The data that sits behind reports that have been produced to guide policy development in government is an important resource that many governments are beginning to make available through various initiatives like https://data.gov, https://data.gov.uk, https://data.gov.au, and https://data.gov.sg to name just a few. In many cases the data is available through the Comprehensive Knowledge Archive Network (CKAN) application programming interface (API).

In this chapter we explore how to make access this public data accessible via the CKAN API utilizing ckanr (Chamberlain, 2015) and then perform a simple analysis of the dataset.

The required packages for this chapter include:

```r
library(ckanr) # Access data from CKAN.
library(magrittr) # Use pipelines for data processing.
library(readr) # Modern data reader: read_csv().
library(dplyr) # Data wrangling: select().
library(jsonlite) # Manage JSON objects.
library(readxl) # Read Excel spreadsheets.
```

As we work through this chapter, new R commands will be introduced. Be sure to review the command’s documentation and understand what the command does. You can ask for help using the ? command as in:

```
?read.csv
```

We can obtain documentation on a particular package using the `help=` option of `library()`:

```
library(help=rattle)
```

This chapter is intended to be hands on. To learn effectively, you are encouraged to have R running (e.g., RStudio) and to run all the commands as they appear here. Check that you get the same output, and you understand the output. Try some variations. Explore.

Copyright © 2013-2015 Graham Williams. You can freely copy, distribute, or adapt this material, as long as the attribution is retained and derivative work is provided under the same license.
1 Introduction

Public and government datasets available through CKAN are a great source of open source data. Some of the available datasets contain data that underlies policy modelling. Having them available allows us to review the actual data that sits behind the policy decision making.

Here we get started using CKAN. After loading the ckanr package we can review the list of ckanr::servers() known to be offering up a CKAN API to their data holdings.

```r
servers()
## [13] "http://dartportal.leeds.ac.uk"
## [14] "http://data.bris.ac.uk/data"
## [16] "http://data.cityofsantacruz.com"
## [18] "http://data.glasgow.gov.uk"
## [19] "http://data.go.id"
## [21] "http://data.gov.br"
## [22] "http://data.gov.ie"
...'
```

For our purposes we will access a particular server. We can check which server is the default with ckanr::get_default_url(). To change to another server we set the environment variable CKANR_DEFAULT_URL using base::Sys.setenv().

```r
get_default_url()
Sys.setenv(CKANR_DEFAULT_URL="http://data.gov.au")
get_default_url()
```

The Australian server contains the public dataset that we will use to demonstrate access through the CKAN API.
2 Server Information

Now we can obtain meta data about the server itself and the extensions it supports.

```r
ckan_info()
## $ckan_version
## [1] "2.3.1b"
##
## $site_url
##
## $site_description
## [1] "Opening up government data"
##
## $site_title
## [1] "data.gov.au"
##
## $error_emails_to
## [1] "alex.sadleir@linkdigital.com.au"
##
## $locale_default
## [1] "en_GB"
##
## $extensions
## [1] "disqus"     "dga_stats"
## [3] "text_view"  "webpage_view"
## [5] "image_view" "recline_view"
## [7] "datasetstore" "datapusher"
## [9] "agls"      "datagovau"
## [11] "datagovau_hierarchy" "googleanalytics"
## [13] "resource_proxy" "spatial_metadata"
## [15] "spatial_query" "harvest"
## [17] "ckan_harvester" "csw_harvester"
## [19] "waf_harvester" "spatial_harvest_metadata_api"
## [21] "ga-report" "sitemap"
## [23] "sentry"    "cesium_viewer"
## [25] "wms_view" "kml_view"
## [27] "geojson_view" "officedocs_view"
## [29] "datajson_harvest" "viewhelpers"
## [31] "dashboard_preview" "linechart"
## [33] "barchart" "piechart"
## [35] "basicgrid" "recline_grid_view"
## [37] "recline_graph_view" "recline_map_view"
## [39] "pdf_view" "odata"
## [41] "zip_view"
```
3 Organizations

Many organizations may contribute resources (such as datasets) to a repository. Resources are organized into packages and packages are provided by an organization.

To check how many organizations are represented on the server we might first query for the `ckanr::organization_list()`. Here we request it be provided in the raw JSON format direct from the API call (as="json"). We transform the JSON data structure into an R data structure (a list in this case) and then `magrittr::extract2()` the result of the CKAN query. We save the result locally to avoid having to query the server again if we wanted the same data (as we do below).

```r
organization_list(as="json") %>%
  fromJSON(flatten=TRUE) %>%
  extract2('result') -> orgs
```

We can then report the number of organizations represented and the names of the columns contained in the organization table.

```r
nrow(orgs)
## [1] 171
names(orgs)
## [1] "image_display_url"       "display_name"          "description"
## [4] "title"                   "package_count"         "created"
## [7] "approval_status"         "is_organization"       "state"
## [10] "image_url"                "revision_id"            "packages"
## [13] "type"                    "id"                      "name"
```
4 Prettify Organizations

Using `jsonlite (Ooms et al., 2015)` we can display formatted JSON objects to make it considerably easier to review using `jsonlite::prettify()`. 

```r
orgs %>% toJSON() %>% prettify()
```

```json
# [  
#   {  
#     "display_name": "Australian Charities and Not-for-profits Commission",  
#     "description": "The independent national regulator of charities",  
#     "title": "Australian Charities and Not-for-profits Commission",  
#     "package_count": 5,  
#     "created": "2013-09-03T01:20:25.828188",  
#     "approval_status": "approved",  
#     "is_organization": true,  
#     "state": "active",  
#     "revision_id": "945f810c-e173-4a90-ba8f-70c9de6f69af",  
#     "packages": 5,  
#     "type": "organization",  
#     "id": "4d907503-b2aa-4c38-ac13-1273e791cb08",  
#     "name": "acnc"  
#   },  
#   {  
#     "image_display_url": "",  
#     "display_name": "ACT Government",  
#     "description": "",  
#     "title": "ACT Government",  
#     "package_count": 110,  
#     "created": "2015-09-16T03:52:10.428251",  
#     "approval_status": "approved",  
#     "is_organization": true,  
#     "state": "active",  
#     "image_url": "",  
#     "revision_id": "86c71d67-38e7-4569-b772-013b6f6d78",  
#     "packages": 110,  
#     "type": "organization",  
#     "id": "9ad7ef22-735a-45c1-9e83-0f702d56984b",  
#     "name": "act-government"  
#   },  
#   {  
#     "display_name": "Australian Institute of Health and Welfare",  
#     "description": "The Australian Institute of Health and Welfare (AI...  
#     "title": "Australian Institute of Health and Welfare",  
#     "package_count": 7,  
#     "created": "2013-05-12T09:15:15.624886",  
#     ....
```
5 Organizations and Packages

The dataset we are interested in has been provided by the Australian Taxation Office. Thus we want to find this organization in `org` dataset. A visual inspection of the dataset shows that the `title` column contains the appropriate strings for us to search.

```r
orgs %>%
  extract("title") %>%
  '=='("Australian Taxation Office") %>%
  which() %>%
  print() -> ato
## [1] 29
```

There are 12 packages available from the ATO.

```r
orgs[ato, 'packages']
## [1] 12
```

We will store the organization identification so as to investigate the packages further.

```r
oid.ato <- orgs[ato, 'id'] %>%
  print()
## [1] "90df1f57-c01f-4589-93bf-600dee01996e"
```

Out of interest we might select a few organizations before and after our organization of interest and list their number of packages.

```r
orgs %>%
  '[(seq(ato-5, ato+5),) %>%
  select(name, packages)
```

<table>
<thead>
<tr>
<th>#</th>
<th>name</th>
<th>packages</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>australianinstituteofcriminology</td>
<td>1</td>
</tr>
<tr>
<td>25</td>
<td>australianmuseum</td>
<td>0</td>
</tr>
<tr>
<td>26</td>
<td>australianpesticidesandveterinarymedicinesauthority</td>
<td>1</td>
</tr>
<tr>
<td>27</td>
<td>australianprudentialregulationauthority</td>
<td>15</td>
</tr>
<tr>
<td>28</td>
<td>australianpublicservicecommission</td>
<td>16</td>
</tr>
<tr>
<td>29</td>
<td>australiantaxationoffice</td>
<td>12</td>
</tr>
<tr>
<td>30</td>
<td>bioregional-assessment-programme</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>brisbane-city-council</td>
<td>71</td>
</tr>
<tr>
<td>32</td>
<td>bureauofinfrastructuretransportandregionaleconomics</td>
<td>18</td>
</tr>
<tr>
<td>33</td>
<td>bureauofmeteorology</td>
<td>40</td>
</tr>
<tr>
<td>34</td>
<td>bureauofresourcesandenergyeconomics</td>
<td>0</td>
</tr>
</tbody>
</table>
6 Packages

Packages contain resources (often datasets). We can list the packages and note that by default `ckanr::package_list()` will limit the query to just 31 packages. We also note the number of packages available over the whole repository.

```r
package_list(as="table")
## [1] "0-5m-contours"          ...  
## [2] "10m-contours"           ...  
## [3] "1-1-000-000-scale-australian-geoscience-map-sheet-index-national-geo...  
## [4] "1-100-000-scale-australian-geoscience-map-sheet-index-national-geosc...  
## [5] "1-250-000-scale-australian-geoscience-map-sheet-index-national-geosc...  
## [6] "19th-century-photographs-by-captain-samuel-sweet" ...  
## [7] "19th-century-photographs-by-ernest-gall" ...  
## [8] "19th-century-photographs-by-townsend-duryea" ...  
## [9] "1m-contours" ...  
## [10] "1-second-srtm-derived-hydrological-digital-elevation-model-dem-h-ver...  
## [12] "2001-02-to-2007-08-local-government-survey-victoria" ...  
## [14] "2007-community-attitudes-to-privacy-survey-data" ...  
## [16] "2008-assembly-election-summary-of-first-preference-votes-by-party-an...  
## [17] "2008-seismic-workstation-packages"  
## [18] "2008-tree-canopy-urbanforest-7d1bf" ...  
## [19] "2009-2059-act-population-projections" ...  
## [21] "2009-2059-act-population-projections-males-by-single-year-of-age" ...  
## [22] "2009-2059-act-population-projections-persons-by-single-year-of-age" ...  
## [23] "2009-green-light-report" ...  
## [25] "2011-seismic-workstation-packages"  
## [26] "2011-tree-canopy-urbanforest-adec6" ...  
## [27] "2012-offshore-petroleum-acreage-release-areas04479" ...  
## [28] "2013-14-austender-ict-contract-statistics"  
## [29] "2013-community-attitudes-to-privacy-survey-data" ...  
## [30] "2013-offshore-petroleum-acreage-release-areasf2a22" ...  
...  
package_list(as="table", limit=NULL) %>% length()
## [1] 7118
```
7 Finding a Specific Package

To illustrate the use of the CKAN API we will access the controversial Corporate Tax Transparency dataset released 17 December 2015 by the Australian Taxation Office.

Then we search for a particular package hosted on the server. We are interested in one provided by the ATO.

```r
package_search(q="australian_taxationoffice", as="json") %>%
  fromJSON(flatten=TRUE) %>%
  extract2('result') %>%
  extract2('results') -> pkgs.ato
names(pkgs.ato)
## [1] "license_title"  "maintainer"
## [3] "relationships_as_object"  "jurisdiction"
## [5] "temporal_coverage_to"  "private"
## [7] "maintainer_email"  "num_tags"
## [9] "geospatial_topic"  "id"
## [11] "metadata_created"  "spatial_coverage"
```

Let’s have a look at the titles to find the one we are interested in.

```r
pkgs.ato$title
# [1] "Corporate Tax Transparency"
# [2] "Cumulative Total Tax Returns Received"
# [5] "Taxation statistics – individual sample files"
# [7] "Ad-hoc requested data"
# [8] "Taxation Statistics 1999-00 to 2003-04"
#[10] "ATO Web Analytics July 2013 to April 2014"
```

```r
pkgs.ato$title %>%
grep('Transparency', .) %>%
extract(pkgs.ato$id, .) %>%
print() ->
pid.ato.trans
## [1] "c2524c87-cea4-4636-acac-599a82048a26"
```

So that provides us with the package identifier for the Corporate Tax Transparency package from the ATO.
8 Resources

We can then list the contents of the package.

```r
pkg.ato.trans <- package_show(pid.ato.trans, as="table") %>% print()
```

```r
## $license_title
## [1] "Creative Commons Attribution 3.0 Australia"
##
## $maintainer
## [1] "allanbarger"
##
## ....
```

This contains quite a bit of meta data.

```r
names(pkg.ato.trans)
```

```r
## [1] "license_title"     "maintainer"    
## [3] "relationships_as_object" "jurisdiction"  
## [5] "temporal_coverage_to" "private"       
## [7] "maintainer_email" "num_tags"       
## [9] "geospatial_topic" "id"            
## [11] "metadata_created" "spatial_coverage" ....
```

```r
pkg.ato.trans$name
```

```r
## [1] "corporate-transparency"
```

```r
pkg.ato.trans$metadata_created
```

```r
## [1] "2015-12-08T02:57:38.485976"
```

```r
pkg.ato.trans$license_title
```

```r
## [1] "Creative Commons Attribution 3.0 Australia"
```

```r
pkg.ato.trans$license_id
```

```r
## [1] "cc-by"
```

```r
pkg.ato.trans$type
```

```r
## [1] "dataset"
```

```r
pkg.ato.trans$num_resources
```

```r
## [1] 1
```

We see there is just 1 resource—this is a package containing a single dataset.
9 Resource Information

We can list some information about the dataset. The description reads:

This report contains the total income, taxable income and tax payable of over 1500 public and foreign private entities for the 2013-14 income year.

```r
names(pkg.ato.trans$resources)
## [1] "cache_last_updated" "package_id"
## [3] "webstore_last_updated" "datastore_active"
## [5] "id" "size"
## [7] "wms_layer" "state"
## [9] "hash" "description"
## [11] "format" "tracking_summary"
```

```r
pkg.ato.trans$resources[1,]$name
```

```r
pkg.ato.trans$resources[1,]$format
## [1] "XLSX"
```

```r
pkg.ato.trans$resources[1,]$url
## [1] "http://data.gov.au/dataset/c2524c87-cea4-4636-acac-599a82048a26/resou...
```
10 Download the Resource (Dataset)

From the supplied URL which points to an Excel spreadsheet we will want to extract the data of interest. Trial and error informs us that the second sheet of the spreadsheet contains the data of interest, though it contains three tables so we need to restrict the data to just the first table on that sheet.

In the following code block we record the url and the XLSX file name to create a temporary file into which we store the downloaded dataset. We then `utils::download.file()` and use `readxl::read_excel()` to save it into `transparency`. The `base::unlink()` does the house keeping to remove the downloaded file.

```r
url <- pkg.ato.trans$resources[1,]$url
temp <- tempfile(fileext=".xlsx")
download.file(url, temp)
transparency <- read_excel(temp, sheet=2, skip=1)[1:1540, 2:6]
unlink(temp)
```

We now have a dataset supplied by a CKAN server available to us in R in a usable form as a data frame.

```r
dim(transparency)
## [1] 1540  5

names(transparency)
## [1] "Name"   "ABN"   "Total Income $"
## [4] "Taxable Income $" "Tax Payable $"

head(transparency)
## Source: local data frame [6 x 5]
##
## Name     ABN Total Income $
## (chr)    (dbl)    (dbl)
## 1 3M AUSTRALIA PTY LTD 90000100096 452892659
## 2 A P EAGERS LIMITED 87009680013 2561960532
## 3 A2 AUSTRALIAN INVESTMENTS PTY LTD 93126014275 110173384
## 4 AAPC LIMITED 87009175820 570890761
## 5 ABACUS GROUP HOLDINGS LIMITED 31080604619 198340272
## 6 ABB GROUP INVESTMENT MANAGEMENT PTY LTD 47082803852 1173330692
##
```

We now have a dataset supplied by a CKAN server available to us in R in a usable form as a data frame.
11 Data Wrangling

ds <- transparency
names(ds) <- c("name", "abn", "total", "taxable", "payable")
ds %<>% mutate(per=100*payable/total, rate=100*payable/taxable)
12 Command Summary

This chapter has introduced, demonstrated and described the following R packages, functions, commands, operators, and datasets:
13 Exercises
14 Further Reading

The Rattle Book, published by Springer, provides a comprehensive introduction to data mining and analytics using Rattle and R. It is available from Amazon. Other documentation on a broader selection of R topics of relevance to the data scientist is freely available from http://datamining.togaware.com, including the Datamining Desktop Survival Guide.

This chapter is one of many chapters available from http://HandsOnDataScience.com. In particular follow the links on the website with a * which indicates the generally more developed chapters.

We identify below other resources that augment the material we have presented in this chapter.

-
15 References


