cBdfd_newstarterguide

March 25, 2023

0.1 Connected Bradford: new starter guide

ICYMI

What is an FDM?

Where do I find my Vertex Workbench and Jupyter Labs?

- Navigate to Vertex Workbench
- Scroll down, find your name, which will be *firstintial-surname-workspace* and click **OPEN JUPYTERLAB** This will take you to **JupyterLabs** where you will be able to setup and access the Connected Bradford databsse via R, Python or SQL.

0.1.1 Jupyter Notebooks and getting started

OK, you're in!

Now what?

This will depend if you plan to use R or Python within Jupyter. Let's start with R...

Using R in Jupyter Before you start using R to query Connected Bradford datasets, you'll need to set up an R environment in this Vertex workspace. Luckily, Sam Relins has created a handy setup guide to walk you through this process, which you can download and follow here

OK, your environment is set and activated (you'll need to activate it everyt time you login) and we're ready to go after opening your newly created R environment (pg9 of Sam's guide).

The first thing you'll notice is that Jupyter isn't quite as pretty or helpful as RStudio. All is OK though and we can still run our normal R scipts here. Here's how you to get started and connected to BigQuery...

[4]:
```
# Start by pulling in bigrquery, which is R's package to run queries via␣
↪BigQuery
library(bigrquery)
```

How do I see all of the tables in one dataset?

[5]:
```
# How do I see all of the tables in one dataset?
# Let's use our Lookup dataset as an example
```

```
# Store the project ID
project_id = "yhcr-prd-phm-bia-core" # this is always the same for all␣
 ↪Connected Bradford projects

# Example dataset - change to whichever dataset you want to query
targetdb1 <-'yhcr-prd-phm-bia-core.CB_LOOKUPS'
targetdb1 <-gsub(' ','',targetdb1)
print (targetdb1)
# Run
```

```
[1] "yhcr-prd-phm-bia-core.CB_LOOKUPS"
```

[6]:
```
# So we've found the dataset you're looking for, now to list the tables

tables = bq_dataset_tables(targetdb1)
#renove the hash below to list the tables
tables
# Run
```

```
[[1]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_AE_CodeToSnomed_lookup

[[2]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_BNF_DMD_SNOMED_lkp

[[3]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_CTV3Codes_Lookup

[[4]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_CTV3ToSnomed_Map

[[5]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_Dataset_ExtractDateRef

[[6]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_EFI_Codelist

[[7]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_IMD_2019

[[8]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_IMD_by_LSOA

[[9]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_LSOA_AREA

[[10]]
```

```
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_LSOA_to_Ward

[[11]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_OpenSafelyCategory

[[12]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_eFI2_SNOMED_CT_012

[[13]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_lsoa_boundaries

[[14]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_lsoa_to_msoa

[[15]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_lsoa_to_ward

[[16]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_msoa_boundaries

[[17]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_person_id_fixer

[[18]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_person_lsoa

[[19]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.tbl_ward_boundaries

[[20]]
<bq_table> yhcr-prd-phm-bia-core.CB_LOOKUPS.vw_DemoGraphics
```

How do I print a table?

[9]:
```r
# There you go. Now we just need to query the table you're after

sql3 <-paste('select * from ',targetdb1,'.tbl_CTV3ToSnomed_Map limit 10 ;', sep
  ↪= "")
#This runs it
tb3 <- bq_project_query(project_id, sql3)

#This loads  it into an R data frame
table <- bq_table_download(tb3)
#This displays it
table
# Run
```

A tibble: 10 × 2

CTV3Code <chr>	SNOMEDCode <chr>
XaBr1	111002
7N104	111002
75161	115006
Xa7Hc	117003
F4360	122003
X00cp	122003
XS7sD	128004
XE0Xz	140004
H121z	140004
F0070	151004

How do I save my new dataframe to BigQuery?

[ ]:
```
#OK, let's say the 'table' example above is a unique dataframe you've just␣
 ↪created
# You'll need to change the project space name below to wherever you want to␣
 ↪move the dataftame to
# "table" is the name of your created dataframe
devtools::install_github("rstats-db/bigrquery")
insert_upload_job("yhcr-prd-phm-bia-core", "INSERT PROJECT SPACE NAME HERE",␣
 ↪"table", stash)
# Run
```

[ ]:
```
# The above will install some packages via GitHub. Vertex doesn't always like it
# If Vertex delays then run the below in addition - that should do it
install.packages("remotes")
remotes::install_github("rstats-db/bigrquery")
# Run
# You should now be able to query your dataframe as a table via BigQuery as we␣
 ↪oulined earlier
# Enjoy!
```

Useful R resources Luckily Google has provided several useful guides that it continues to update. The below link also walks through how to get started with R and BigQuery:

Use R with BigQuery

Saving dataframes from R to BigQuery

Other resources bigQueuryR

Querying in the Cloud

Using Python in Jupyter There a couple of ways to access BigQuery via Python. One is using google.cloud.bigquery() to run SQL syntax, and the second is using Python's magic-cell functionality.

Let's run through both below.

0.1.2 Option 1: google.cloud.bigquery()

Note: the below is a script that can be copied into a Python notebook. It can't be executed here, as this guide is being produced as an R Markdown file.

Import libraries. from google.cloud import bigquery

Instantiate bigQuery client. client = bigquery.Client()

The bigquery client will accept the SQL syntax as a string. For example:

Define string of SQL syntax

my_sql_syntax = " " SELECT year_of_birth FROM yhcr-prd-phm-bia-core.CY_MYSPACE_CMC.person WHERE year_of_birth IS NOT NULL LIMIT 5 " "

Run SQL string and store in a pandas.DataFrame

client_query_output = client.query(my_sql_syntax).to_dataframe() client_query_output

Output will look like this

year_of_birth | ————-- | 1900 | 1900 | 1900 | 1900 | 1900 |

0.1.3 Option 2: Magic Cells

Note: the below is a script that can be copied into a Python notebook. It can't be executed here, as this guide is being produced as an R Markdown file.

%load_ext google.cloud.bigquery

New cell

%%bigquery

SELECT

gender,

COUNT(person_id) as person_count

FROM `yhcr-prd-phm-bia-core.CY_MYSPACE_RS.asc_person_demos`

GROUP BY gender

ORDER BY gender

Output will look like this

gender	person_count
FEMALE	27888
MALE	21751
UNKNOWN	2

Note: to save the query as a dataframe just type the name of your data frame after %%bigquery. e.g. %%bigquery python_demo (you will not see a table output when you execute this

Useufl Python resources Google: IPython Magic

Google: Visualize BigQuery data in Jupyter notebooks

Working with BigQuery and a Python Notebook