Making spreadsheets machine-readable

With Dave from ScraperWiki
About Me / Databaker

David McKee

dragon@scraperwiki.com

@dragondave on Twitter
Background

Databaker takes complicated published spreadsheets, and converts them into tabular CSV.

By way of example we will feature a project we have delivered for the UK’s Office of National Statistics (ONS).
The Problem

in Jan-Mar 2012, there were 1.09 million unemployed women aged between 16 and 64 in the UK (figures not seasonally adjusted)

How do you use this data?
How do we use this data?

- Too difficult! Keep the complicated shape
- Laborious, manual process to convert
- Complex, error-prone manual workflow
- Maybe it shouldn’t have been in a spreadsheet to start with… but too late now!
What goes wrong?

- Ad-hoc downstream mangling of data vs. approved data in readily usable format

- Analysis based on what’s easy, rather than what’s important

- Not easy to import into existing tools to inform decisions - slows decision making
Aim: convert human-readable spreadsheets into something truly machine readable
Beautiful, dangerous spreadsheets

- Focused on the printed page, not data reuse

- Need to migrate away to something else - genuinely tabular CSV is portable

- Specifically, portable to http://www.ons.gov.uk/ons/data/web/explorer
Break the problem down

- Identify cells which act as headers
  - Want to be robust against minor changes
    - especially inserted rows
  - Select each dimension’s headers in turn

- Identify cells which act as values
  - May want to do this from the intersection of headers

- Determine correct set of header cells for each value
  - Needs a simple but robust rule
Identifying cells

- ‘Bags’ of cells
  - non-contiguous
  - limited to a single sheet

- How to describe where they are?
  - Gender: all the cells which are ‘All Persons’, ‘Male’ or ‘Female’
  - Date: all the cells below ‘All Persons’, except blank, italic cells or the Gender cells we already described

```
gender = tab.filter(one_of(['Male', 'Female', 'All Persons']))
date = tab.filter('All Persons').fill(DOWN)
    .not_blank().not_italic().difference(gender)
```
What about the values?

- Non-contiguous
- Contains unwanted numbers
- Below indicators, right of dates
  
  \[
  \text{values} = \text{indicators}.\text{waffle}(\text{dates})
  \]
Matching values to dimension cells

- Good practice - **above** or to the **left**!

- Header either **directly** up/left of value or the **closest**
  - `date.dimension("date", DIRECTLY, LEFT)`
  - `indicator.dimension("indicator", DIRECTLY, ABOVE)`
  - `gender.dimension("gender", CLOSEST, ABOVE)"`
Selectors

- You can only select what you can describe
- Select by
  - text, formatting, borders, font, font size, cell ref.
- Combine groups of cells
  - set operators (union, difference, intersection)
  - get cells below this bag and right of another bag
- Get other cells nearby
  - relative movement (the cells one beneath these)
  - fill (all the cells to the right)
- All these transform a bag of cells into another bag of cells
## SUMMARY OF NATIONAL LFS DATA

### A02 Labour Force Survey Summary

<table>
<thead>
<tr>
<th>All aged 16 &amp; over</th>
<th>Total economically active</th>
<th>Total employed</th>
<th>Unemployed</th>
<th>Economically inactive</th>
<th>Economic Activity rate</th>
<th>Employment rate</th>
<th>Unemployment rate</th>
<th>Economic inactivity rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Persons</td>
<td>MGSL</td>
<td>MGSF</td>
<td>MGRZ</td>
<td>MGSC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jan-Mar 2010</td>
<td>49,637</td>
<td>31335244</td>
<td>6564932</td>
<td>28824552</td>
<td>2510652</td>
<td>18351663</td>
<td>6306539497243</td>
<td>58012369301</td>
</tr>
<tr>
<td>Jan-Mar 2011</td>
<td>50,054</td>
<td>31695579</td>
<td>6674312</td>
<td>25240308</td>
<td>2455271</td>
<td>18366545</td>
<td>633099643056</td>
<td>58405711174</td>
</tr>
<tr>
<td>Apr-Jun 2011</td>
<td>50,157</td>
<td>31768477</td>
<td>7742457</td>
<td>29265215</td>
<td>2493631</td>
<td>18397863</td>
<td>633192391228</td>
<td>585347567578</td>
</tr>
<tr>
<td>Jul-Sep 2011</td>
<td>50,248</td>
<td>31690970</td>
<td>8108037</td>
<td>29068520</td>
<td>2622496</td>
<td>18567304</td>
<td>630687736262</td>
<td>57849788822</td>
</tr>
<tr>
<td>Oct-Dec 2011</td>
<td>50,339</td>
<td>31799559</td>
<td>6244686</td>
<td>25128872</td>
<td>2670687</td>
<td>18536929</td>
<td>631705851602</td>
<td>57865201137</td>
</tr>
<tr>
<td>Jan-Mar 2012</td>
<td>50,431</td>
<td>31858843</td>
<td>3296458</td>
<td>25233403</td>
<td>2625439</td>
<td>18572001</td>
<td>631733294058</td>
<td>57967308849</td>
</tr>
<tr>
<td><strong>Change on qtr</strong></td>
<td><strong>92</strong></td>
<td><strong>59</strong></td>
<td><strong>105</strong></td>
<td><strong>-45</strong></td>
<td><strong>32</strong></td>
<td><strong>0.0</strong></td>
<td><strong>0.1</strong></td>
<td><strong>-0.2</strong></td>
</tr>
<tr>
<td><strong>Change %</strong></td>
<td><strong>0.2</strong></td>
<td><strong>0.2</strong></td>
<td><strong>0.4</strong></td>
<td><strong>-0.7</strong></td>
<td><strong>0.2</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Change on year</strong></td>
<td><strong>307</strong></td>
<td><strong>163</strong></td>
<td><strong>-7</strong></td>
<td><strong>170</strong></td>
<td><strong>203</strong></td>
<td><strong>-0.1</strong></td>
<td><strong>-0.4</strong></td>
<td><strong>0.5</strong></td>
</tr>
<tr>
<td><strong>Change %</strong></td>
<td><strong>0.7</strong></td>
<td><strong>0.5</strong></td>
<td><strong>0.0</strong></td>
<td><strong>0.9</strong></td>
<td><strong>1.1</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Male

| Jan-Mar 2010      | 24,236                   | 16903241       | 1945317    | 15362508             | 1540732             | 7344418         | 6973957347243  | 63382082765                       |
| Jan-Mar 2011      | 24,445                   | 17079751       | 7061117    | 15651748             | 1429003             | 7365462         | 69669121215   | 64027639655                       |
| Apr-Jun 2011      | 24,457                   | 17142690       | 7166934    | 15700433             | 1446557             | 7350376         | 69994321699   | 64092120343                       |
| Jul-Sep 2011      | 24,547                   | 17193711       | 4597125    | 15547732             | 1531998             | 7467857         | 698473414282  | 63373296343                       |
| Oct-Dec 2011      | 24,597                   | 17189421       | 1582946    | 15890246             | 1548180             | 7458413         | 696775065929  | 63383125922                       |
| Jan-Mar 2012      | 24,647                   | 17176013       | 1683123    | 15867024             | 1567293             | 7471439         | 69687671065   | 63677701516                       |
| **Change on qtr** | **51**                   | **38**         | **80**     | **-42**              | **13**               | **0.0**         | **0.2**          | **-0.3**                         |
| **Change %**      | **0.2**                   | **0.2**        | **0.5**    | **-0.7**             | **0.2**              |                 |                  |                                |
| **Change on year**| **202**                   | **96**         | **19**     | **78**               | **106**              | **-0.2**        | **-0.5**        | **0.4**                         |
| **Change %**      | **0.8**                   | **0.6**        | **0.1**    | **5.4**              | **1.4**              |                 |                  |                                |
Complications

- Headers in the wrong place - fake spans
  - ‘66’ needs to look left, ‘77’ right

- Real spans are not a problem
  - can just use the top-left cell
  - can also get the whole span as a bag of cells
Recipes

- Scripts to convert spreadsheets
  - Technical users who don’t consider themselves programmers can write them
  - All the power of Python

```python
from databaker.constants import *

def per_file(tables):  
    return """"

def per_tab(tab):  
    obs = tab.filter("MGSL").assert_one().shift(DOWN).fill(RIGHT).fill(DOWN).is_number().is_notItalic()

    tab.col('A').one_of(['Male', 'Female', 'All Persons']).dimension('gender', CLOSEST, ABOVE)
    tab.col('A').is_date().dimension(TIME, DIRECTLY, LEFT)
    tab.regex("All aged .+"').dimension('ages', CLOSEST, UP)
    tab.filter("Total economically active").fill(LEFT).fill(RIGHT).is_not_blank().dimension('indicator', DIRECTLY, ABOVE)

    tab.dimension('adjusted_yy', tab.name)
    return obs
```
Bigger Picture

- Critical task of correctly allocating headers done
- Training critical - they need to write their own recipes
- Lots of post processing to get identifiers consistent and correct
- Currently ONS-specific output format, intend to change
More information

- Databaker is open source
- Databaker documentation / downloads
  - [https://scraperwiki.github.io/eot-docs/](https://scraperwiki.github.io/eot-docs/)
- dragon@scraperwiki.com
- @dragondave on Twitter
Thank you!

- Wouldn’t have happened without:
  - UN OCHA & ONS - spreadsheet problems, £
  - Open Source
  - OKFN - libraries, organising csv,conf

- and you! for listening