Exercício_1_17 – Consolidar consultas (acrescentar)[1]

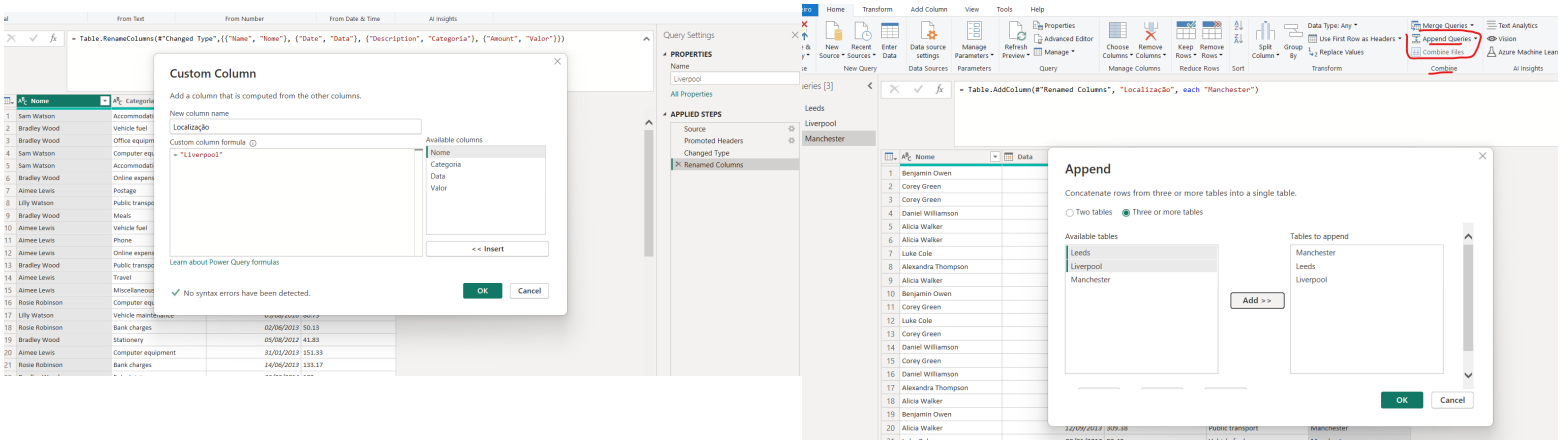
No "**Power BI**" existe um recurso, bastante útil, que permite juntar linhas com dados correspondentes: "**Append Queries**".

Comecemos por importar os dados armazenados nos ficheiros CSV guardados na pasta "**exercicio_1_17**". Faça um duplo clique no ficheiro "**Leeds.csv**", carregando os seus dados no "**Editor do Power Query**". Seguidamente, dentro do "**Editor do Power Query**", clique em "**Home → New Source → Text/CSV**" e clique duas vezes em "**Liverpool.csv**" para importar o ficheiro. Repita o processo para carregar "**Manchester.csv**".

Os três ficheiros, acabados de importar, contêm, no essencial, o mesmo tipo de dados. Devemos começar por remover as diferenças entre as colunas, fazendo com que tenham os mesmos cabeçalhos e o mesmo tipo de dados nas três consultas. Começando o processo com a consulta "**Leeds**", vamos atribuir as seguintes designações às colunas: "**Nome**"; "**Data**"; "**Valor**" e "**Categoria**". Seguidamente vamos alterar as consultas "**Liverpool.csv**" e "**Manchester.csv**", em conformidade. Na consulta "**Liverpool**", somente, será necessário renomear as colunas. Na consulta "**Manchester**" será necessário renomear as colunas e remover a coluna "**Date Paid**" (não existe nas outras consultas).

Poderá ser útil acrescentar uma coluna que identifique o local em que cada pessoa trabalha. Portanto, adicionaremos uma coluna "**Localização**" a cada uma das três consultas. Faremos isto, seguindo o caminho "**Add column → custom column**" e atribuindo a designação "**Localização**" à nova coluna com o texto "Leeds", "Liverpool" ou "Manchester", conforme o caso. Tratadas as consultas, podemos acrescentar uma nova consulta que faça a sua consolidação, seguindo o seguinte caminho "**Home → (combine) → append queries → append queries as new**".

Concluído o exercício, grave o ficheiro criado no "**Power BI**" com a designação "**exercicio_1_17_R.pbix**".



[1] Gamble, G. (2018). *Power BI Step-by-Step Part 3: Power Query, Parameters, Templates & Custom Functions*

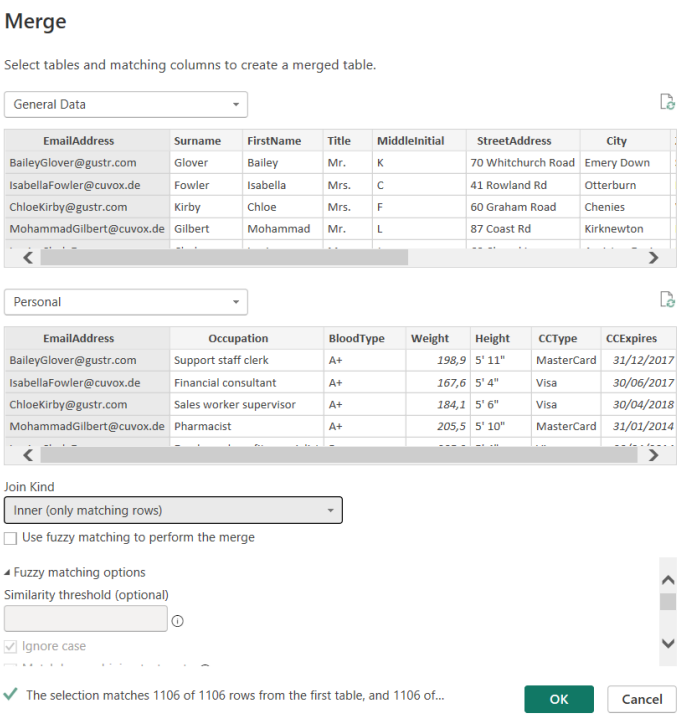
Exercício_1_18 – Consolidar consultas (intercalar)[2]

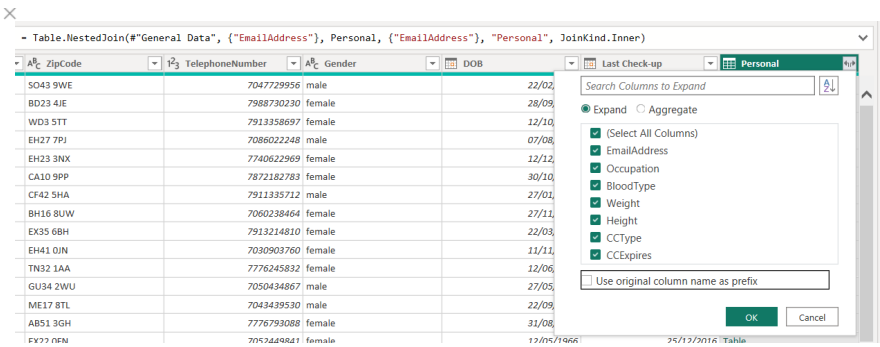
No exercício anterior, vimos como o recurso "**append queries**" pode ser usado para juntar linhas de dados de diferentes consultas. Neste exercício, veremos um outro recurso similar, "**merge queries**", que é usado para juntar colunas de dados de diferentes consultas.

Vamos considerar que temos duas fontes de dados, uma contém informação pública e outra contém informação confidencial. Neste caso, temos dados sobre utentes de uma unidade de saúde: no ficheiro "**Data.xlsx**" temos dados públicos e no ficheiro "**Personal.xlsx**" temos dados privados. Pretendemos reunir estes dois ficheiros que contêm dados sobre os mesmos indivíduos, identificados pela coluna de endereço de e-mail. A utilização do recurso "**merge queries**" carece de uma coluna comum a ambas as consultas. Após tratar os dados importados e renomear as consultas de acordo com o tipo de dados, podemos proceder à sua intercalação, seguindo o seguinte caminho "**Home → (combine) → merge queries → merge queries as new**".

Selecionada a consulta "Dados Públicos", na primeira caixa de diálogo será apresentada esta consulta, sendo agora necessário especificar que o endereço de e-mail é a coluna com a qual queremos fazer a correspondência (coluna comum a ambas as consultas). Isto é feito clicando simplesmente na coluna respetiva. De seguida, indicamos a segunda consulta, "Dados Privados", com a qual pretendemos fazer a intercalação e indicamos a coluna com a qual queremos fazer a correspondência (endereço de e-mail). Neste caso, pretendemos apenas que as colunas correspondentes de ambas as consultas sejam intercaladas. Assim, escolhemos a opção **inner** (*only matching rows*). Finalmente, clicamos em OK e é criada uma nova consulta que contém os dados públicos e uma coluna com os dados privados que é necessário expandir, desativando a opção "*use original column name as prefix*", que neste caso não tem utilidade. Também podemos desativar a coluna do endereço de e-mail ou então eliminá-la à posteriori, depois de confirmar que a intercalação foi bem realizada.

Concluído o exercício, grave o ficheiro criado no "**Power BI**" com a designação "**exercicio_1_18_R.pbix**".





Exercício_1_19 – Agrupar dados[3]

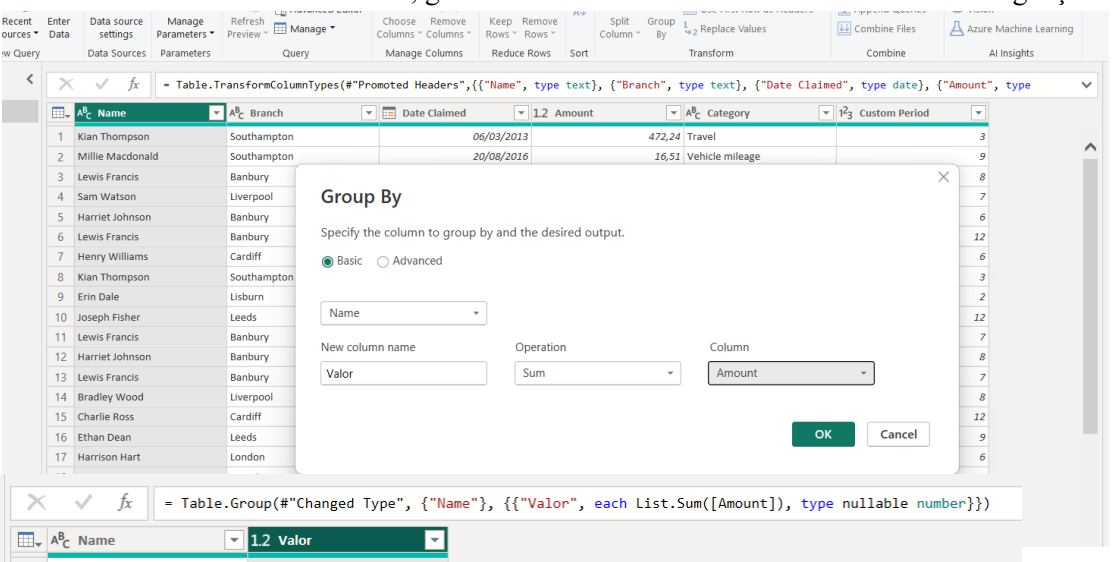
Quando utilizamos fontes de dados que possuem muitas linhas, e mais linhas do que são realmente necessárias, pode ser útil agregar os dados antes de proceder à sua importação. O comando *Group by* do Query Editor permite-nos fazer isso.

Vejamos como podemos utilizar o recurso indicado anteriormente, quando ligamos a uma pasta de trabalho do Excel. Seguindo o caminho "**Home → get data → Excel workbook**" naveguemos até à pasta "**exercicio_1_19**" e façamos um duplo clique em "**All Expenses Data.xlsx**" e entremos no "**Power Query Editor**". Os dados guardados neste ficheiro são relativos ao detalhe de cada reclamação feita aos diversos gestores de conta. O nível de granularidade apresentado pode não ser necessário, sendo suficiente conhecer o total reclamado a cada gestor de conta. Assim, os dados podem ser agrupados pela coluna "**Name**" em que é mostrado o valor reivindicado junto de cada um dos gestores de conta.

Para usar o recurso de agrupar dados, basta destacar a coluna "**Name**" e clicar em "**Home → group by**". Como selecionamos a coluna "**Name**", o nome é mostrado como grupo por coluna. Seguidamente, vamos especificar o nome da nova coluna, pode ser "**Valor**", e a operação que queremos é somar valores da coluna "**Amount**".

Finalmente, após validar o agrupamento podemos ver que, agora, temos apenas uma linha para cada indivíduo e o valor total que foi reivindicado.

Concluído o exercício, grave o ficheiro criado no "**Power BI**" com a designação "**exercicio_1_19_R.pbix**".



	Name	Valor
1	Kian Thompson	20036,46
2	Millie Macdonald	21347,55
3	Lewis Francis	20211,8
4	Sam Watson	18808,93
5	Harriet Johnson	24159,31
6	Henry Williams	21620,34
7	Erin Dale	24124,68
8	Joseph Fisher	17485,71
9	Bradley Wood	20657,05
10	Charlie Ross	22477,88
11	Ethan Dean	16313,61
12	Harrison Hart	22792,52
13	Benjamin Owen	18618,83
14	Jacob Spencer	19989,71
15	Noah Johnson	17431,56
16	Ethan Wells	17820,22

& Custom Functions.