**Exercise:**

**Open Refine Basics**

**Summary**

OpenRefine is a powerful tool for working with data. It is well suited to cleaning and correcting large sets of data and accepts plugins for enrichment and export. These lessons cover the basics of working with OpenRefine: creating projects, cleaning & transforming data, and exporting for use in other applications.

Attribution—These lessons are derived from work that is Copyright © Software Carpentry (<http://software-carpentry.org/>), [CC-BY-4.0](https://creativecommons.org/licenses/by/4.0/), with minor changes made to accommodate our sample datasets.

**Lesson 1 – Getting Started**

**Motivations for the OpenRefine Lesson**

* It is important to know what you did to your data. Additionally, journals, granting agencies, and other institutions are requiring documentation of the steps you took when working with your data. With OpenRefine, you can capture all actions applied to your raw data and share them with your publication as supplemental material.
* All actions are easily reversed in OpenRefine.
* You *must* save your work to a new file; OpenRefine *does not* modify your original dataset.
* Data is often very messy, and this tool saves a lot of time on cleaning headaches.
* Data cleaning steps often need repeating with multiple files. OpenRefine is perfect for speeding up repetitive tasks by replaying previous actions on multiple datasets.
* Some concepts such as clustering algorithms are quite complex, but OpenRefine makes it easy to introduce them, use them, and show their power.

**Before we get started**

Note: this is a Java program that runs on your machine (not in the cloud). It runs inside your browser, but no web connection is needed.

Follow the [Setup](https://github.com/crowesn/beyond-tei/blob/master/openrefine/setup.md) instructions to install OpenRefine.

If after installation and running OpenRefine, it does not automatically open for you, point your browser at <http://127.0.0.1:3333/> or [http://localhost:3333](http://localhost:3333/) to launch the program.

**Basics of OpenRefine**

You can find out a lot more about OpenRefine at [http://openrefine.org](https://github.com/crowesn/beyond-tei/blob/master/openrefine) and check out some great introductory videos. There is a [Google Group](https://groups.google.com/forum/?hl=en#!forum/openrefine) that can answer a lot of beginner questions and problems. There is also an [OpenRefine Google Plus community](https://plus.google.com/communities/117280693504889048168) where you can find a lot of help and a lot of folks from the life sciences are members. As with other programs of this type, OpenRefine libraries are available too, where you can find a script you need and copy it into your OpenRefine instance to run it on your dataset.

**Features**

* Open source ([source on GitHub](https://github.com/OpenRefine/OpenRefine)).
* A large growing community, from novice to expert, ready to help.
* Works with large-ish datasets (100,000 rows). Does not scale to many millions. (yet).

**Lesson 2 – Working with OpenRefine**

**Creating a Project**

Start the program. (Double-click on the openrefine.exe file (or google-refine.exe if using an older version). Java services will start on your machine, and OpenRefine will open in your Firefox browser).

Launch OpenRefine (see Getting Started with OpenRefine).

OpenRefine can import a variety of file types, including tab separated (\*.tsv), comma separated (\*.csv), Excel (\*.xls, \*.xlsx), JSON, XML, RDF as XML, Google Spreadsheets. See the [OpenRefine Importers page](https://github.com/OpenRefine/OpenRefine/wiki/Importers) for more information.

In this first step, we'll browse our computer to the sample data file for this lesson.

If you haven't already, download the data to your computer: <https://raw.githubusercontent.com/crowesn/beyond-tei/master/datasets/hor_v005_addedValue_notCleaned.csv>

Once OpenRefine is launched in your browser, the left margin as options to **Create Project**, **Open Project**, or **Import Project**. Here we create a project, which just takes a couple steps:

1. click **Create Project** to bring you to "Get data from" "**This Computer**".
2. Click **Browse...** and select file hor\_v005\_addedValue\_notCleaned.csv. Click **Open** in the browse window to finish selection
3. Click **Next>>** under the browse button to upload data into OpenRefine.
4. OpenRefine gives you a preview - a chance to show you it understood the file. If, for example, your file was really tab-delimited, the preview might look strange, you would choose the correct separator in the box shown and click **Update Preview** (look in bottom panel). If this is the wrong file, click **<<Start Over** (look in upper left region).
5. If all looks well, click **Create Project>>** (look in upper right corner).

Note that at step 1, you could upload data in a standard form from a web address by using "Get data from" "**Web Addresses (URLs)**". However, this won't work for all URLs.

**Faceting**

*Exploring data by applying multiple filters*

OpenRefine supports faceted browsing as a mechanism for

* seeing a big picture of your data, and
* filtering down to just the subset of rows that you want to change in bulk.

Typically, you create a facet on a particular column. The facet summarizes the cells in that column to give you a big picture on that column, and allows you to filter to some subset of rows for which their cells in that column satisfy some constraint. That's a bit abstract, so let's jump into some examples.

[More on faceting](https://github.com/OpenRefine/OpenRefine/wiki/Faceting)

* Scroll over to the *For What Committed* column.
* Click the down arrow and choose > Facet > Text facet.
* In the left margin, you'll see a box containing every unique, distinct value in the *For What Committed* column and how many times that value occurs in the column.
* Try sorting this facet by name and by count. Do you notice any problems with the data? What are they?
* Hover the mouse over one of the names in the Facet list. You should see that you have an "edit" function available. You could use this to fix an error immediately, and OpenRefine will ask if you want to make the same correction to every value it finds like that one (or not). But OpenRefine offers even better ways to find and fix these errors, which we'll use instead.

**Exercise**

Using faceting, find out how many years are represented in the dataset.

Is the column formatted as Number, Date, or Text? How does changing the format change the faceting display?

Which years have the most and least observations?

**Cluster**

In OpenRefine, clustering means "finding groups of different 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. Likewise, "Gödel" and "Godel" probably refer to the same person. Clustering is a very powerful tool for cleaning datasets which contain misspelled or mistyped entries. OpenRefine has several clustering algorithms built in. Experiment with them, and learn more about these algorithms and how they work.

* In the *For What* *Committed* Text Facet we created in the step above, click the *Cluster* button.
* In the resulting pop-up window, you can change the Method and the Keying Function. Try different combinations to see what different mergers of values are suggested.
* Select the *key collision* method and *metaphone3* keying function. It should identify three clusters. Click the *Merge?* box beside each, then click *Merge Selected and Recluster* to apply the corrections to the dataset.
* Try selecting different Methods and Keying Functions again, to see what new mergers are suggested. You may find there are still improvements that can be made, but don't Merge again; just Close when you're done. We'll now see other operations that will help us detect and correct the remaining problems, and that have more general uses besides.

[More on clustering](https://github.com/OpenRefine/OpenRefine/wiki/Clustering-In-Depth)

**Split**

If data in a column needs to be split into multiple columns, and the parts are separated by a common separator (say a comma, or a space), you can use that separator to divide up the bits into their own columns.

* Let us suppose we want to split the *Disposal* column into separate columns for repeated outcomes.
* Go to the drop-down tab at the top of the *Disposal* column. Choose Edit Column > Split into several columns.
* In the pop-up, in the Separator box, replace the comma with a forward slash (/).
* Remove the check in the box that says "Remove the column".
* Click "OK". You'll get some new columns called *disposal 1*, *disposal 2*, and so on.
* Notice that in some cases *disposal 1* and *disposal 2* are empty. Why is this? What do you think we should do about it?

**Undo / Redo**

It's common while exploring and cleaning a dataset to discover after you've made a change that you really should have done something else first. OpenRefine provides Undo and Redo operations to make this easy.

* Click where it says Undo / Redo on the left side of the screen. All the changes you have made so far are listed here.
* Click on the step that you want to go back to, in this case the previous step. The added columns will disappear.
* Notice that you can still click on the last step and make the columns reappear, and back, and forth.

**Trim Leading and Trailing Whitespace**

Strings with spaces at the beginning or end are particularly hard for we humans to tell from strings without, but the blank characters will make a difference to the computer. We usually want to remove these. OpenRefine provides a tool to remove blank characters from the beginning and end of any entries that have them.

* In the header for the column *disposal*, choose Edit cells > Common transforms > Trim leading and trailing whitespace.

# Lesson 3 – Filter, Exclude, Sort

## Filtering

There are many entries in admissions table. We can filter it to work on a subset of the data in the list for the next set of operations.

1. Select **For what commited** drop-down menu **> Text filter**. Note that a **For what commited** facet will appear on the left margin.
2. Type in **&** and press return. There are 86 matching rows of the original 4821 rows (and are selected for the subsequent steps).
3. At the top, change the view to **Show** 50 **rows**. This way you will see the first 50 rows.

## Challenge

What causes are selected by this procedure? How would you restrict this to one of the causes selected?

### Excluding entries

While we could type more letters of text, or click **case sensitive**, another way to filter is to **include** and/or **exclude** entries in a facet. If you still have your facet for **For what commited**, you can use it, or use drop-down menu **> Facet > Text facet** to create a new one. Only the names that agree with your **Text filter** will remain.

## Challenge

Use **include / exclude** to exclude one of the causes. Below are some suggested steps.

1. In the facet (left margin), click on one of the causes, such as \*Assault & Battery. Notice that when you click on the value, or hover over it, there are entries to the right for edit and include.
2. Click **include**. This will explicitly include this cause, and exclude others that are not expicitly included. Notice that the option now changes to **exclude**.
3. Click **include** and **exclude** on the other causes (**Homeless & Destitute**) and notice how the two entries appear and disappear from the table.

## Sort

You can sort the data by a column by using the drop-down menu in that column. There you can sort by **text**, **numbers**, **dates** or **booleans** (logical expressions). You can specify what order to put **Blanks** and **Errors**.

If this is your first time sorting this table, then the drop-down menu for the selected column shows : **> Sort...**. Select the way to sort (such as **numbers**)

## Challenge

Sort by month. How can you ensure that months are in order?

If you try to resort a column that you have already used, the drop-down menu changes slightly, to **> Sort** without the ..., to remind you that you have already used this column. It will give you additional options:

* **> Sort > Sort...**
* **> Sort > Reverse**
* **> Sort > Remove sort**

### Sorting by multiple columns.

You can sort by multiple columns by performing sort on additional columns. The sort will depend on the order in which you select columns to sort. To restart the sorting process with a particular column, check the **sort by this column alone** box in the sort pop-up menu.

## Challenge

Try sorting by a **year** after you have sorted by **month**. What happens to ordering?

Try sorting first by **year** and then by **month**. Be sure to check the **sort by this column alone** box when sorting by **year** to remove earlier sorts.

If you go back to one of the already sorted colunms and select **> Sort > Remove sort**, that column is removed from your multiple sort. If it is the only column sorted, then data reverts to its original order.

## Challenge

Sort by **year**, **month** and **day** in some order. Be creative: try sorting as **numbers** or **text**, and in reverse order (**largest to smallest** or **z to a**).

Use **> Sort > Remove sort** to remove the sort on the second of three columns. Notice how that changes the order.

# Lesson 4 – Numbers

## Numbers

When a table is imported into OpenRefine, all columns are treated as having text values. We saw earlier how we can sort interpreting column values as numbers, but this did not change the cells in a column from text to numbers.

Be sure to remove **Text filter** facets from the left margin so that we can examine the whole dataset.

To transform cells in the History Number column to numbers, use the column pulldown to **> Edit cells > Common transforms… > To number**. You will notice the History Number values change from left-justified to right-justified, and black to green color.

## Challenge

* Transform three more columns, including **Age** and **No of Reader**, from text to numbers.

### Numeric facet

Sometimes there are non-number values or blanks in a column and we want to find them. We can do that with a **Numeric facet**.

## Challenge

* For a column you transformed to numbers, edit one or two cells, replacing the numbers with text (such as abc) or blank (no number or text).
* Use the pulldown to **> Facet > Numeric facet**, which will add a facet to the left margin.
* Notice that there are several checkboxes in this facet: **Numeric**, **Non-numeric**, **Blank**, **Error**. Below these are counts of the number of cells. You should see checks for **Non-numeric** and **Blank** if you changed some values.
* Experiment with checking or unchecking these boxes to select subsets of your data.

When done examining the numeric data, remove this facet by clicking the **x** in the upper left corner of its panel.

## Scatterplot facet

Now that we have multiple columns as numbers, we can see how they relate to one another using the scatterplot facet. Select a numeric column, say Age, and use the pulldown menu to **> Facet > Scatterplot facet**. A new window called **Scatterplot Matrix** will appear. There are squares for each pair of numeric columns organized in an upper right triangle. Each square has little dots for the cell values from each row.

## Challenge

* Examine the scatterplots overall. Do the patterns make sense?
* If you have **Age** and **No of reader**, why does it have the pattern it does?

## Examine pair of columns in detail

We can examine one pair of columns by clicking on its square in the **Scatterplot Matrix**. A new facet with only that pair will appear in the left margin.

## Challenge

* Click in the scatterplot facet in the margin and drag to highlight a rectangle. This will subset the data to those entries.

## Challenge

* Click on the **Scatterplot Matrix** square for **recordID** and **period** to get that as a facet in the left margin.
* Redo the **> Text filter** on **scientificName** to reduce to **bai**.
* Notice the change in the scatterplot. It might be easier to see if you click **export plot** to put it on a new browser tab.

**Lesson 5 – Scripts**

**Scripts**

* OpenRefine saves every change, every edit you make to the dataset in a file you can save on your machine.
* If you had 20 files to clean, and they all had the same type of errors, and all files had the same columns, you could save the script, open a new file to clean, paste in the script and run it. Voila, clean data.
* In the Undo / Redo section, click Extract, save the bits desired using the check boxes.
* Copy the code and paste it into a text editor. Save it as a .txt file.

To run these steps on a new dataset, import the new dataset into OpenRefine, open the Extract / Apply section, paste in the .txt file, click Apply.

# Lesson 6 – Save and Export

## Saving and Exporting a Project

In OpenRefine you can save or export the project. This means you're saving the data and all the information about the cleaing steps you've done. Once you've saved a Project, you can open it up again and be just where you left off.

### Saving

By default OpenRefine is saving your project. If you close OpenRefine and open it up again, you'll see a list of your projects. You can click on any one of them to open it up again.

### Exporting

You can also export a project, for instance if you wanted to send it to someone else.

* Go to the 'Export' button in the top right. Click 'Export project'. This will save a compressed file that you can then open in OpenRefine that contains all the data and steps.

## Exporting Cleaned Data

Save your work when you are done by exporting it in the desired format. Save your files with meaningful names, no spaces. OpenRefine does not change your original dataset (hooray!).

* Go to 'Export' in the top right. Click on the file type you want to export the data in. 'Tab-separated values' (TSV) or 'Comma-separated values' (CSV) would be good choices.

That file will get exported to your default Download directory. That file can then be opened in a spreadsheet program or imported into programs like R or Python, which we'll be discussing later in our workshop. Remember from our lesson on Spreadsheets that using widely-supported, non-proprietary file formats like TSV or CSV improves the ability of yourself and others to use your data.

**Lesson 7 – Other Resources**

**Identify other Resources about OpenRefine.**

OpenRefine is more than a simple data cleaning tool. People are using it for all sorts of activities. Here are some other resources that might prove useful.

OpenRefine has its own web site with documentation and a book:

* [OpenRefine web site](http://openrefine.org/)
* [OpenRefine Documentation for Users](https://github.com/OpenRefine/OpenRefine/wiki/Documentation-For-Users)
* [OpenRefine documentation Wiki site](https://github.com/OpenRefine/OpenRefine/wiki/Documentation-For-Users)
* [Using OpenRefine](http://www.worldcat.org/title/using-openrefine-the-essential-openrefine-guide-that-takes-you-from-data-analysis-and-error-fixing-to-linking-your-dataset-to-the-web/oclc/889271264) book by Ruben Verborgh, Max De Wilde and Aniket Sawant
* [OpenRefine history from Wikipedia](https://en.wikipedia.org/wiki/OpenRefine)

In addition, see these other useful resources:

* [Grateful Data](https://github.com/scottythered/gratefuldata/wiki) is a fun sight with many resources devoted to OpenRefine, including a nice tutorial.
* [Margaret Heller](http://www.gloriousgeneralist.com/) shows how she uses OpenRefine for [Measuring and Counting Impact in Repositories](http://www.gloriousgeneralist.com/2014/12/notes-on-measuring-and-calculating-impact-in-institutional-repositories/).
* [Intersect Course Resources](http://www.intersect.org.au/course-resources) has Jared Berghold's **Cleaning & Exploring your data with Open Refine** (scroll down page to find).
* [Enipedia OpenRefine Tutorial](http://enipedia.tudelft.nl/wiki/OpenRefine_Tutorial)

There are more advanced uses of OpenRefine, such as bringing in column or cell data using web locators (URLs or APIs). The links above can give you a start on your journey.

**Challenge**

Visit one of these sites and share what you find with another person,