito aso a criatividade, o que levou a que houvessem algumas diferenças entre o que seria realmente essencial incluir nos modelos a construir. Posto isso repara-se que a matéria não foi problema, todos foram assimilando ao longo do semestre e o resultado viu-se nos momentos em que era preciso algum conhecimento mais técnico em relação a algum ponto do trabalho.

A GameOn ficou satisfeita com o nosso trabalho e convidou-nos logo para produzir o próximo trabalho do género que estão a pensar fazer, de forma que isso nos deixa contentes ao ver o esforço recompensado.

Em termos de melhorias no trabalho, claro que haveria muitas. Se pudéssemos por tudo o que envolve uma base de dados do género da que fizemos, esta ficaria muito grande e não é isso que este trabalho almejava. Pela nossa experiencia percebemos que este trabalho primava pela qualidade do que se fazia e não pela quantidade de coisas que se punha nos nossos modelos. É a velha ideia de que se se sabe fazer para um caso, então podem vir 4 ou 5 que também se saberá fazer pois nesta cadeira, e no SQL em geral, as coisas fazem-se sempre pelo mesmo caminho. Pode-se inventar um pouco e desviar do caminho, mas a solução para problemas idênticos será sempre através das mesmas ferramentas.

Posto isto é da nossa satisfação concluir este trabalho e dizer que foi do nosso maior agrado aprender tudo o que está relacionado com o SQL, pois a nosso ver, tudo isso é muito poderoso em termos de bases de dados.

7. Ferramentas utilizadas

- TerraER, para o modelo conceptual
- MySQLWorkbench, para a implementação lógica da base de dados e para a criação do script de implementação da base de dados física.
- Microsoft Word 2013, seguindo o template fornecido, foi usada esta ferramenta para a produção do relatório.
- Microsoft Power Point 2013, para a criação da apresentação digital do trabalho
- MySQL server, para a implementação final da base de dados física.

8. Bibliografia

- Database Systems: A Practical Approach To Design, Implementation, and Management, 4ª edição, 2005, Thomas Connolly, Carolyn Begg