**Modelo físico**

Nesta etapa é efetuada a tradução do modelo lógico desenvolvido anteriormente para um modelo físico a ser implementado num SGBD. Esta etapa é também desenvolvida no sentido de otimizar o modo como os dados são guardados e acedidos. Existem diversas estruturas de armazenamento dados. Ao passo que algumas delas são extremamente eficientes na tarefa de guardar dados, outras são mais eficientes no modo como os dados são acedidos. A escolha das estruturas de armazenamento é naturalmente condicionada pelo SGBD. O modelo de dado físico deve ser totalmente dirigido para a natureza dos dados e aquilo que se pretende realizar com eles. Destacam-se os seguintes passos a tem conta na transformação para modelo físico:

* Tradução do modelo lógico para o SGBD
* Organização dos ficheiros e índices
* Desenho das vistas dos utilizadores
* Mecanismos de segurança
* Monitorizar e afinar o sistema operativo

**Tradução do modelo lógico para o SGBD escolhido**

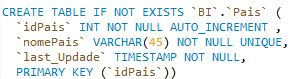
Neste passo foi decidido como representar as relações base identificadas no modelo lógico, de modo a que as relações e as suas restrições possam ser suportadas pelo SGBD escolhido. Para tal ser possível os três passos seguintes terão de ser corretamente efetuados: desenhar as relações base, desenhar as representações dos dados derivados e desenhar as restrições gerais (de negócio), que descreveremos nos pontos seguintes.

**Desenho das relações base**

Inicialmente é preciso colocar à disposição toda a informação sobre as relações apresentadas no modelo lógico de dados. Informação que diga respeito a domínios, valores por defeito, valores nulos e todo o tipo de restrições devem ser apresentados nesta fase. Para cada relacionamento, e de forma muito parecido com o que foi feito para o modelo lógico, será apresentada a informação necessária, mas agora tendo em consideração o SGBD escolhido.

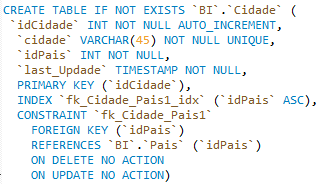
**Relação País**

Em ordem a evitar redundância de dados e ter um maior controlo sobre os dados foi criado a relação país que contem os seguintes atributos:

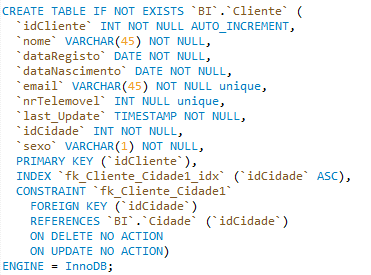


**Relação Cidade**

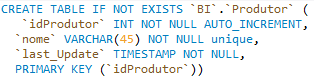
Pelo mesmo motivo que a relação país foi criado uma relação cidade que contém os seguintes atributos:



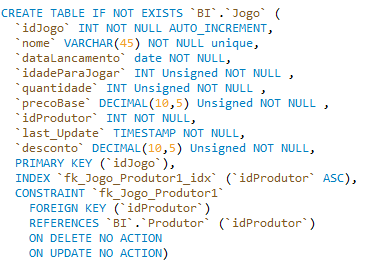
**Relação Cliente**



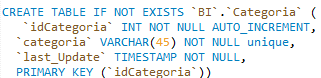
**Relação Produtor**



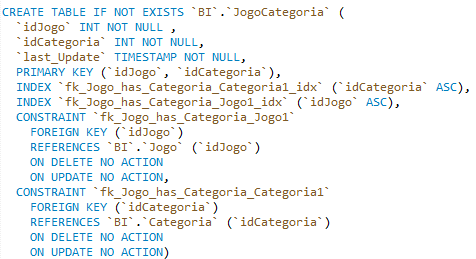
**Relação Jogo**



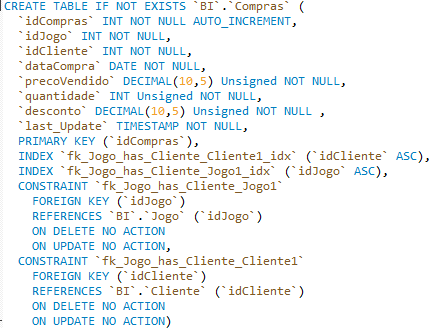
**Relação Categoria**



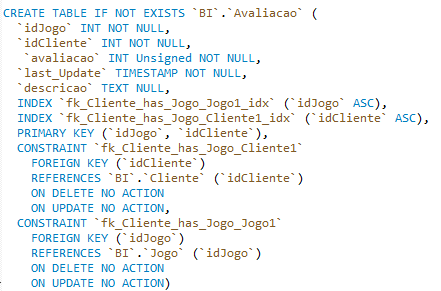
**Relação JogoCategoria**



**Relação Compras**



**Relação Avaliação**



**Desenho das representações dos dados derivados**

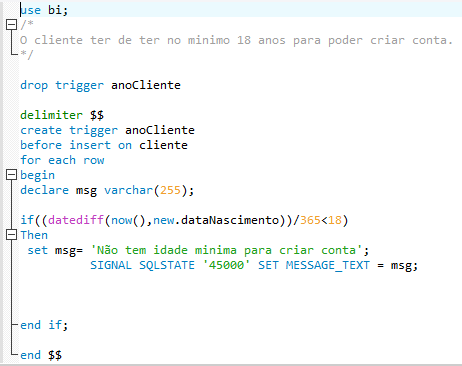
**Desenho das restrições gerais**

As soluções encontradas podem ser simplesmente implementadas com SQL standard ou utilizando mecanismos específicos do SGBD como por exemplo, um *trigger*. A escolha da solução vai depender muito das capacidades do SGBD e caso alguma das restrições não possam ser implementadas pela base de dados, a solução será implementar na aplicação. De seguida, cada uma das restrições irá ser apresentada e a sua respetiva solução.

Nesta fase são definidas as restrições gerais (ou regras de negócio) que servem para garantir a coerência no "mundo real" dos dados armazenados.

**Relação utilizador:**

O utilizador para se registar tem de possuir uma idade superior a 18 anos.



O email e o número de telemóvel serem únicos. Para isso recorreu-se a *constraint unique* que irá garantir que o email e o numero de telemóvel únicos para cada utilizador.

**Relação País**

Como não podem existir dois países com o mesmo nome usou-se a *constraint unique* para garantir que cada país possui um nome diferente.

**Relação Cidade**

Como não podem existir duas cidades com o mesmo nome usou-se a *constraint unique* para garantir que cada cidade possui um nome diferente.

**Relação Produtor**

Como não podem existir dois produtores de jogos com o mesmo nome, usou-se a *constraint unique* para garantir que cada produtor possuiu um nome diferente.

**Relação Jogo**

O nome de cada jogo tem de ser diferente, ou seja, não pode existir dois jogos com o mesmo nome.

A quantidade de um jogo nunca pode ser negativa.

O preço nunca pode ser negativo.

A idade para jogar nunca pode ser negativa.

O desconto em que o jogo se encontra não pode ser negativo.

Para a primeira restrição usou-se a *constraint unique* e assim garantiu-se que dois jogos não possuem o mesmo nome.

Para as restantes como o *mysql* permite que os tipos numéricos sejam *Unsigned* então acrescentamos essa cláusula, permitindo assim que todas as quantidades preços e idades sejam igual ou superior a zero.

**Relação Categoria**

Como não pode existir duas categorias com o mesmo nome usou-se a *constraint unique* e assim garantiu-se que duas categorias não possuem o mesmo nome.

**Relação Compras**

A quantidade não pode ser negativa;

O preço de venda não pode ser negativo;

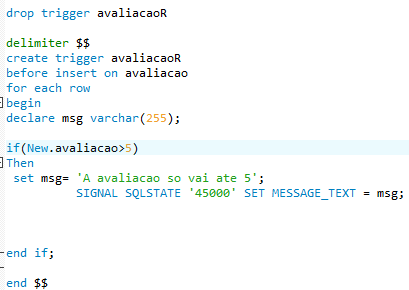
O desconto não pode ser negativo;

Para evitar que a quantidade, o preço e o desconto fossem negativos acrescentamos a cláusula *Unsigned,* visto tratarem-se de tipos numéricos*.*

**Relação Avaliação**

Como não queremos que a avaliação seja negativa então acrescentamos a cláusula *Unsigned*.

A avaliação não pode ser maior que 5, para isso foi criado um *trigger* para que a avaliação não exceda o número 5*.*



**Organização dos ficheiros e índices**

Esta secção aborda a forma como estão guardados os ficheiros que constituem o sistema de base de dados. Falaremos também acerca dos índices que foram criados no sentido de aumentar a *performance* do sistema.

**Análise de transações**

Nesta secção iremos analisar mais detalhadamente as transações realizadas.

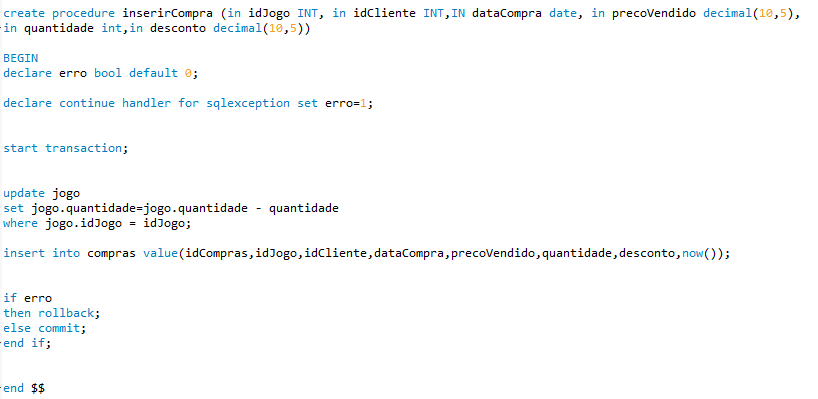
Considere as transações em baixo:

* Inserir a compra de um jogo feita por um cliente e atualizar a quantidade disponível do jogo;
* Inserir um jogo e uma categoria referente ao jogo;

**Comprar um jogo por parte de um cliente**

A entidade jogo é uma das entidades chaves neste universo. Deste modo, sempre que é registada uma compra é inserido um registo na tabela Compras. Como é óbvio, a compra resulta numa diminuição da quantidade do jogo, logo, é necessário que se atualize a tabela Jogo removendo a quantidade comprada.

Transação em questão:



**Inserir um jogo e a sua categoria**

Como um jogo tem sempre uma categoria associada, então é natural que sempre que inserimos um jogo temos de inserir na tabela jogoCategoria a relação existente entre o jogo e a categoria.

Transação em questão:



Pelo que podemos espectar, as tabelas mais acedidas serão as entidades do modelo conceptual assim como as tabelas que derivam na transição do modelo conceptual para o modelo lógico, em especial, as dos relacionamentos de muitos para muitos, tais como:

* Compras;
* Jogo;
* Jogo\_Categoria;

Estas tabelas estarão em constante atualização, então para cada um dos registos estará associado um *TIMESTAMP* que determina a data da ultima modificação.

Em relação a tabela Compras cada registo terá a si associado uma data para determinar a data de cada compra.

**Escolha da organização os ficheiros**

A organização dos ficheiros que constituem a base de dados em armazenamento secundário determina significativamente a eficiência no acesso aos dados. A Base de Dados foi criada em *MySQL* pelo que é o motor da mesma que decide a forma de organizar a informação.

O motor usado foi o *InnoDB* que mantém a sua área de *buffer* para armazenar dados e índices em memória principal. A forma como os ficheiros estão organizados acaba por ser ainda mais eficiente do que mapeamento (*hashing*). Para além disso, o motor disponibiliza meios de suporte às transações (ACID – Atomicidade, Consistência, Isolamento e Durabilidade) e para chaves estrangeiras.

**Desenho das vistas dos utilizadores**

Uma das questões importantes no desenvolvimento de um Sistema de Gestão de Base de Dados é a restrição das vistas de cada utilizador. A nossa implementação tem em conta que existirão N tipos de utilizadores:

* Administrador – Tem acesso a toda a Base de Dados e pode executar todas as ações que entender, podendo também apresentar conclusões sobre os dados apresentados para depois apresentar ao gerente da loja;
* Cliente – Poderá consultar os jogos disponíveis e toda a informação relativa aos menos, poderá também consultar todas as suas compras efetuadas assim como as suas avaliações;

Em baixo apresentamos, para as várias vistas possíveis, o código SQL respetivo.