Data Cleaning with Open Refine

LAB COURSE NOTES & EXERCISES

Pere-Pau Vázquez | Data Visualization | 2022

Índex

1. Introduction

Open Refine is a free open source tool designed to clean data. It was born as Freebase Gridworks (and later Google Refine) is a tool for performing different tasks known as data wrangling. Essentially, all data sources commonly have some defects, and before working with them, it is necessary to remove them.

As a tool, it has the look and feel of a spreadsheet, but it behaves more like a database since complex operations such as filtering or clustering can be applied. However, everything happens in a spreadsheet layout, where the data is arranged in rows and columns.

Open Refine can be downloaded from https://openrefine.org/, and the webpage also contains several videos that help users understand the features. This tutorial does not assume that the readers have watched the videos. However, it might be interesting to give them a try before starting with the exercises.

1.1 TUTORIAL OBJECTIVES

The goal of this tutorial is to give the reader a gentle introduction to Open Refine through the step-by-step description of several useful features. However, the tutorial is not intended to be a deep technical document covering all the functions, only the most common will be explained, and the readers are intended to further explore other ones.

There are many other methods to clean data. You could ever use any popular spreadsheet software. However, proper use of Open Refine may save you a lot of time.

1.2 ORGANIZATION

In this tutorial, we start with an overview of the tool and a detailed description of several available procedures. To illustrate those, concrete data files will be used, mostly from the Barcelona Open Data site. Finally, we will propose some small exercises that revisit some of the features, and a final task where you will have to clean a data file from scratch.

2. Overview

As already mentioned, Open Refine is a free tool that provides data wrangling features. The software can be downloaded from https://openrefine.org/download.html.

The way to work with Open Refine consists of the following steps:

1. Open the file
2. Explore the data
3. Cleanse the data
4. Save
5. Optionally, apply the same modifications to other files.

When opening the file, we need to be aware of some information, such as the file format or the encoding, since the software can perform different modifications upon loading that can save us a ton of effort.

Before we perform different data wrangling tasks, the initial step is to understand how the data looks. We need to be aware of several issues such as whether it contains typos, whether the data formats were interpreted properly (e.g. numbers loaded as numbers and not text), and so on.

Once we have a feeling of how the data looks and the potential defects to correct, we can proceed to modify the data.

Finally, the file can be saved.

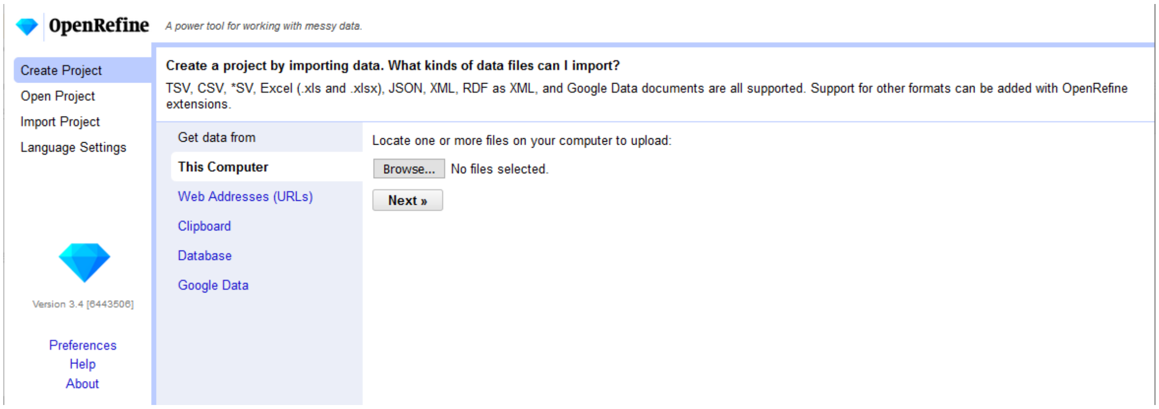
Eventually, if the modifications were performed systematically and in a general manner, those can be applied (in the same order) to a different file with a single operation.

Open Refine has powerful operations that let you modify a lot of records at the same time. But it also has very useful undoing commands that let you undo (and redo) multiple editions. Moreover, the system also provides very powerful previsualization of the results before applying a command, so that we can interactively check whether the command we are writing will perform as expected.

Before we dive into the different possibilities, let's give a small overview of the layout and organization of the application.

2.1 LABOUT

After downloading and installing, running Open Refine will open a new tab in your browser (everything shown here has been tested with Mozilla Firefox), and you can start working. Initially, the software looks like the screen shown in Figure 1.



Create a project by importing data. What kinds of data files can I import?

TSV, CSV, *SV, Excel (.xls and .xlsx), JSON, XML, RDF as XML, and Google Data documents are all supported. Support for other formats can be added with OpenRefine extensions.

Locate one or more files on your computer to upload:

No files selected.

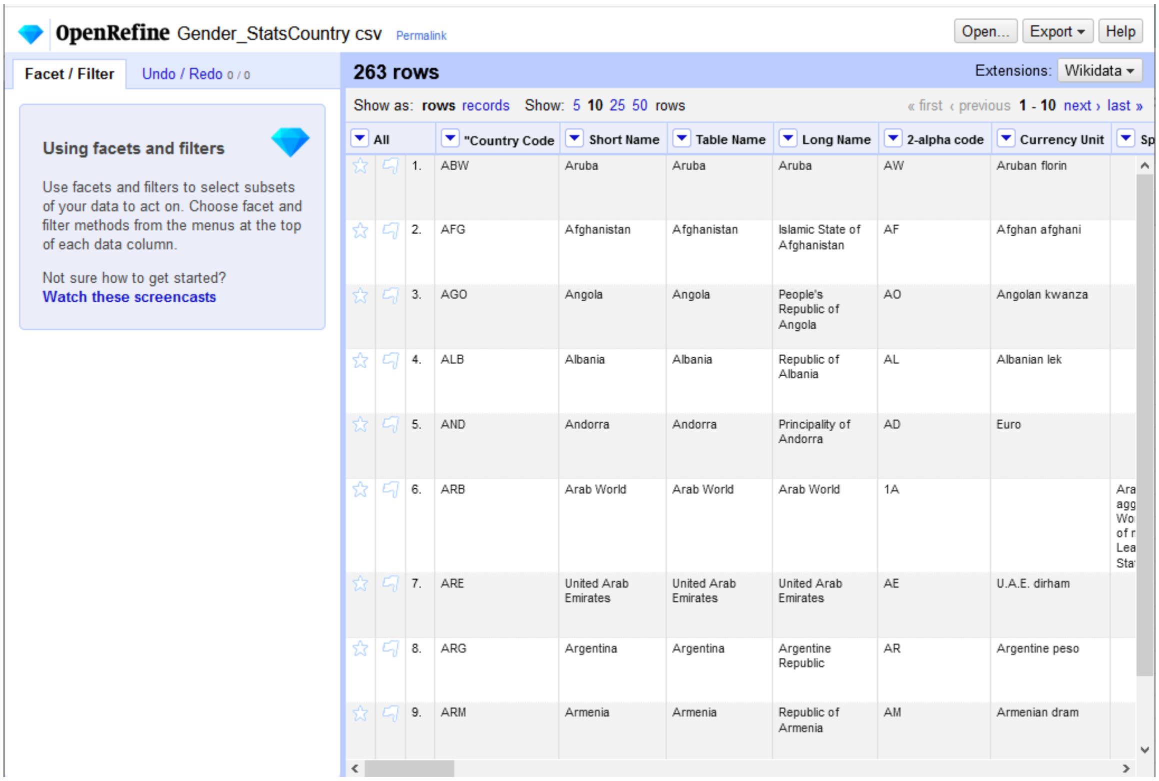
Although the application runs as a tab in a browser, no information is sent through the network (unless activating certain functions that require so). Therefore, your data will stay safe on your computer.

Note that the software may load different data formats besides the popular XML or CSV. You can even load files directly from URLs.

After opening a file, the application looks like the image in Figure 2.

The left pane will contain some data on the operations performed, as we will see later. The top menu lets you change or export the file, and the operations will be mostly carried out upon selection of the dropdown menu shown as a small arrow pointing down that appears next to each column and next to the initial one, called "All".

Operations executed on a single column, triggered by the dropdown menu appearing next to a column, will affect only the data in that column (see Figure 3). The menu that appears when using the dropdown of the "All" column is slightly different and contains operations that can be applied to the whole dataset at once.



Using facets and filters

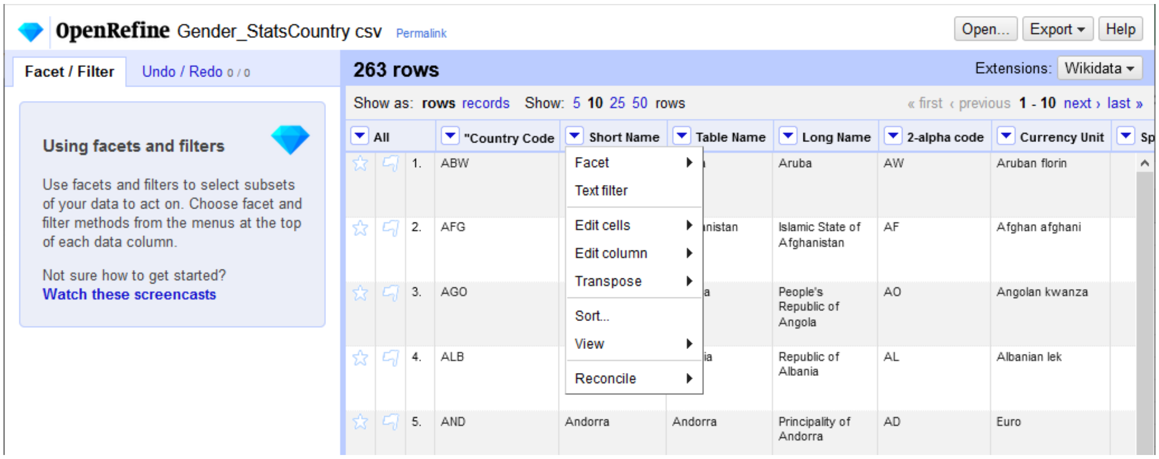
Use facets and filters to select subsets of your data to act on. Choose facet and filter methods from the menus at the top of each data column.

Not sure how to get started?
Watch these screencasts

263 rows

Show as: rows records Show: 5 10 25 50 rows « first ‹ previous 1 - 10 next › last »

All	"Country Code	Short Name	Table Name	Long Name	2-alpha code	Currency Unit	Sp
1.	ABW	Aruba	Aruba	Aruba	AW	Aruban florin	
2.	AFG	Afghanistan	Afghanistan	Islamic State of Afghanistan	AF	Afghan afghani	
3.	AGO	Angola	Angola	People's Republic of Angola	AO	Angolan kwanza	
4.	ALB	Albania	Albania	Republic of Albania	AL	Albanian lek	
5.	AND	Andorra	Andorra	Principality of Andorra	AD	Euro	
6.	ARB	Arab World	Arab World	Arab World	1A		Ara agg Wo of r Lea Sta
7.	ARE	United Arab Emirates	United Arab Emirates	United Arab Emirates	AE	U.A.E. dirham	
8.	ARG	Argentina	Argentina	Argentine Republic	AR	Argentine peso	
9.	ARM	Armenia	Armenia	Republic of Armenia	AM	Armenian dram	



Using facets and filters

Use facets and filters to select subsets of your data to act on. Choose facet and filter methods from the menus at the top of each data column.

Not sure how to get started?
Watch these screencasts

263 rows

Show as: rows records Show: 5 10 25 50 rows « first ‹ previous 1 - 10 next › last »

All	"Country Code	Short Name	Table Name	Long Name	2-alpha code	Currency Unit	Sp
1.	ABW			Aruba	AW	Aruban florin	
2.	AFG		nistan	Islamic State of Afghanistan	AF	Afghan afghani	
3.	AGO		a	People's Republic of Angola	AO	Angolan kwanza	
4.	ALB		ia	Republic of Albania	AL	Albanian lek	
5.	AND	Andorra	Andorra	Principality of Andorra	AD	Euro	

Above the column titles, there is a row of operations that lets the user change the view to explore more (or fewer) records at once (the option of 10 records is the default) and navigate through the rest of the records (options "next", "last" on the right).

2.2 EXPLORING DATA

For the rest of the examples, we encourage you to replicate everything that is described here. The data files used for the examples are available through Racó.

With the aforementioned button and the sliders, you can explore the dataset. This exploration is crucial since it has two objectives: get an idea of what the data contains, analyze the potential data defects.

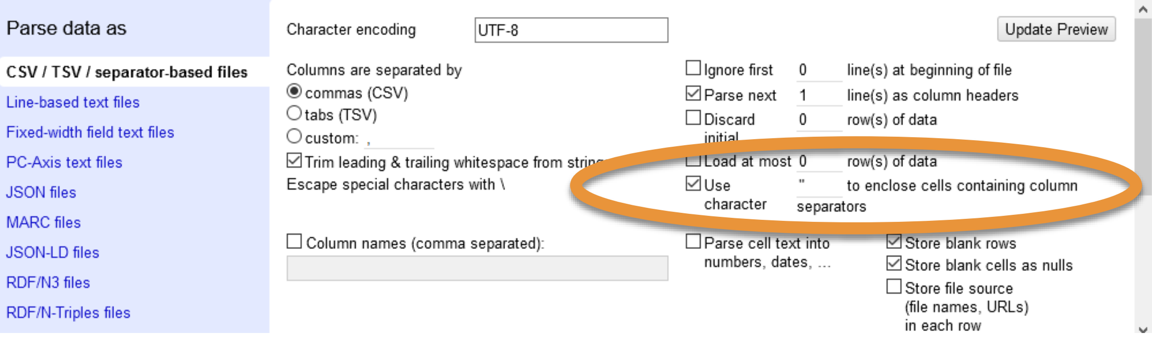
To properly explore the data, we will load a real dataset. The first task you have to do is to open Open Refine and load the file *Gender_StatsCountry.csv*.

After the file is selected through the "Browse" button, you will find out the first problem: all the data appears in the same row.

The reason is that, despite being a CSV file, it has its odds that results in two issues:

1. The loader has not detected line changes.
2. The loader has not inferred row names.

There are several ways of addressing this problem using Open Refine. The simplest one is letting Open Refine know that the file contains the symbol " to enclose cells containing column separators. We need to disable the option highlighted in Figure 4.



Character encoding UTF-8

Columns are separated by
● commas (CSV)
○ tabs (TSV)
○ custom: ,
☑ Trim leading & trailing whitespace from strin
Escape special characters with \

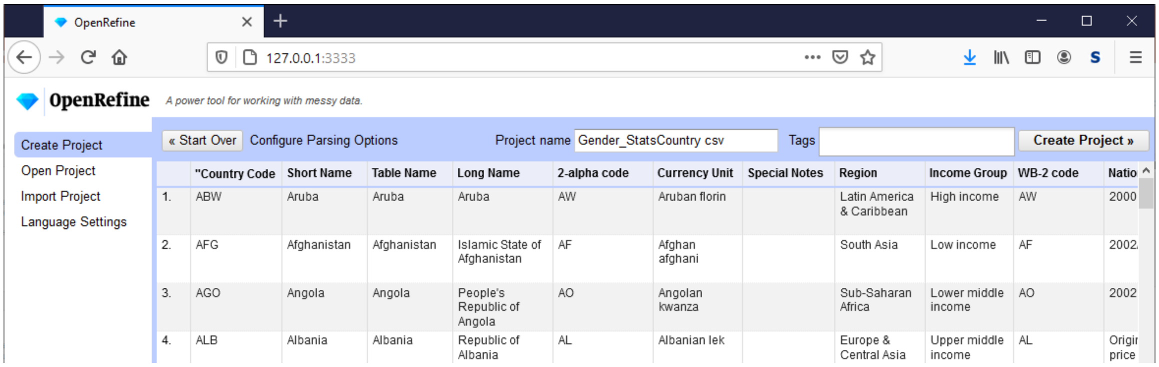
□ Column names (comma separated):

□ Ignore first 0 line(s) at beginning of file
☑ Parse next 1 line(s) as column headers
□ Discard initial 0 row(s) of data
□ Load at most 0 row(s) of data
☑ Use character " to enclose cells containing column separators

□ Parse cell text into numbers, dates, ...
☑ Store blank rows
☑ Store blank cells as nulls
□ Store file source (file names, URLs) in each row

Figure 4: Ensuring Open Refine parses data properly.

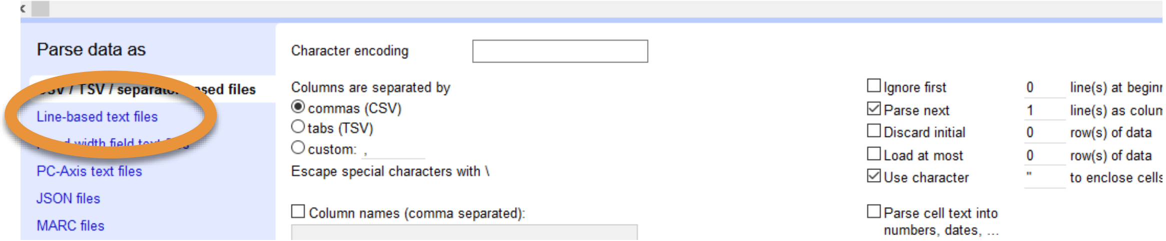
The result will be the one appearing in Figure 5. Note that the result is not completely perfect, because we see some columns where the header contains the symbol ", such as "Country Code". We will deal with this later.



Configure Parsing Options Project name Gender_StatsCountry csv Tags

	"Country Code	Short Name	Table Name	Long Name	2-alpha code	Currency Unit	Special Notes	Region	Income Group	WB-2 code	Natio
1.	ABW	Aruba	Aruba	Aruba	AW	Aruban florin		Latin America & Caribbean	High income	AW	2000
2.	AFG	Afghanistan	Afghanistan	Islamic State of Afghanistan	AF	Afghan afghani		South Asia	Low income	AF	2002
3.	AGO	Angola	Angola	People's Republic of Angola	AO	Angolan kwanza		Sub-Saharan Africa	Lower middle income	AO	2002
4.	ALB	Albania	Albania	Republic of Albania	AL	Albanian lek		Europe & Central Asia	Upper middle income	AL	Origi price

Another parsing option is the one that lets Open Refine the input as a line-based file. In such a case, changing the option to the one on the left as shown in Figure 6 may solve the problem.



Character encoding

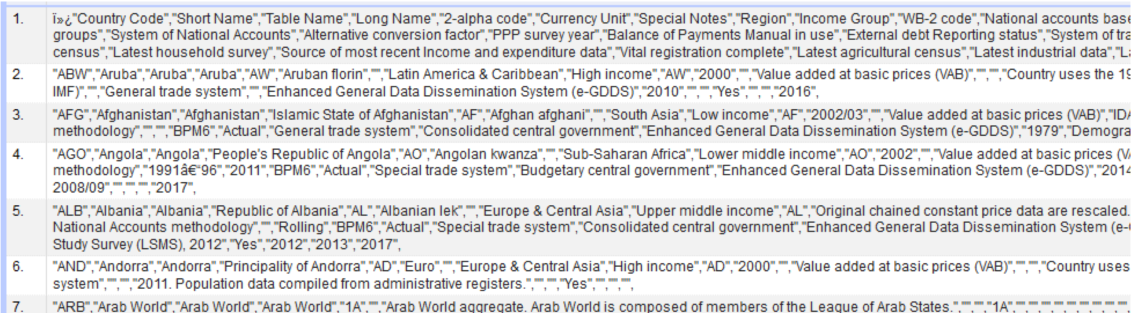
Columns are separated by
● commas (CSV)
○ tabs (TSV)
○ custom: ,
Escape special characters with \

□ Column names (comma separated):

□ Ignore first 0 line(s) at begin
☑ Parse next 1 line(s) as colun
□ Discard initial 0 row(s) of data
□ Load at most 0 row(s) of data
☑ Use character " to enclose cells
□ Parse cell text into numbers, dates, ...

Figure 6: Loaded data after the previous modification.

Unfortunately, just doing so for our concrete example file, does not solve the problem properly, since, in this case, it does not detect file headers. The result of the reading procedure using this option, instead of the previous one would be the one shown in Figure 7. Note that each entry contains a line where all the values are stored as a single record.



```
1.  ï»¿"Country Code","Short Name","Table Name","Long Name","2-alpha code","Currency Unit","Special Notes","Region","Income Group","WB-2 code","National accounts base
    groups","System of National Accounts","Alternative conversion factor","PPP survey year","Balance of Payments Manual in use","External debt Reporting status","System of tra
    census","Latest household survey","Source of most recent Income and expenditure data","Vital registration complete","Latest agricultural census","Latest industrial data","L
2.  "ABW","Aruba","Aruba","Aruba","AW","Aruban florin","","Latin America & Caribbean","High income","AW","2000","","Value added at basic prices (VAB)","","","Country uses the 19
    IMF)","","General trade system","","Enhanced General Data Dissemination System (e-GDDS)","2010","","","Yes","","","2016",
3.  "AFG","Afghanistan","Afghanistan","Islamic State of Afghanistan","AF","Afghan afghani","","South Asia","Low income","AF","2002/03","","Value added at basic prices (VAB)","ID
    methodology","","","BPM6","Actual","General trade system","Consolidated central government","Enhanced General Data Dissemination System (e-GDDS)","1979","Demogra
4.  "AGO","Angola","Angola","People's Republic of Angola","AO","Angolan kwanza","","Sub-Saharan Africa","Lower middle income","AO","2002","","Value added at basic prices (V
    methodology","1991â€"96","2011","BPM6","Actual","Special trade system","Budgetary central government","Enhanced General Data Dissemination System (e-GDDS)","2014
    2008/09","","","","2017",
5.  "ALB","Albania","Albania","Republic of Albania","AL","Albanian lek","","Europe & Central Asia","Upper middle income","AL","Original chained constant price data are rescaled.
    National Accounts methodology","","Rolling","BPM6","Actual","Special trade system","Consolidated central government","Enhanced General Data Dissemination System (e-
    Study Survey (LSMS), 2012","Yes","2012","2013","2017",
6.  "AND","Andorra","Andorra","Principality of Andorra","AD","Euro","","Europe & Central Asia","High income","AD","2000","","Value added at basic prices (VAB)","","","Country uses
    system","","","2011. Population data compiled from administrative registers.","","","Yes","","","",
7.  "ARB","Arab World","Arab World","Arab World","1A","","Arab World aggregate. Arab World is composed of members of the League of Arab States.","","","1A","","","","","","","","",
```

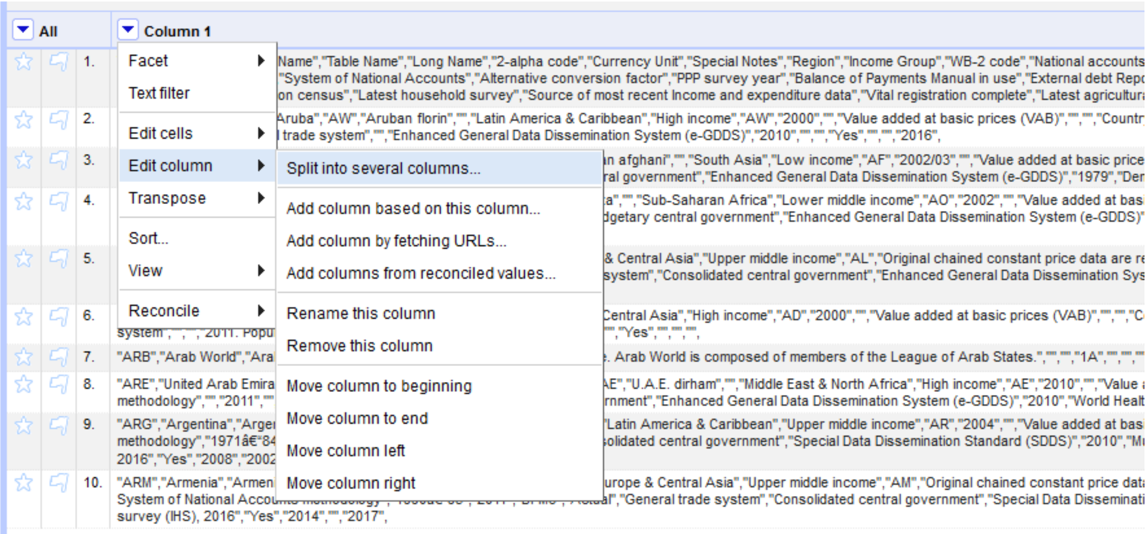
3. Data wrangling

In this section, we will introduce several data edition options. To illustrate those, we will start from the previous example, that is the file *Gender_StatsCountry.csv* loaded by using as parsing option the "line-based file". This will give us the opportunity of showing the most basic edition tasks.

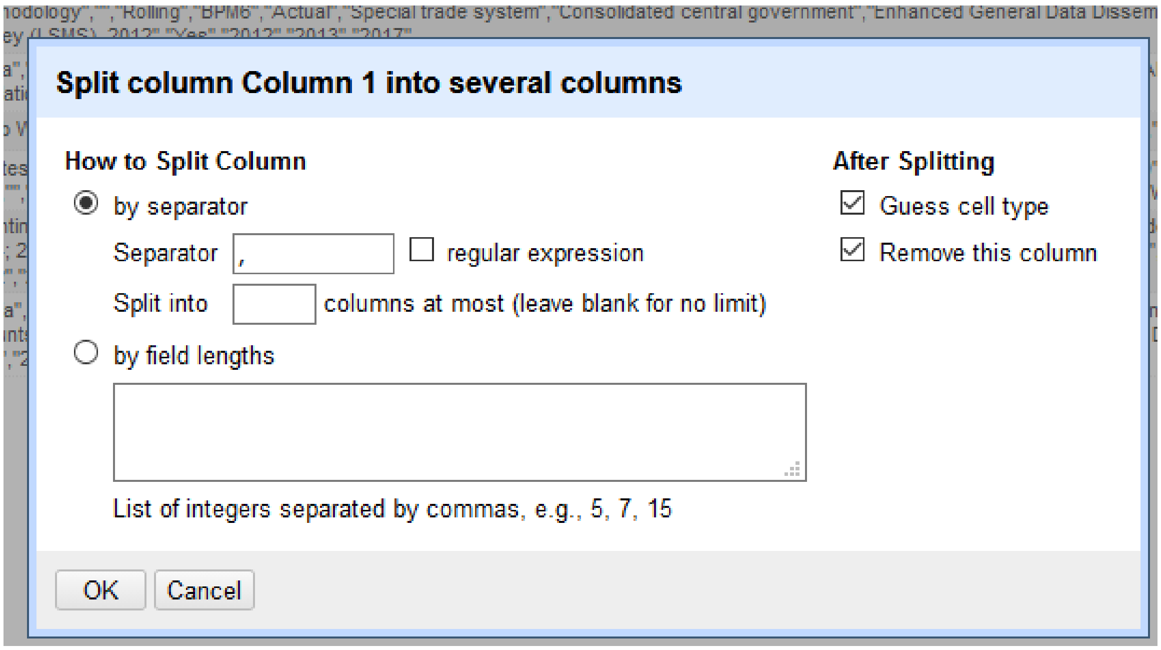
3.1 BASIC DATA EDITIONS

The first thing we want to correct in this version of the file is to separate the data into different columns. Note that we only have two columns we can operate with "All" and "Column 1". We can achieve this data splitting through (shown in Figure 8):

Dropdown Column 1 → Edit column → Split into several columns



To properly separate, we further need to tell Open Refine what separator is the one that delimits columns. In our case, it is the comma. Thus, we fill the form that appears upon selecting the previous option accordingly, as illustrated in Figure 9.



Split column Column 1 into several columns

How to Split Column

● by separator

Separator: , □ regular expression

Split into: (blank) columns at most (leave blank for no limit)

○ by field lengths

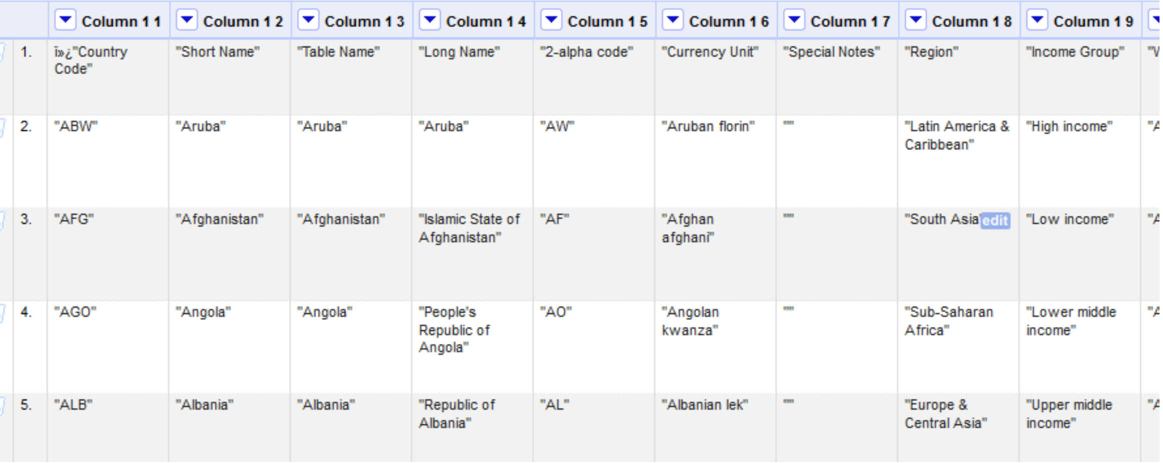
List of integers separated by commas, e.g., 5, 7, 15

After Splitting

☑ Guess cell type

☑ Remove this column

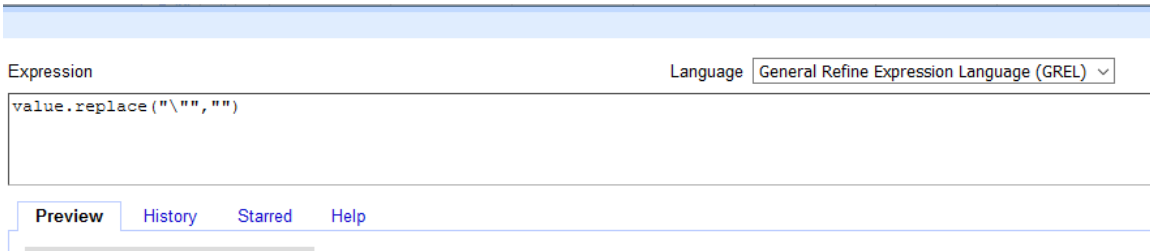
The result will be a new version where the columns will be properly separated, as shown in Figure 10.



	Column 11	Column 12	Column 13	Column 14	Column 15	Column 16	Column 17	Column 18	Column 19
1.	ï»¿"Country Code"	"Short Name"	"Table Name"	"Long Name"	"2-alpha code"	"Currency Unit"	"Special Notes"	"Region"	"Income Group"
2.	"ABW"	"Aruba"	"Aruba"	"Aruba"	"AW"	"Aruban florin"	""	"Latin America & Caribbean"	"High income"
3.	"AFG"	"Afghanistan"	"Afghanistan"	"Islamic State of Afghanistan"	"AF"	"Afghan afghani"	""	"South Asia"	"Low income"
4.	"AGO"	"Angola"	"Angola"	"People's Republic of Angola"	"AO"	"Angolan kwanza"	""	"Sub-Saharan Africa"	"Lower middle income"
5.	"ALB"	"Albania"	"Albania"	"Republic of Albania"	"AL"	"Albanian lek"	""	"Europe & Central Asia"	"Upper middle income"

Now we can proceed to clean some obvious typos: the inverted commas that plague all the records. Recall that this would be mostly avoided if we had properly loaded the file, but this example is only for didactic purposes.

To replace a symbol across the file, we can select the option "Transform" in the "All" column dropdown. This will show a form that can be completed with a function. In our case, it will be *value.replace(symbol to replace, replacing* *symbol)*. Note that inputting certain symbols in any programming language can be tricky. Since the symbol needs to be enclosed with inverted commas, and the symbol to replace is inverted commas too, the symbol to replace would be interpreted as the closing mark unless we modify it in a certain way. This is achieved using the symbol "\", which tells the language not to interpret what is coming next. Finally, to eliminate the symbol, we must tell the application that it must be replaced with the null string, which is represented by opening and closing inverted commas. Therefore, the concrete command to use is the one shown in Figure 11.



Expression

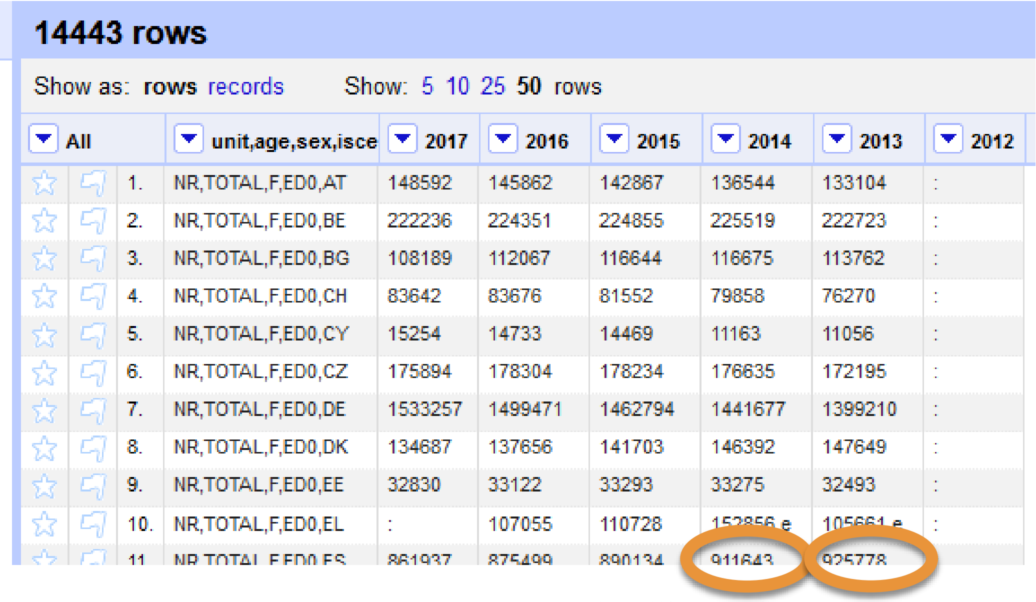
Language: General Refine Expression Language (GREL)

```
value.replace("\"","")
```

Figure 11: Replacing inverted commas across the file.

For the following examples, let's open the file called educ_uoe_enra02.tsv that contains the information regarding education levels in different countries of Europe.

Note that, after opening the file, among other problems, several fields have white space and the "e" letter after the value (see Figure 12).



14443 rows

Show as: rows records   Show: 5 10 25 50 rows

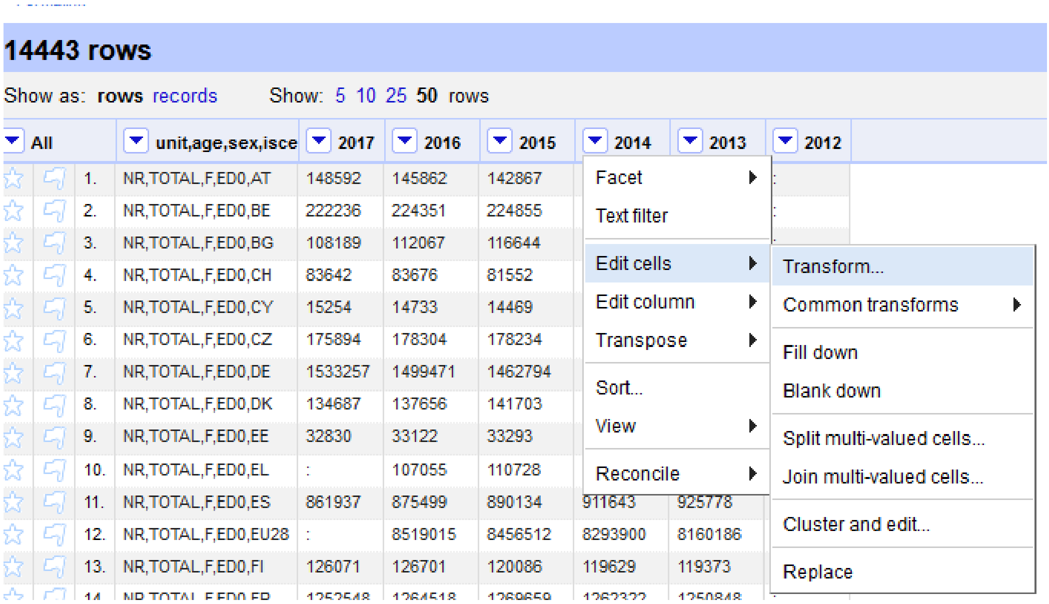
All			unit,age,sex,isce	2017	2016	2015	2014	2013	2012
☆		1.	NR,TOTAL,F,ED0,AT	148592	145862	142867	136544	133104	:
☆		2.	NR,TOTAL,F,ED0,BE	222236	224351	224855	225519	222723	:
☆		3.	NR,TOTAL,F,ED0,BG	108189	112067	116644	116675	113762	:
☆		4.	NR,TOTAL,F,ED0,CH	83642	83676	81552	79858	76270	:
☆		5.	NR,TOTAL,F,ED0,CY	15254	14733	14469	11163	11056	:
☆		6.	NR,TOTAL,F,ED0,CZ	175894	178304	178234	176635	172195	:
☆		7.	NR,TOTAL,F,ED0,DE	1533257	1499471	1462794	1441677	1399210	:
☆		8.	NR,TOTAL,F,ED0,DK	134687	137656	141703	146392	147649	:
☆		9.	NR,TOTAL,F,ED0,EE	32830	33122	33293	33275	32493	:
☆		10.	NR,TOTAL,F,ED0,EL	:	107055	110728	152856 e	105661 e	:
☆		11.	NR,TOTAL,F,ED0,ES	861937	875499	890134	911643	925778	

Figure 12: Extra text in different fields.

We can modify this with different strategies, in this case, we are going to use the option "Edit cells" of the column to edit. Therefore, what we need to do is:

The dropdown of the column → Edit cells → Transform

This is illustrated in Figure 13.



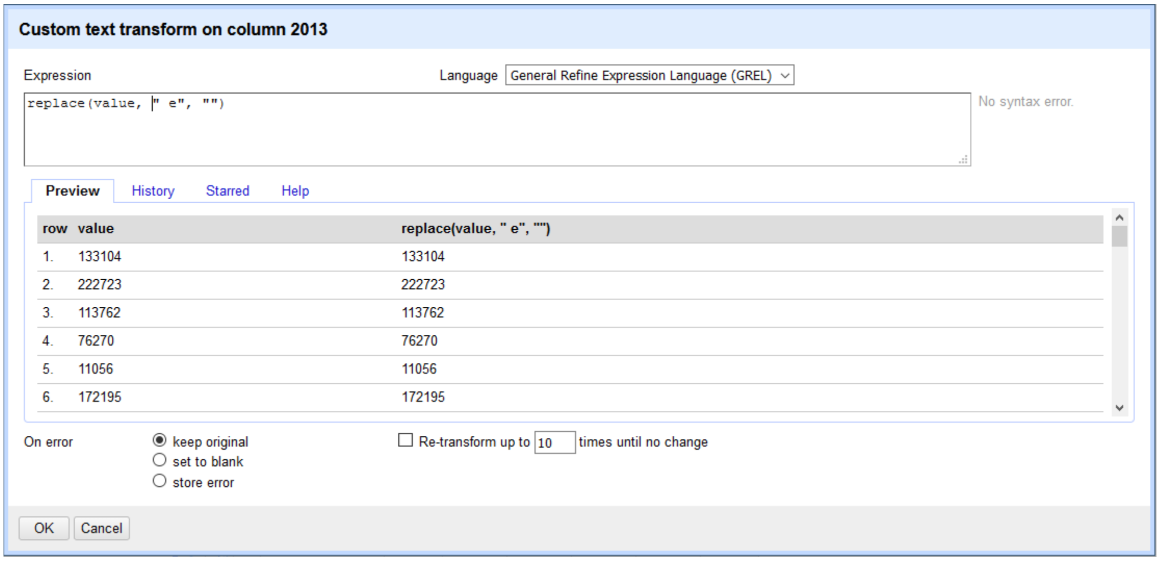
14443 rows

Show as: rows records Show: 5 10 25 50 rows

	All	unit,age,sex,isce	2017	2016	2015	2014	2013	2012
	1.	NR,TOTAL,F,ED0,AT	148592	145862	142867			
	2.	NR,TOTAL,F,ED0,BE	222236	224351	224855			
	3.	NR,TOTAL,F,ED0,BG	108189	112067	116644			
	4.	NR,TOTAL,F,ED0,CH	83642	83676	81552			
	5.	NR,TOTAL,F,ED0,CY	15254	14733	14469			
	6.	NR,TOTAL,F,ED0,CZ	175894	178304	178234			
	7.	NR,TOTAL,F,ED0,DE	1533257	1499471	1462794			
	8.	NR,TOTAL,F,ED0,DK	134687	137656	141703			
	9.	NR,TOTAL,F,ED0,EE	32830	33122	33293			
	10.	NR,TOTAL,F,ED0,EL	:	107055	110728			
	11.	NR,TOTAL,F,ED0,ES	861937	875499	890134	911643	925778	
	12.	NR,TOTAL,F,ED0,EU28	:	8519015	8456512	8293900	8160186	
	13.	NR,TOTAL,F,ED0,FI	126071	126701	120086	119629	119373	
	14.	NR,TOTAL,F,ED0,FR	1252548	1264518	1269659	1262322	1250848	

Figure 13: Transforming fields of a column.

Then, we can use the *replace* custom function and see the preview of the operation in the bottom part, as shown in Figure 14.



Custom text transform on column 2013

Expression Language General Refine Expression Language (GREL)

```
replace(value, " e", "")
```

No syntax error.

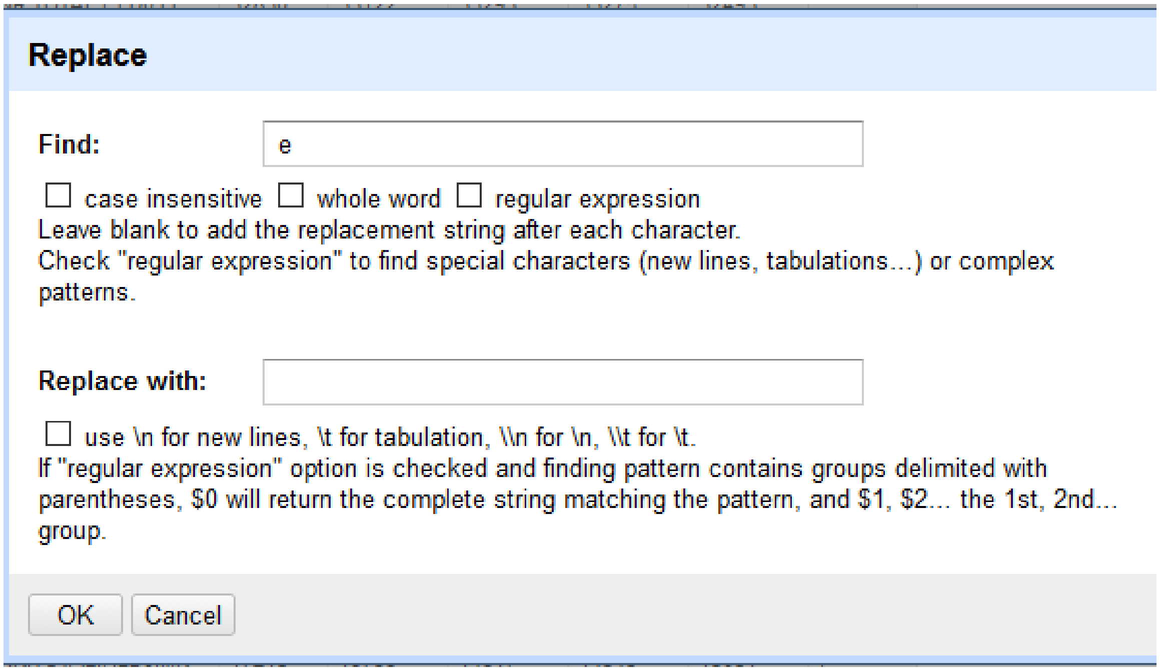
row	value	replace(value, " e", "")
1.	133104	133104
2.	222723	222723
3.	113762	113762
4.	76270	76270
5.	11056	11056
6.	172195	172195

On error ● keep original
○ set to blank
○ store error

□ Re-transform up to 10 times until no change

Figure 14: Replacing function with preview.

The same can be achieved with the replace command, which would not provide a preview, as shown in Figure 15. Note that in both cases, the whitespace is important, otherwise, it would replace the letter "e", not the extra whitespace and the letter.


Replace

Find: e

□ case insensitive □ whole word □ regular expression
Leave blank to add the replacement string after each character.
Check "regular expression" to find special characters (new lines, tabulations...) or complex patterns.

Replace with:

□ use \n for new lines, \t for tabulation, \\n for \n, \\t for \t.
If "regular expression" option is checked and finding pattern contains groups delimited with parentheses, $0 will return the complete string matching the pattern, and $1, $2... the 1st, 2nd... group.

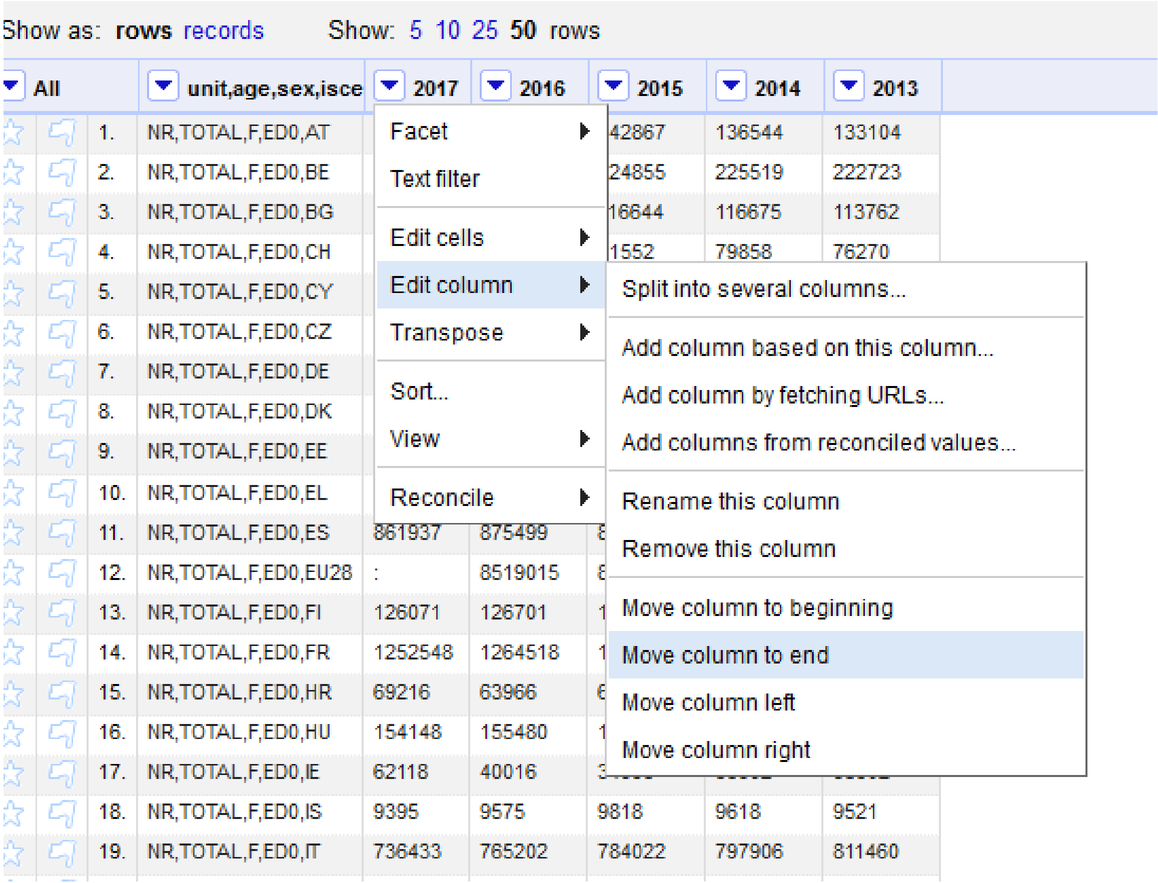
Figure 15: Replacing command. Observe the "e" is preceded by white space.

We can now proceed with the same operations for the other columns.

To remove incomplete data, we can completely erase a full column. In our current example, the data for the year 2012 is only available for Germany. If our problem lets us do so, we can completely erase the year 2012 column. In other cases, we should complete the missing data with real values.

To erase a column, we just need to go to "Edit column" and select "Remove this column".

Another thing that we see is that the years are not ordered in ascending order. We can move columns with the "Edit column" option. In this case, we move the year 2017 to the end by using "Edit column" and "Move column to end", as illustrated in Figure 16.



All	unit,age,sex,isce	2017	2016	2015	2014	2013
1.	NR,TOTAL,F,ED0,AT			42867	136544	133104
2.	NR,TOTAL,F,ED0,BE			24855	225519	222723
3.	NR,TOTAL,F,ED0,BG			16644	116675	113762
4.	NR,TOTAL,F,ED0,CH			1552	79858	76270
5.	NR,TOTAL,F,ED0,CY					
6.	NR,TOTAL,F,ED0,CZ					
7.	NR,TOTAL,F,ED0,DE					
8.	NR,TOTAL,F,ED0,DK					
9.	NR,TOTAL,F,ED0,EE					
10.	NR,TOTAL,F,ED0,EL					
11.	NR,TOTAL,F,ED0,ES	861937	875499	8		
12.	NR,TOTAL,F,ED0,EU28	:	8519015	8		
13.	NR,TOTAL,F,ED0,FI	126071	126701	1		
14.	NR,TOTAL,F,ED0,FR	1252548	1264518	1		
15.	NR,TOTAL,F,ED0,HR	69216	63966	6		
16.	NR,TOTAL,F,ED0,HU	154148	155480	1		
17.	NR,TOTAL,F,ED0,IE	62118	40016	3		
18.	NR,TOTAL,F,ED0,IS	9395	9575	9818	9618	9521
19.	NR,TOTAL,F,ED0,IT	736433	765202	784022	797906	811460

Figure 16: Moving the 2017 column to the end.

We can reorder the other columns with the moving commands ("Move column to beginning", "Move column to end", "Move column left", and "Move column right").

Other minor editions consist in erasing trailing whitespaces, collapsing consecutive whitespaces, and changing the data type (e.g. text to number). You can access all of these transformations by selecting the "Common transforms" option under the "Edit" menu, as depicted in Figure 17.

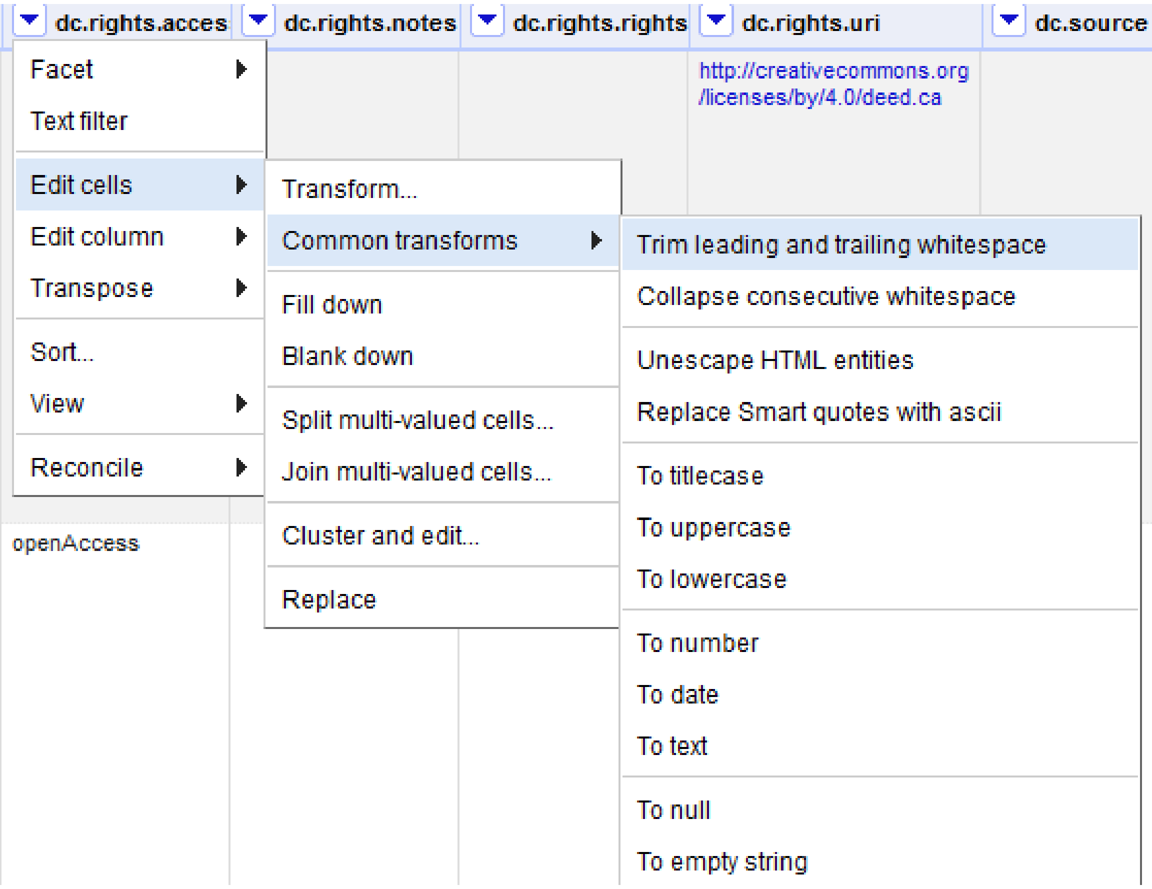


Figure 17: Simple data transformations.

Note that you can also change data to uppercase lowercase, and so on, with the same menu.

Other not-so-obvious operations consist of filling blank values with the previous value that appears in a column or filling the following values with blanks. Both of those are also accessible with "Edit cells" + "Fill down" or "Blank down".

So far, the presented operations are rather simple, and many software packages such as spreadsheets or text editors may have some available function that might do the job as easily. In the following section, we will see some complex operations that are typically only available through data wrangler applications or programmatically.

3.2 ADVANCED DATA EDITIONS

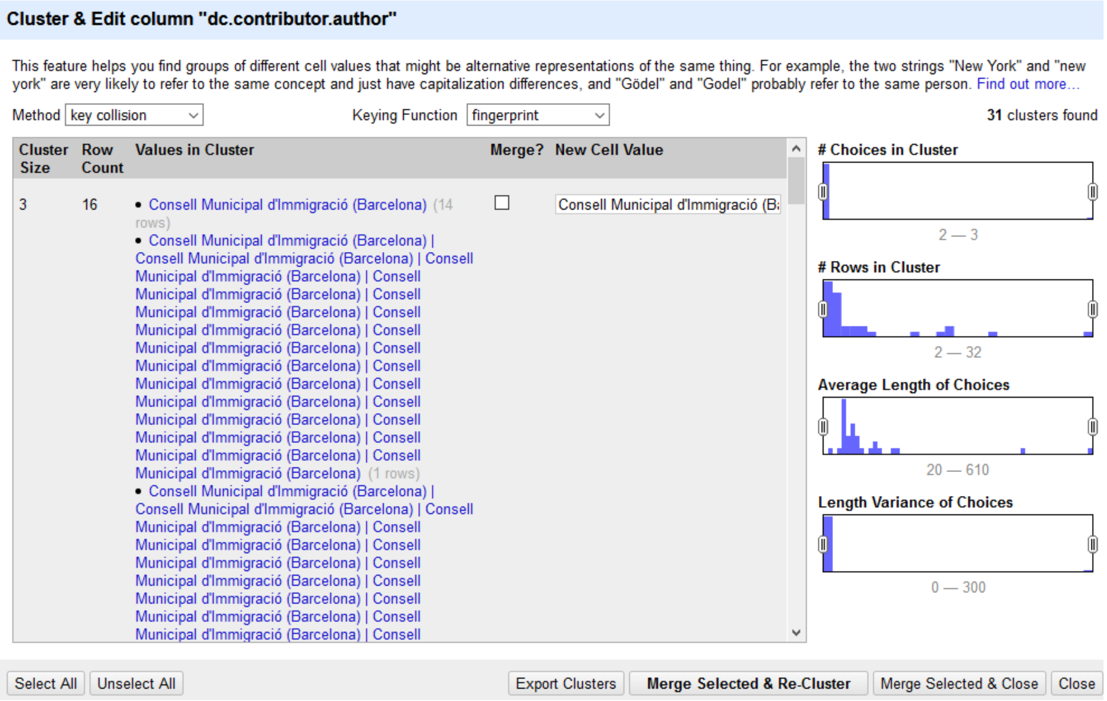
In the following section, we will deal with some more advanced transformations that can lead to both detecting defective data and its proper correction. The complexity of most of the following techniques can also be tackled by using a custom program to clean the data. However, data wrangling software provides some features such as previews of the operation effect, or the possibility of undoing the editions that are not simple with a custom program.

The first operation we are going to see is the "Cluster & Edit" function. Its main purpose is to collectively inspect and edit cells. A common problem in data analysis happens when the same value appears written differently for many cells. This is not easy to detect, since we need to inspect the data and this may become unfeasible if the file is very large.

The "Cluster & Edit" function finds groups of representations and opens a detailed view that provides some insights on the groups, such as the number of rows that have each representation, and offers the possibility of merging and editing values of multiple rows (even with original different values) at once.

To use the function, we will move to the file named Publicacions_369.csv. It stores information on publications released by different departments, with the names of the authors, titles, and all sorts of other data.

If we cluster by column "dc.contributor.author", we will find that some names were written differently, as shown in Figure 18.



Cluster & Edit column "dc.contributor.author"

This feature helps you find groups of different cell values that might be alternative representations of the same thing. For example, the two strings "New York" and "new york" are very likely to refer to the same concept and just have capitalization differences, and "Gödel" and "Godel" probably refer to the same person. Find out more...

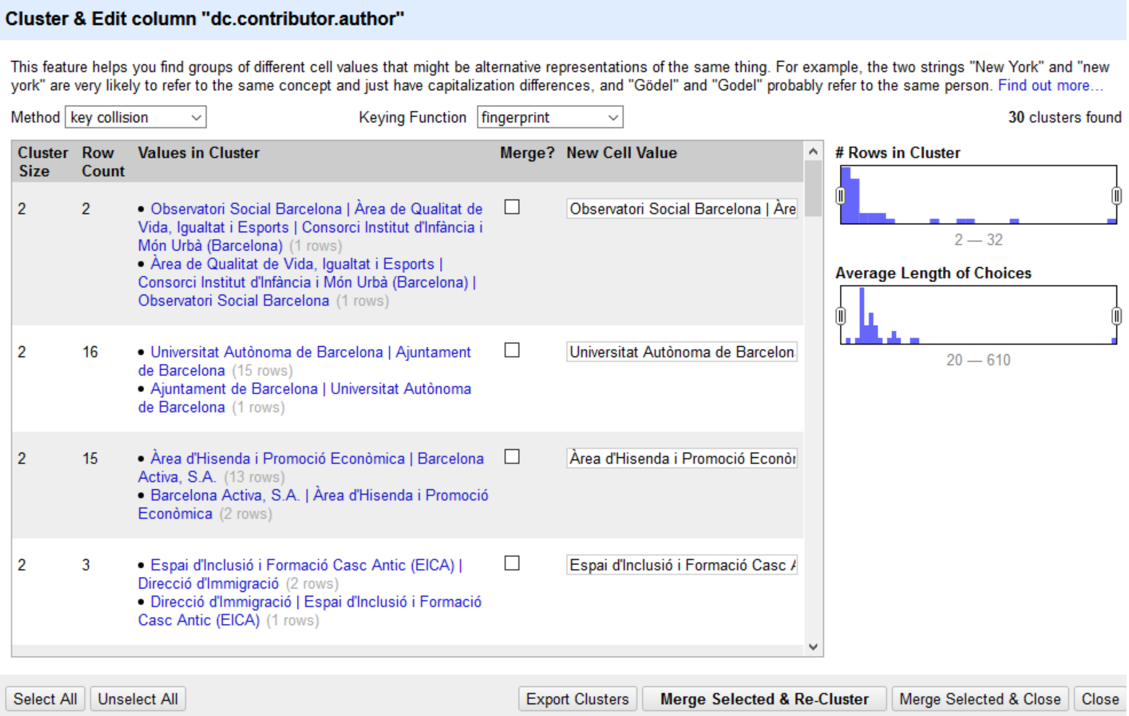
Method key collision Keying Function fingerprint 31 clusters found

Cluster Size	Row Count	Values in Cluster	Merge?	New Cell Value
3	16	• Consell Municipal d'Immigració (Barcelona) (14 rows) • Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) (1 rows) • Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell Municipal d'Immigració (Barcelona) | Consell	□	Consell Municipal d'Immigració (B

Figure 18: Clustering column.

The charts on the right show the statistics of the found values. The left part shows the different names that were applied to the values in the column. The algorithm tries to find names that are repeated but written wrongly (e.g. due to a typo, because wrongly placed trailing spaces, and so on). As a result, we will be offered a set of candidate values that the algorithm believes should be the same, and thus, we can fix it.

In our example, we can see that the “Consell Municipal d’Immigració (Barcelona)” is wrongly written (repeating the name several times) in some rows. By clicking on the “Merge” checkbox and the “Merge Selected & Re-Cluster”, the rows will be updated and will disappear from the view since all the values are the same now (see Figure 19). Similar names might not be detected as belonging to the same cluster. By iteratively correcting the wrong ones, you should converge to the proper solution. The next iteration shows some examples with rows that have the same authors, but they were ordered differently. If the order does not matter (in some cases it might), we can fix this easily. In this case, we can fix the four at once.



Cluster & Edit column "dc.contributor.author"

This feature helps you find groups of different cell values that might be alternative representations of the same thing. For example, the two strings "New York" and "new york" are very likely to refer to the same concept and just have capitalization differences, and "Gödel" and "Godel" probably refer to the same person. Find out more…

Method key collision

Keying Function fingerprint

30 clusters found

Cluster Size	Row Count	Values in Cluster	Merge?	New Cell Value
2	2	• Observatori Social Barcelona | Àrea de Qualitat de Vida, Igualtat i Esports | Consorci Institut d'Infància i Món Urbà (Barcelona) (1 rows) • Àrea de Qualitat de Vida, Igualtat i Esports | Consorci Institut d'Infància i Món Urbà (Barcelona) | Observatori Social Barcelona (1 rows)	□	Observatori Social Barcelona | Àre
2	16	• Universitat Autònoma de Barcelona | Ajuntament de Barcelona (15 rows) • Ajuntament de Barcelona | Universitat Autònoma de Barcelona (1 rows)	□	Universitat Autònoma de Barcelon
2	15	• Àrea d'Hisenda i Promoció Econòmica | Barcelona Activa, S.A. (13 rows) • Barcelona Activa, S.A. | Àrea d'Hisenda i Promoció Econòmica (2 rows)	□	Àrea d'Hisenda i Promoció Econòı
2	3	• Espai d'Inclusió i Formació Casc Antic (EICA) | Direcció d'Immigració (2 rows) • Direcció d'Immigració | Espai d'Inclusió i Formació Casc Antic (EICA) (1 rows)	□	Espai d'Inclusió i Formació Casc A

Figure 19: Re-clustering result after fixing the first issue.

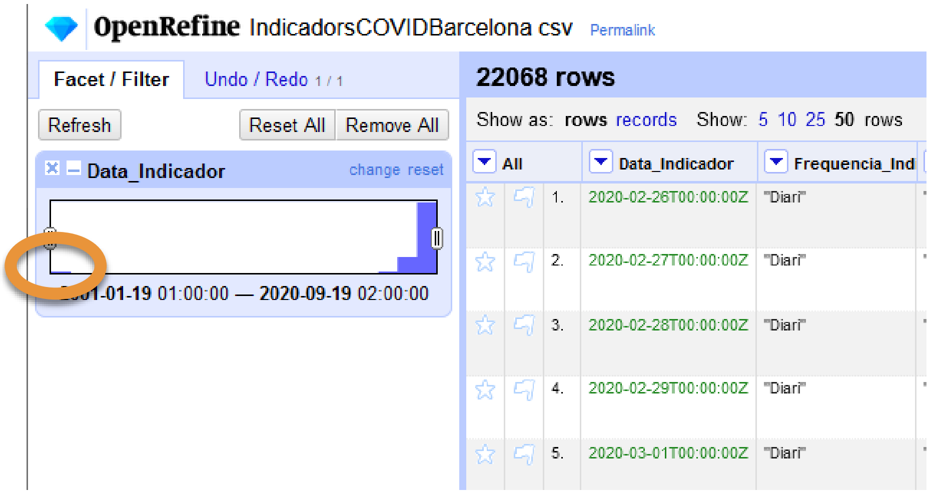
The clustering operation is not only useful to detect typos but also to detect anomalies in values.

For example, in columns that store numerical values, the cluster operation will show the distribution, and thus we may be able to detect some values that are out of the expected range.

Another operation that is complementary to the clustering is the "Facet". It can show us the distribution of values. We are going to use the file IndicadorsCOVIDBarcelona.csv downloaded from the Barcelona Open Data site on the 19th of September.

We want to track how frequently the data was gathered. To do so, we are going to facet the date ("Data_Indicador"). Unfortunately, the loading process did not detect the column as a date. We can fix it with the proper transformation option (to date).

Then, we can facet the column with "Facet" and select the "Timeline facet" option. We will see something strange: some indicators were captured in 2001 (see Figure 20).



All	Data_Indicador	Frequencia_Ind
1.	2020-02-26T00:00:00Z	"Diari"
2.	2020-02-27T00:00:00Z	"Diari"
3.	2020-02-28T00:00:00Z	"Diari"
4.	2020-02-29T00:00:00Z	"Diari"
5.	2020-03-01T00:00:00Z	"Diari"

Figure 20: Faceting the date column.

We can inspect what happens here by adjusting the right view to the rows in this range. This can be easily done by dragging the controllers at the left and right borders of the facet view (shown in Figure 21).

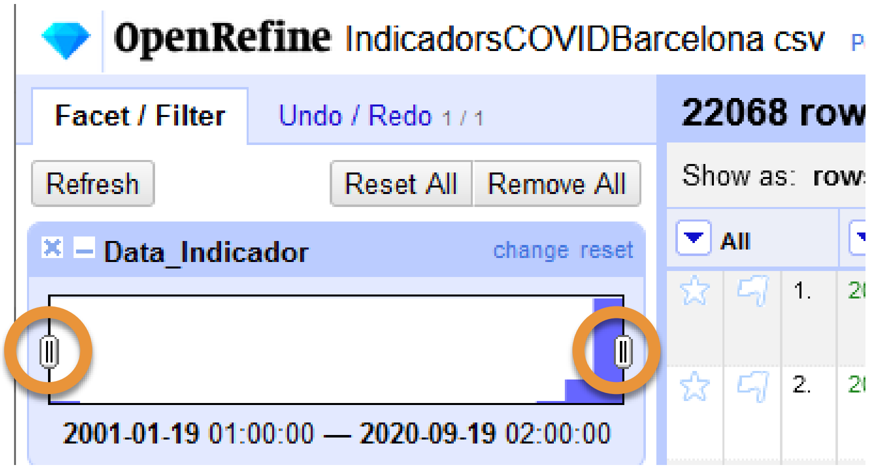
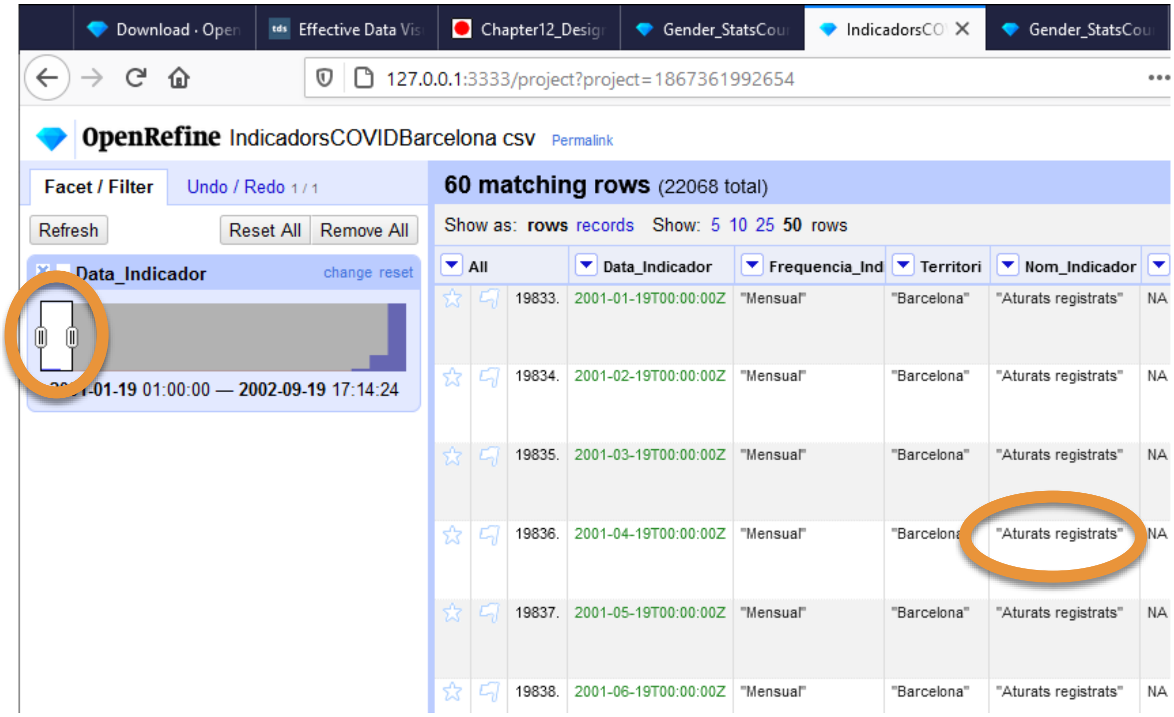


Figure 21: Widgets to resize the faceting preview.

If we drag the right widget to leave the elements of the year 2001 within the selection, we can see the problem, as depicted in Figure 22.



60 matching rows (22068 total)

Show as: rows records Show: 5 10 25 50 rows

All	Data_Indicador	Frequencia_Ind	Territori	Nom_Indicador	
19833.	2001-01-19T00:00:00Z	"Mensual"	"Barcelona"	"Aturats registrats"	NA
19834.	2001-02-19T00:00:00Z	"Mensual"	"Barcelona"	"Aturats registrats"	NA
19835.	2001-03-19T00:00:00Z	"Mensual"	"Barcelona"	"Aturats registrats"	NA
19836.	2001-04-19T00:00:00Z	"Mensual"	"Barcelona	"Aturats registrats"	NA
19837.	2001-05-19T00:00:00Z	"Mensual"	"Barcelona"	"Aturats registrats"	NA
19838.	2001-06-19T00:00:00Z	"Mensual"	"Barcelona"	"Aturats registrats"	NA

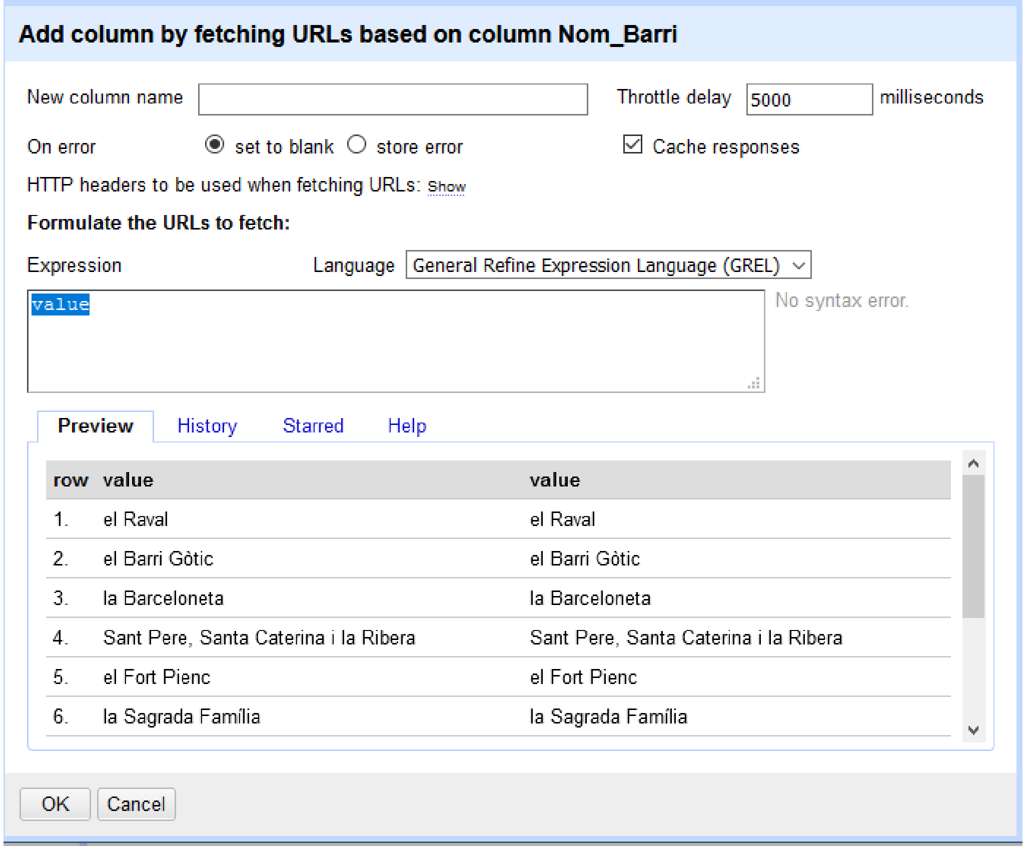
Figure 22: Rows of 2001 belong to another database since they show the number of unemployed people.

We can remove those rows that are wrong with the "All" option named "Remove matching rows". Upon this change, we will see that there are still some rows with non-valid dates.

Finally, a more complex operation consists in augmenting the data. We need this when there is missing data, or we need to derive new values. Open Refine provides a powerful mechanism through the use of web queries from URL addresses.

For example, in the file named 2018_padro_nivell_academic.csv, we have neighborhood names. If we want to use the values of this file and put them in a map, we would need some sort of location to generate the geometric elements. So what we are going to do now is to get a GPS position from a neighborhood.

To do so, we go to the column named "Nom_Barri" and select "Edit column" and "Add column by fetching URLs". The form that will open looks like the one in Figure 23.



Add column by fetching URLs based on column Nom_Barri

New column name Throttle delay 5000 milliseconds

On error ● set to blank ○ store error ☑ Cache responses

HTTP headers to be used when fetching URLs: Show

Formulate the URLs to fetch:

Expression Language General Refine Expression Language (GREL)

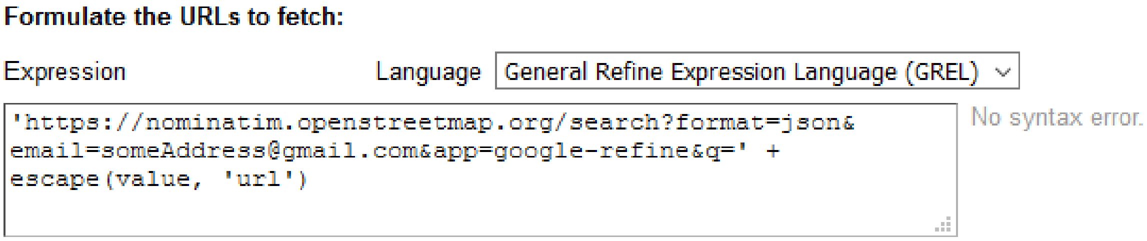
```
value
```

No syntax error.

row	value	value
1.	el Raval	el Raval
2.	el Barri Gòtic	el Barri Gòtic
3.	la Barceloneta	la Barceloneta
4.	Sant Pere, Santa Caterina i la Ribera	Sant Pere, Santa Caterina i la Ribera
5.	el Fort Pienc	el Fort Pienc
6.	la Sagrada Família	la Sagrada Família

Figure 23: Data augmentation through fetching from a URL.

Then, we need to define a new column name (top box), change the delay to something faster (e.g. 1500ms, since Openstreetmap does not want more than one call per second), and enter the URL of the query. It should be written as in figure 24.



Formulate the URLs to fetch:

Expression Language General Refine Expression Language (GREL)

```
'https://nominatim.openstreetmap.org/search?format=json&
email=someAddress@gmail.com&app=google-refine&q=' +
escape(value, 'url')
```

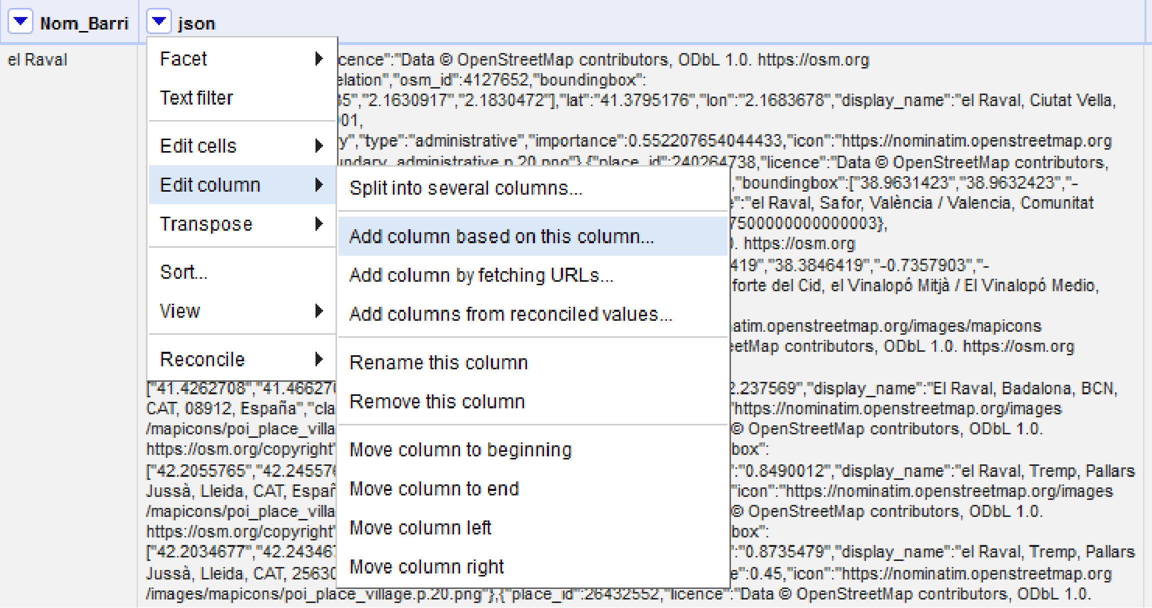
No syntax error.

Figure 24: OpenStreetMap query.

Note that in this case, the preview window is unable to fully resolve the query, but it shows what query will be applied for each value.

The result will be a new column, where each cell contains a JSON code that contains (among a lot of other information), the GPS position of the neighborhood.

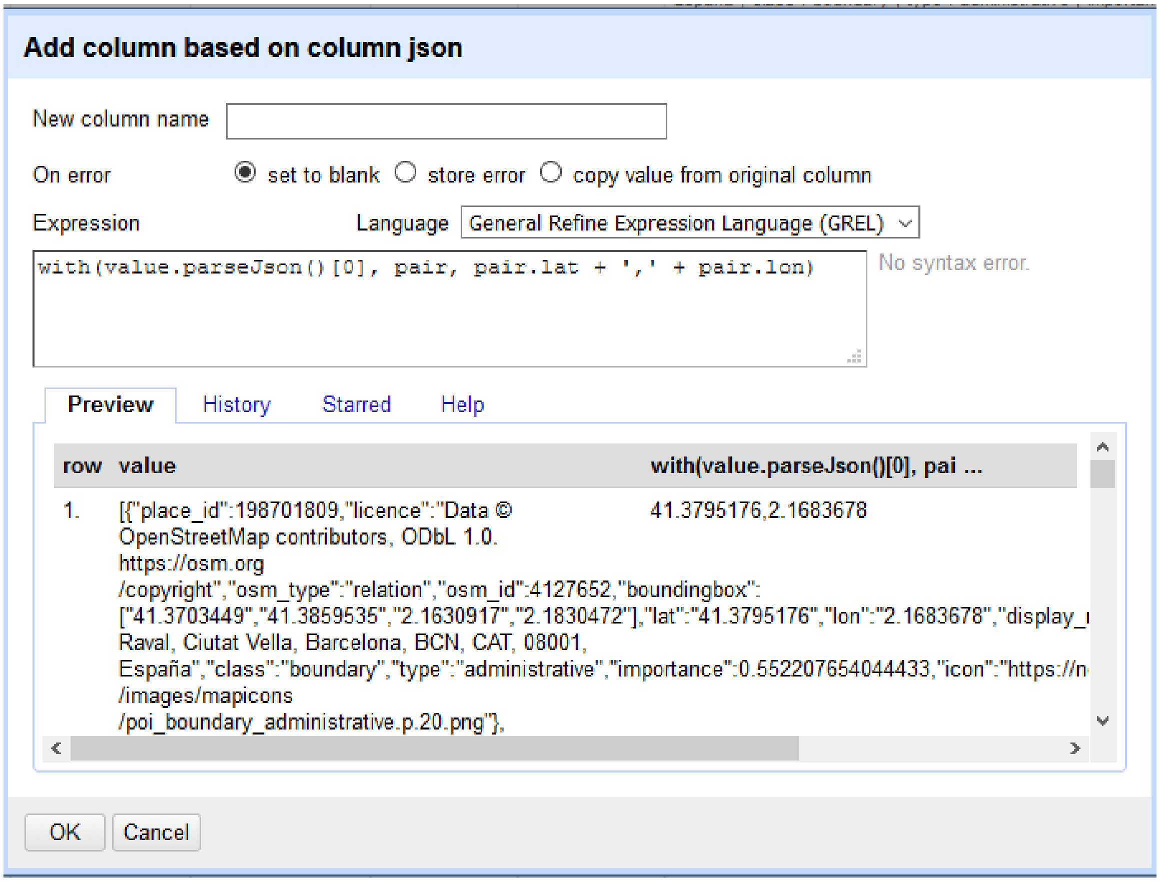
To extract the information of the proper fields, we can add a new column using this information with the command "Edit column" and "Add column based on this column", as shown in Figure 25.



cence":"Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org
elation","osm_id":4127652,"boundingbox":
5","2.1630917","2.1830472"],"lat":"41.3795176","lon":"2.1683678","display_name":"el Raval, Ciutat Vella,
01,
y","type":"administrative","importance":0.552207654044433,"icon":"https://nominatim.openstreetmap.org
238,"licence":"Data © OpenStreetMap contributors,
,"boundingbox":["38.9631423","38.9632423","-
":"el Raval, Safor, València / Valencia, Comunitat
75000000000000003},
. https://osm.org
419","38.3846419","-0.7357903","-
forte del Cid, el Vinalopó Mitjà / El Vinalopó Medio,
atim.openstreetmap.org/images/mapicons
etMap contributors, ODbL 1.0. https://osm.org
["41.4262708","41.46627 2.237569","display_name":"El Raval, Badalona, BCN,
CAT, 08912, España","cla "https://nominatim.openstreetmap.org/images
/mapicons/poi_place_villa © OpenStreetMap contributors, ODbL 1.0.
https://osm.org/copyright" box":
["42.2055765","42.24557 :"0.8490012","display_name":"el Raval, Tremp, Pallars
Jussà, Lleida, CAT, Espa "icon":"https://nominatim.openstreetmap.org/images
/mapicons/poi_place_villa © OpenStreetMap contributors, ODbL 1.0.
https://osm.org/copyright" box":
["42.2034677","42.24346 :"0.8735479","display_name":"el Raval, Tremp, Pallars
Jussà, Lleida, CAT, 2563 e":0.45,"icon":"https://nominatim.openstreetmap.org
/images/mapicons/poi_place_village.p.20.png"},{"place_id":26432552,"licence":"Data © OpenStreetMap contributors, ODbL 1.0.

Figure 25: Adding a new column to extract the GPS coordinates from the JSON code.

In this case, we extract the latitude and longitude using the *parseJson* function and the *with* expression. The code is illustrated in Figure 26.



Add column based on column json

New column name

On error ● set to blank ○ store error ○ copy value from original column

Expression Language General Refine Expression Language (GREL)

```
with(value.parseJson()[0], pair, pair.lat + ',' + pair.lon)
```

No syntax error.

row	value	with(value.parseJson()[0], pai ...
1.	[{"place_id":198701809,"licence":"Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org /copyright","osm_type":"relation","osm_id":4127652,"boundingbox": ["41.3703449","41.3859535","2.1630917","2.1830472"],"lat":"41.3795176","lon":"2.1683678","display_ Raval, Ciutat Vella, Barcelona, BCN, CAT, 08001, España","class":"boundary","type":"administrative","importance":0.552207654044433,"icon":"https://n /images/mapicons /poi_boundary_administrative.p.20.png"},	41.3795176,2.1683678

Figure 26: Extracting the GPS position from the JSON code.

Note how the preview already shows the result.

Then, this result may be split into two different columns.

Recall that you can export the current file at any moment with the "Export" menu on top.

4. Exercises

1. Analyze and clean the file "inca_od_20191115.csv".
2. Analyze and clean the file "airQualityDailySummary.csv".