Desk Instructions

For the COVID-19 Management Information Weekly Update

Miles Drake

Victoria Avila

Tom Crines

2021-09-28

Table of Contents

[1 Before you start… 1](#_Toc83722766)

[2 Weekly update 2](#_Toc83722767)

[3 SPARQL queries 8](#_Toc83722768)

[4 Understanding the R script 12](#_Toc83722769)

[5 Troubleshooting 14](#_Toc83722770)

[6 Project setup (GitHub) 16](#_Toc83722771)

[7 Reporting issues 17](#_Toc83722772)

# Before you start…

Brief documentation on the various parts of the R code can be found later in this document. The R code has been made to be robust. Only on very rare occasions does anything need to be changed.

If a data set produces an error, the issue can usually be fixed by making a small change to the project’s configuration tables.

Barring exceptional circumstances, the R code should never need to be changed.

# Weekly update

This section details the weekly update process. It assumes that you already have the project’s GitHub repository downloaded and correctly set up. If it is your first time maintaining this project, or you need to set up the GitHub repository, please read the next section.

## Step 1: Pulling the latest version of the GitHub repository

This step is only necessary if you want to use Git Bash. Using Git Bash is strongly recommended. You can skip it if you are going to upload files on to GitHub manually.

If you aren’t working in a project cloned from GitHub, please follow the instructions in the PROJECT SETUP section before proceeding.

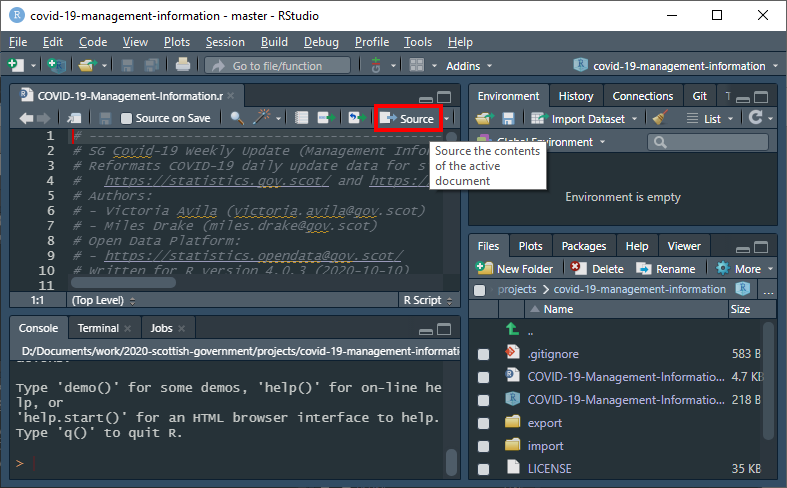
1. Open the project in RStudio: COVID-19-Management-Information.Rproj.
2. In the RStudio Terminal tab, run git pull origin master.



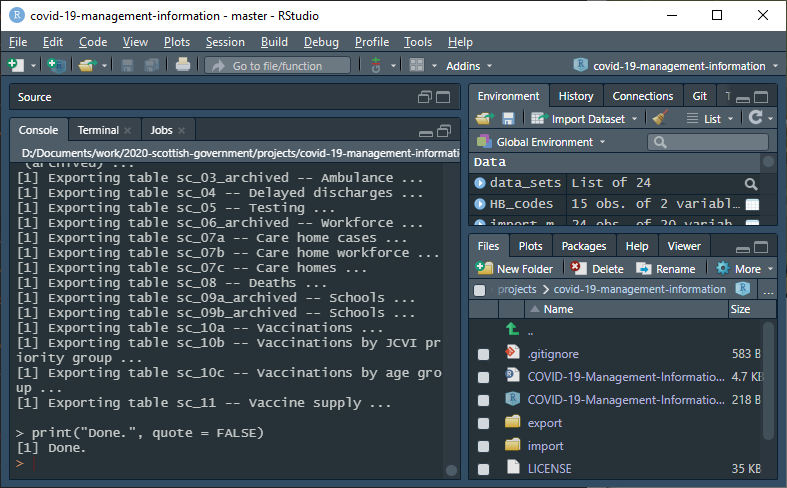
This will update all files in the project to the latest version uploaded to GitHub.

## Step 2: Running the R code

1. Open the main R script: COVID-19-Management-Information.r.
2. Click “Source” to run the R code.



The console will output its progress as it converts each Excel worksheet to CSV format. You will be notified when the script has successfully finished running.



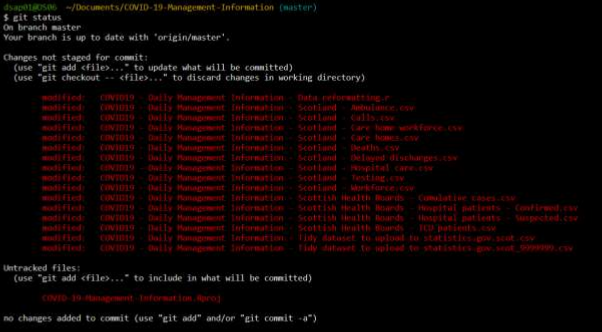
If you double click on the main script, RStudio will open with the folder the script is as the working directory.

The scripts/ folder contains R scripts that are called by the main script. They will not work when run stand-alone.

## Step 3: Uploading the new data sets to GitHub

All the output files should have been created in the export/ folder.

1. Run git status in the Terminal to confirm.

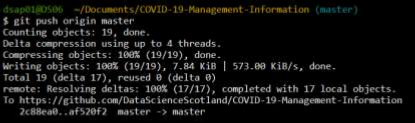


1. Run git add . (git add period) to stage the changes.
2. Run git commit -m "type your own commit message here" to commit the changes.



The commit message will appear in the git history, and will show next to the files when viewing on GitHub.

1. Run git push origin master to push the files from your local repository to GitHub.



The changes should then be visible on the remote repository.

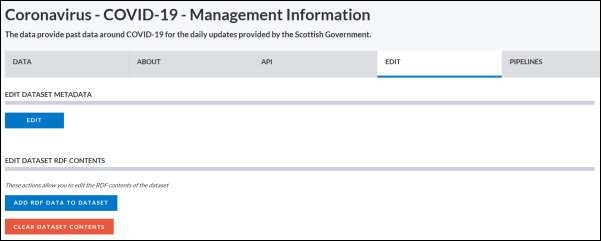
## Manually the new data sets to GitHub

Alternatively (though not recommended), you can upload the new data sets to GitHub through GitHub’s web interface.

1. Go to the GitHub folder:  
   https://github.com/DataScienceScotland/COVID-19-Management-Information
2. Click on “Upload files” and select all the files to upload.
3. Scroll down to the bottom of the page and click on “Commit changes”.

## Step 4: Uploading the new data sets to statistics.gov.scot

1. Navigate to statistics.gov.scot admin site and log in:  
   <https://pmd3-production-admin-sg.publishmydata.com/admin>
2. Go to the Covid-19 – Management Information dataset:  
   <https://pmd3-production-admin-sg.publishmydata.com/resource?uri=http%3A%2F%2Fstatistics.gov.scot%2Fdata%2Fcoronavirus-covid-19-management-information>
3. Click on “EDIT” tab and then on “CLEAR DATASET CONTENTS”

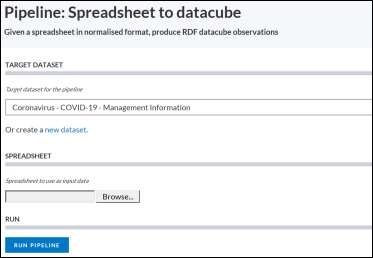


Once you make any changes to the dataset, the system will automatically create a new draft called “Untitled”. You can rename it if you like, but since this is the only one you will have in your accounts and it"s going to be used for 30 minutes, you can leave it unchanged.

1. Click on the “PIPELINES” tab and then on the first type of pipeline – Spreadsheet to datacube.



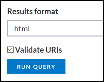
1. Select the “Coronavirus – COVID-19 – Management Information” dataset as the target dataset and export/upload-to-open-data-platform.csv as the input data.



1. Run the pipeline.

## Step 5: Quality Assurance for statistics.gov.scot

1. Click on the dataset and go to the “API” tab. Check the number of observations under the “DATA LINKED RESOURCES” section.
2. Download the whole dataset as “CSV” and compare the number of observations. Numbers should match.
3. Select a slice of the dataset and check it downloads fine.
4. Go to “TOOLS/SPARQL Query” and run the SPARQL queries in the section below, one query at a time. Make sure you check “Validates URIs”.



1. All the queries should give no results and all the URIs should come up in green.



## Step 6: Publish on statistics.gov.scot

1. Publish the draft by clicking on Publish at the top of the window.





# SPARQL queries

Please note that the following code spills off the side of the page. This is intentional. This allows the code to be copied and pasted without breaking the line structure.

The SPARQL queries can also be found in plain text format in the project repository, in the docs folder.

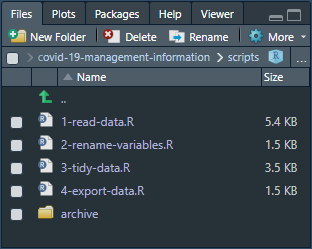
# 1. Identifies any areas not in Atlas  
# ----------------------------------------------------------------------------------  
PREFIX qb: <http://purl.org/linked-data/cube#>  
  
select distinct ?area where {graph <http://statistics.gov.scot/graph/coronavirus-covid-19-management-information> {  
?obs a qb:Observation ;  
<http://purl.org/linked-data/sdmx/2009/dimension#refArea> ?area .  
}  
OPTIONAL {?area <http://publishmydata.com/def/ontology/foi/memberOf> ?collection .}  
FILTER (!bound(?collection))  
}  
  
  
# 2. Identifies any archived geographies  
# ----------------------------------------------------------------------------------  
PREFIX qb: <http://purl.org/linked-data/cube#>  
  
select distinct ?area where {graph <http://statistics.gov.scot/graph/coronavirus-covid-19-management-information> {  
?obs a qb:Observation ;  
<http://purl.org/linked-data/sdmx/2009/dimension#refArea> ?area .  
}  
?area <http://statistics.data.gov.uk/def/statistical-geography#status> "Archived"  
}  
  
  
# 3. Identifies any observations with multiple values - count  
# ----------------------------------------------------------------------------------  
PREFIX qb: <http://purl.org/linked-data/cube#>  
  
SELECT ?DataSet ?s (count(?s) as ?NumValues)  
WHERE {  
?s <http://statistics.gov.scot/def/measure-properties/count> ?o.  
?s qb:dataSet <http://statistics.gov.scot/data/coronavirus-covid-19-management-information>.  
}  
GROUP BY ?DataSet ?s  
HAVING (?NumValues>1)  
  
  
# 4. Identifies any observations with multiple values - ratio  
# ----------------------------------------------------------------------------------  
PREFIX qb: <http://purl.org/linked-data/cube#>  
  
SELECT ?DataSet ?s (count(?s) as ?NumValues)  
WHERE {  
?s <http://statistics.gov.scot/def/measure-properties/ratio> ?o.  
?s qb:dataSet <http://statistics.gov.scot/data/coronavirus-covid-19-management-information>.  
}  
GROUP BY ?DataSet ?s  
HAVING (?NumValues>1)  
  
  
# 5. Identifies multiple labels for units  
# ----------------------------------------------------------------------------------  
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  
  
SELECT ?MeasureUnits (count(?MeasureUnits) as ?NumLabels)  
WHERE {  
?MeasureUnits a <http://purl.org/linked-data/sdmx/2009/concept#unitMeasure>.  
?MeasureUnits rdfs:label ?label .  
}  
GROUP BY ?MeasureUnits  
HAVING (?NumLabels>1)  
  
  
# 6. Identifies multiple dimension values  
# ----------------------------------------------------------------------------------  
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  
  
SELECT ?DimensionValue (count(?DimensionValue) as ?NumLabels)  
WHERE {  
?DimensionValue a <http://www.w3.org/2004/02/skos/core#Concept>.  
?DimensionValue rdfs:label ?label .  
}  
GROUP BY ?DimensionValue  
HAVING (?NumLabels>1)  
  
  
# 7. Identifies duplicate concept schemes  
# ----------------------------------------------------------------------------------  
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  
  
SELECT ?ConceptScheme (count(?ConceptScheme) as ?NumLabels)  
WHERE {  
?ConceptScheme a <http://www.w3.org/2004/02/skos/core#ConceptScheme>.  
?ConceptScheme rdfs:label ?label .  
}  
GROUP BY ?ConceptScheme  
HAVING (?NumLabels>1)  
  
  
# 8. Identifies duplicate values in dataset  
# ----------------------------------------------------------------------------------  
PREFIX qb: <http://purl.org/linked-data/cube#>  
  
SELECT ?DataSet ?s (count(?s) as ?NumValues)  
WHERE {  
?s <http://statistics.gov.scot/def/measure-properties/index> ?o.  
?s qb:dataSet ?DataSet.  
}  
GROUP BY ?DataSet ?s  
HAVING (?NumValues>1)  
LIMIT 100  
  
  
# 9. Identifies any datasets which have dropped dimensions:  
# ----------------------------------------------------------------------------------  
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  
  
SELECT \*  
WHERE {  
?s <http://purl.org/linked-data/cube#dimension> ?x.  
FILTER( !EXISTS { ?x rdfs:label ?y.} )  
}  
  
  
# 10. Checks for missing Units  
# ----------------------------------------------------------------------------------  
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  
  
SELECT distinct ?unit ?unit\_label  
WHERE {  
?s <http://purl.org/linked-data/sdmx/2009/attribute#unitMeasure> ?unit .  
OPTIONAL {?unit rdfs:label ?unit\_label }  
FILTER(!bound(?unit\_label))  
}

# Understanding the R script

The R script has been designed to be robust. It does its best to cope with the (at times chaotic) data sets.

Most of the tables contained within the data set are very similar. Rather than have a separate script for each table, the master R script does the following:

1. Download the latest versions of the two data sets (whole-of-Scotland and Health Boards).  
   It will download the latest versions automatically, based on your system clock.
2. Load the rules for each table contained within the data set from the following files:  
   import/data-set-rules.csv  
   import/data-set-structure.csv
3. Using the variables given in the rules tables, run a for() loop through the R script, one for each table.  
   The scripts run can be found in the scripts/ folder.



1. Combine all of the tables (the whole data set) into one large table, for uploading to statistics.gov.scot.
2. Export all of the tables to:  
   export/  
   The whole data set is exported to:  
   export/upload-to-open-data-platform.csv

The R script has been commented with the aim to explain what each section of the script does.

Because the R script is run as a loop, remember that any change to the R script applies to every table in the data set.

## Observing the data

Each table can be found under the data\_sets list.

* Metadata is attached to each list to allow you to identify what each table is.  
  Metadata can be found under data\_sets > table\_name > metadata.
* Data can be found under data\_sets > table\_name > data.

## The settings tables

The settings – which determine the rules for importing each table – are stored as CSV files. They are found in the import/ folder. The key tables are:

1. data-set-rules.csv, which contains the importing rules and flags for each table.  
   These rules tell R how to interpret each table. This means that the R script can be rather small, and (hopefully) easier to understand. It also means that any change to the R script applies to all tables.
2. import/data-set-structure.csv, which contains the names of the variables / columns.  
   For each variable, its original name and new name is provided. The data type is specified to allow the script to perform sanity checking and QA on that variable. You can also define columns to be ignored with skip. Skipped columns are still checked for existence, which helps confirm that the table hasn’t changed.

You can modify these settings tables to add new tables, or fix problems with the rules for importing the existing tables.

# Troubleshooting

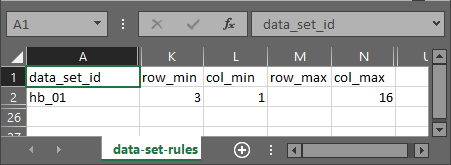
## Changing a table’s size, dimensions, or range

The most common error involves a table’s dimensions changing. This is usually because the maintainer of the data set has added a comment beside (or even inside) the table. R typically interprets this as more data, and will attempt to expand the table to suit.

Because this could silently induce errors in the table for upload to statistics.gov.scot, columns and dimensions must now be strictly defined. This is intended behaviour, to ensure the integrity of the data.

If the dimensions for each table found in the data set do not match what is defined in data-set-rules.csv, the R script will stop with an error.

In the example below, the table hb\_01 starts at row 3, column 1. It has 16 columns. It has no defined maximum rows, so it will fetch the data from all rows.



If a table’s dimensions are changed, modify its rules in data-set-rules.csv accordingly.

## Changing a table’s variable names

The table import/data-set-structure.csv contains all of the variables found in each data set.

This is a large table, but it allows the R script to automatically rename variables and confirm that column names and types have not changed.

## Adding new variables

1. Open import/data-set-rules.csv.
2. For the chosen table, increase the value of col\_max accordingly.
3. Save import/data-set-rules.csv.
4. Open import/data-set-structure.csv.
5. Add new rows containing the column names (see example below).
6. Save import/data-set-structure.csv.
7. Run (source) the R script.

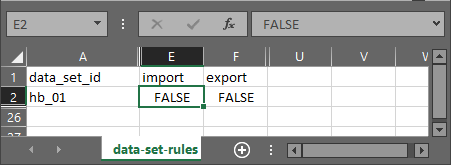
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| data\_set\_id | col\_id | col\_name\_old | col\_name\_new | col\_type |
| example | 11 | Old column name (from Excel) | New column name (for output) | numeric |

If the number of columns defined in import/data-set-structure.csv does not match the number of columns found, or the names of the columns do not match, R will produce an error. This is intended behaviour, to ensure the integrity of the data.

## Disabling a table entirely

If you can not fix a table, you can change its entry in data-set-rules.csv.

1. Open data-set-rules.csv.
2. Scroll to the table to be disabled.
3. Change the variables import and export for that data set to FALSE.
4. Save your changes.
5. Run (source) the R script again.



# Project setup (GitHub)

If you have an existing “COVID-19-Management-Information” project, it would probably be best to delete it before continuing. This will avoid the confusion of having two projects with the same name.

1. Open RStudio.
2. From the top menu, select File > New Project > Version Control > Git
3. In the pop-up window, copy “https://github.com/DataScienceScotland/COVID-19-Management-Information” into the Repository URL field.
4. Project directory name will auto-populate.
5. Browse to the directory you want to create the project in for “Create project as a subdirectory of”
6. Tick open in a new session
7. Click create project

# Reporting issues

**Reporting issues with the R script**

* Miles Drake - miles.drake@gov.scot

**Reporting issues with the data published on gov.scot**

* Catriona Hayes - Catriona.Hayes@gov.scot
* Neil Grant - Neil.Grant@gov.scot
* Web team - WEBSITE@gov.scot

**Reporting issues with uploading data to statistics.gov.scot**

* Bill Roberts - support@swirrl.com

**Reporting issues with GitHub**

* Joseph Adams - Joseph.adams@nrscotland.gov.uk