Normalização

A normalização, no caso do setor de migração, não seria necessária. Pois teríamos de relacionar os dados, como eu fiz posteriormente, para que as informações ficassem completas, sem haver um voo sem aeroporto ou empresa.

Mas, para fins educacionais, ela foi realizada em 6 passos. Onde “separação” significa, de fato, separ a tabela com foco da principal, desvinculando-as.
Os 6 passos são:

1. Separação dos aeroportos de origem;
2. Separação dos aeroportos de destino;
3. Unificação das tabelas de aeroporto;
4. Separação das empresas;
5. Separação dos voos;
6. Criação dos relacionamentos entre as tabelas.

Primeiro passo: A separação basicamente consiste, como também nas próximas, em remover, da tabela principal, as colunas que dizem respeito a nova tabela. Que neste caso são:

- aeroporto_de_origem_sigla
- aeroporto_de_origem_nome
- aeroporto_de_origem_uf
- aeroporto_de_origem_regiao
- aeroporto_de_origem_pais
- aeroporto_de_origem_continente

Após toda separação de tabelas, realizei uma remoção de dados duplicados e renomeando colunas, para que condizessem com a nova tabela. Afinal, os prefixos dos nomes das colunas de origem são utilizados para identificar ao que diz respeito.

Segundo passo: Esta separção é igual à anteiror, apenas trocando o prefixo “aeroporto_de_origem” para “aeroporto_de_destino”.

Terceiro passo: Este é importante para que não seja necessário ter duas tabelas contendo o mesmo tipo de informação, e até o mesmo registro. O que duplicaria o trabalho, visto que seria necessário relacionar a tabela principal com mais uma. E neste caso sem poder misturá-las, pois por mais que carreguem os mesmos dados, estão em ordens diferentes, então o valor **x** de *id* 10 na tabela **a** não será o mesmo na tabela **b**.

Como ambas contem as mesmas informações, é mais prático unificá-las. Realizada antes da criação dos relacionamentos, evita o problema citado anteriormente, pois só haverá um lugar para consutar aquele tipo de dado.

Para unificá-las utilizei o *DataFrame.merge* com *reindexação* aplicada e colunas renomeadas. Após a *reindexação*, feita na forma padrão, os valores de *indexação* anteriores são salvos em uma coluna chamada *index*, que sempre renomeio para “id” concatenado à sigla da tabela (Tendo a tabela **aeroporto** como exemplo: **id_aeroporto**).

Quarto passo: É aplicado tal quais os passos um e dois. Porém, com a *reindexação* da tabela, remoção de duplicados e nova nomenclatura.

Quinto passo: Separação das colunas referentes ao voo em si, sem dados de aeroporto ou empresa, como: **passageiros**, **assentos**, **horas_voadas** e **bagagem_kg**. Também aplicando as normalizações já citadas.

Sexto passo: Relaciona os dados de voos com os aeroportos de origem e destino e a empresa responsável por ele.

Para relacionar as tabelas entre si, tive de utilizar colunas únicas, ou “quase únicas”. Tais colunas devem conter dados que não se repítam, o que não aconteceu em todos os casos. Por isso utilizei pelo menos duas colunas para identificar os registros.

A tabela voo não possui nada que a relacione com as outras tabelas, portanto utilizo o *DataFrame* original, onde está tudo naturalmente relacionado, para conseguir identificar os registros relacionados à cada voo.

Em ambos relacionamentos utilizei a sigla, tanto da empresa quanto do aeroporto, e seu nome. Como dito anteirormete, houveram siglas iguais, que consegui diferenciar baseando-me nos nomes, pois eles tinham pelo menos uma variação.

Sendo capaz de identificar os registros com sucesso, apenas extraio o valor coluna de indentificação, antigo valor de *index* da tabela original, e salvo, na tabela de voos, em uma nova coluna, com o mesmo nome da coluna extraida.

Tendo feito isso, possuo 4 *DataFrames*:

- **Original:** Utilizado para extrair e relacionar os dados;
- **Voo:** Armazena os voos registrados junto ao *id* do aeroporto e empresa;
- **Empresa:** Contém os registros de empresas;
- **Aeroporto:** Possuí os aerorpotos utilizados para voo.