**Normalização da Base de Dados**

O processo de normalização, é um passo fundamental para a construção de um sistema de base de dados rigoroso.

Para isso, resolvemos fazer um povoamento *ad hoc* (feito numa folha de cálculo) das tabelas da nossa base de dados para nos ajudar a seguir os quatro primeiros passos (exectuando a normalização avançada) do processo de normalização (*Connolly, T., Begg, C., Database Systems, A Practical Approach to Design, Implementation, and Management, Addison-Wesley, 4ª Edição, 2004).*

1. **Unormalized Form (UNF)**

Ao chegarmos ao processo de normalização, tínhamos já concluído o modelo conceptual do nosso sistema, bem como um primeiro esboço do modelo lógico que foi alvo de um povoamento *ad hoc*, cujo resultado se apresenta de seguida (colunas a vermelho correspondem a chaves primárias e a azul a chaves estrangeiras):

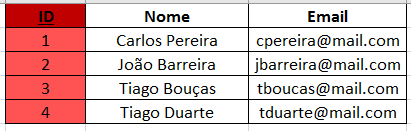


Tabela 1 - Cliente

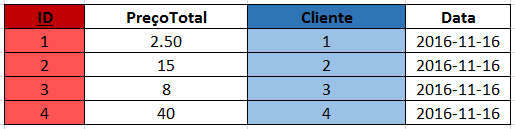


Tabela 2 - Reserva



Tabela 4 - Viagem

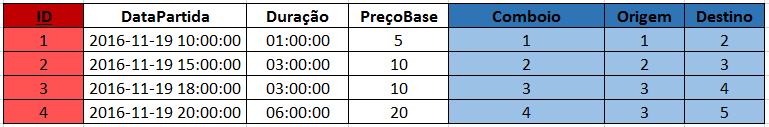


Tabela 3 - Bilhete

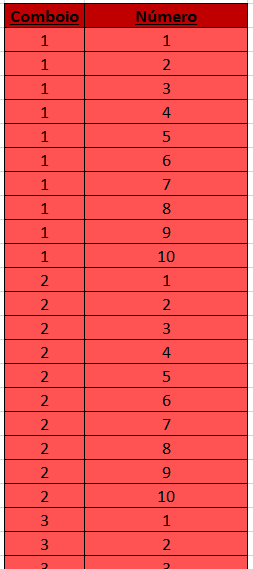


Tabela 5 - Estação

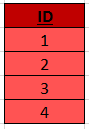


Tabela 6 - Comboio

Tabela 7 - Lugar (não está na imagem: 10 lugares tanto para o comboio 3 como para o 4)

1. **First Normal Form (1NF)**

* **A interseção entre cada linha e cada coluna (i.e. cada célula) deve conter um e um só valor.**

O nosso esquema inicial já obedecia a esta regra, pelo que não foi necessário alterar nada.

1. **Second Normal Form (2NF)**

* **Todos os atributos não primários de uma tabela devem ser (totalmente) funcionalmente dependentes da chave primária dessa tabela**.

Todas as tabelas do nosso esquema inicial obedecem a esta regra:

* + **Cliente** – *nome* e *email* são atributos (totalmente) funcionalmente dependentes de cada cliente (ID).
  + **Reserva** – *preço total*, *cliente* e *data* são atributos funcionalmente (totalmente) dependentes de cada reserva (ID).
  + **Bilhete** – *classe*, *preço* e *reserva* são atributos funcionalmente (totalmente) dependentes de cada bilhete (*lugar* e *viagem*). Além disso, nenhum dos atributos é funcionalmente dependente apenas de um dos dois atributos da chave primária composta.
  + **Viagem** – *data de partida*, *duração*, *preço* *base*, *comboio*, *origem* e *destino* são atributos (totalmente) funcionalmente dependentes de cada viagem (ID).
  + **Estação** – *nome* e *país* são atributos (totalmente) funcionalmente dependentes de cada estação (ID).
  + **Comboio** –esta tabela não tem quaisquer atributos que não a chave primária (i.e. não tem atributos não primários).
  + **Lugar** –esta tabela não tem quaisquer atributos que não a chave primária composta (i.e. não tem atributos não primários)

1. **Third Normal Form (3NF)**

* **Nenhum atributo não primário deve ser transitivamente dependente em relação à chave primária (i.e. todas as colunas de uma tabela devem apenas poder ser determinadas pela coluna da PK e não por qualquer outra coluna da tabela).**

A única tabela que desobedecia a esta critério era a tabela *Bilhete*, visto que a coluna relativa ao *desconto* associado à *classe* do bilhete, deduzia-se pela classe do bilhete e não pelo bilhete em si. Isto é, a coluna *desconto* era transitivamente dependente em relação à chave primária através da coluna *desconto*.

O que resolvemos fazer foi manter a coluna relativa à classe (e.g. Sénior, Estudante, etc) e verificar que não seria necessária a coluna *desconto* se criássemos um mecanismo através de um *trigger* que sempre que era adicionado um novo bilhete, era também feito um teste de casos para determinar a classe e calcular o preço final respetivo (*preçobase \* desconto*).

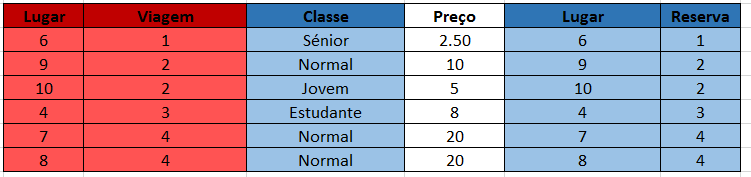
A nova classe Bilhete ficou, então, da seguinte forma:

Tabela 3 (v2) - Bilhete