Tutorial 1. Getting the data.

The purpose of this tutorial is to give you experience of getting data via BigQuery. We will export our data and push it into git for use with Google Colab, however, we can use BigQuery directly. We will not go into using BigQuery directly with Google Colab.

1.0

Let’s go to BigQuery using you UHI google account: <https://console.cloud.google.com/home>

Now, scroll down on the left hand menu and click on BigQuery:

A screenshot of a cell phone

Description automatically generated

You should now see something like the following if you have done Tutorial 0:

A screenshot of a social media post

Description automatically generated

In the left hand pane, click on “+ Add Data”:

A screenshot of a cell phone

Description automatically generated

Then explore public datasets:

A screenshot of a cell phone

Description automatically generated

Type in “new york taxi trips” into the search bar at the top:

A screenshot of a social media post

Description automatically generated

Click on the NYC TLC Trips result:

A screenshot of a cell phone

Description automatically generated

Then click on “VIEW DATASET”:

A screenshot of a cell phone

Description automatically generated

As you can see, you have access to all public datasets, if you scroll up, you can close this and you will see your uhi-project:

A picture containing knife

Description automatically generated

1.1

The good thing is, this will allow you to have all of the data you need for your assignment as well.

Let’s discuss why we are doing this. We want to find out if weather has an effect on the number of taxi trips taken in New York City. To carry out this analysis, we need all of the taxi trips taken in New York and the weather data per day over the same period of time. Thankfully we have access to both.

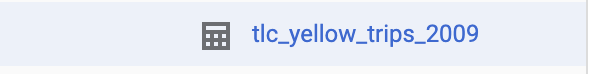
**For the purposes of the tutorial, we are only going to use 2009 and make an assumption about where the weather is measured from i.e. Central Park. In the assignment, it is ok to use Central Park for the weather data, but the date range will be much larger.**

Now, why don’t we explore the data we need.

The Taxi data is actually stored here:

bigquery-public-data > new\_york > tlc\_yellow\_trips\_2009

If you select it



You will get the following:

A screenshot of a social media post

Description automatically generated

As you can see at the bottom of the page, is a description of the data. A lot of these things we won’t particularly need for this example, but it is good to have a look through the different types.

Let’s try a query:



This means, select everything from the tlc\_yellow\_trips\_2009 but only show me 10. Notice the backticks around the table used and the use of dots between the different parts of the structure i.e.

`bigquery-public-data.new\_york.tlc\_yellow\_trips\_2009`

Now, let’s “Run” this query:

A picture containing ball, person, player, drawing

Description automatically generated

And we get:

A screenshot of a cell phone

Description automatically generated

You can’t see all of the columns, but you can move around in here to see some more of the data.

Now, let’s have a look at the weather data. Luckily for us, it’s in here too.

bigquery-public-data > noaa\_gsod > gsod2009

There are many years of weather data in here:

A screenshot of a cell phone

Description automatically generated

Even as far back as the 1929. Again, selecting this you will get some information about the dataset.

A screenshot of a cell phone

Description automatically generated

Now, let’s have a look at the data:



This query will take a while and will return weather for the whole of America. We want to narrow it down to Central Park in New York. Thankfully, I did this for you:



As you will see, you get 365 days returned, so the weather for each day of that year:

A picture containing table, drawing

Description automatically generated

1.2 Collate the data

What we need to do, for our little example, is collate the weather data for each day and find the number of taxi trips for each day. We also want to add a day of the week numerical field (1 Monday – 7 Sunday) as it is likely that the day of the week will have an influence on taxi usage.

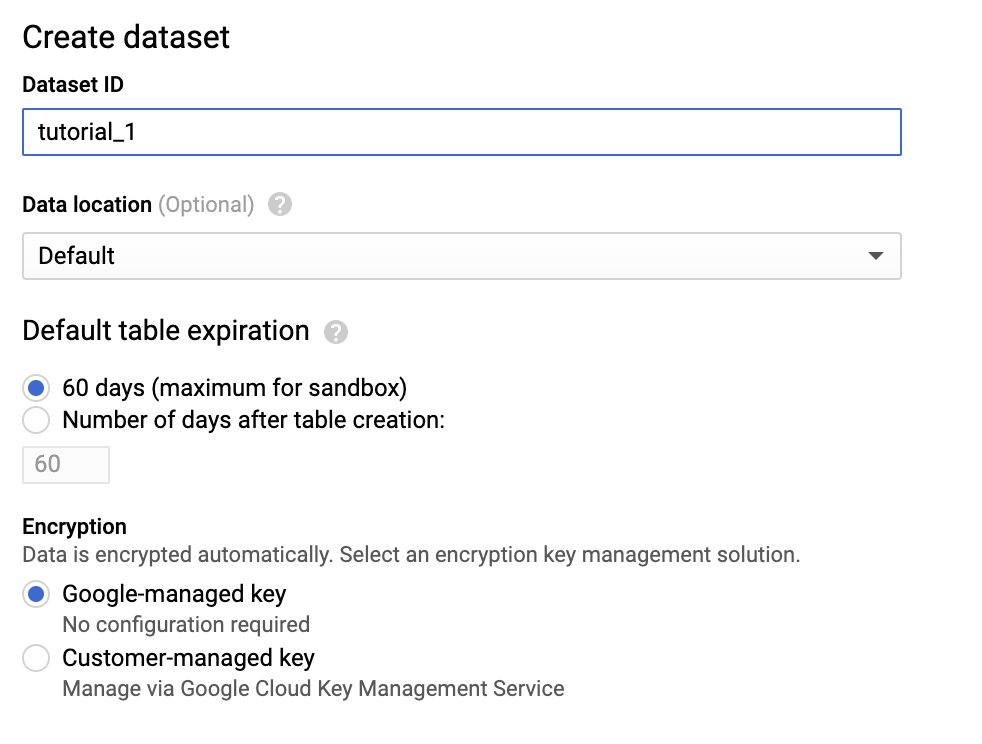
Click on uhi-project:



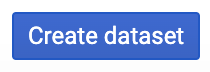
Next, click on



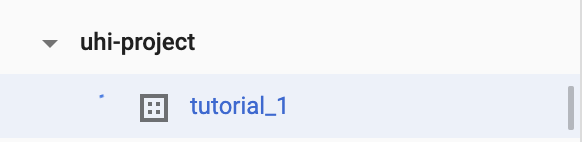
In the side drawer, add a Dataset ID (you will need to make another for the assignment):



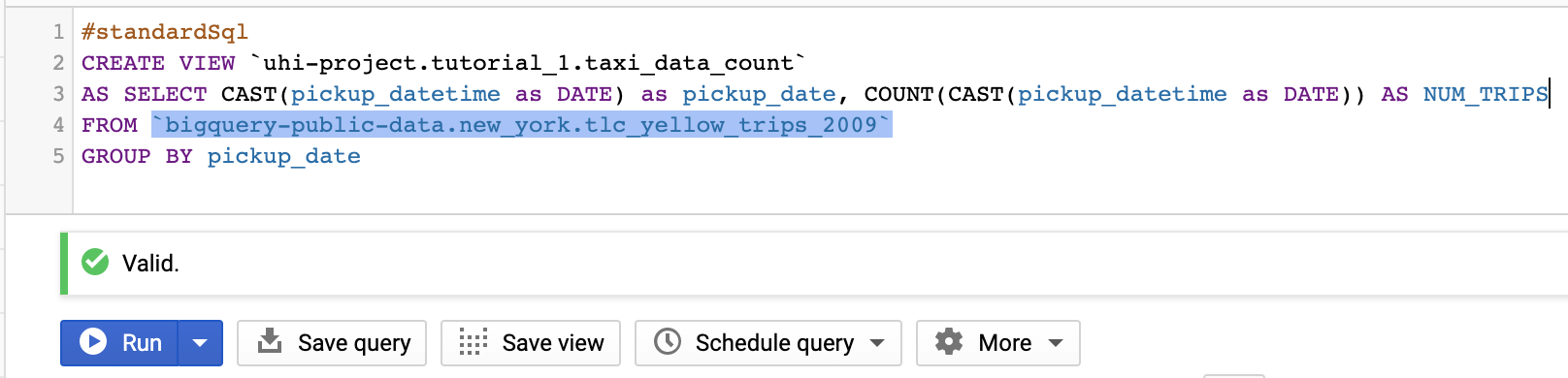
Next, click “Create Dataset”:



Now, under uhi-project, you will see:



We will now create a view:

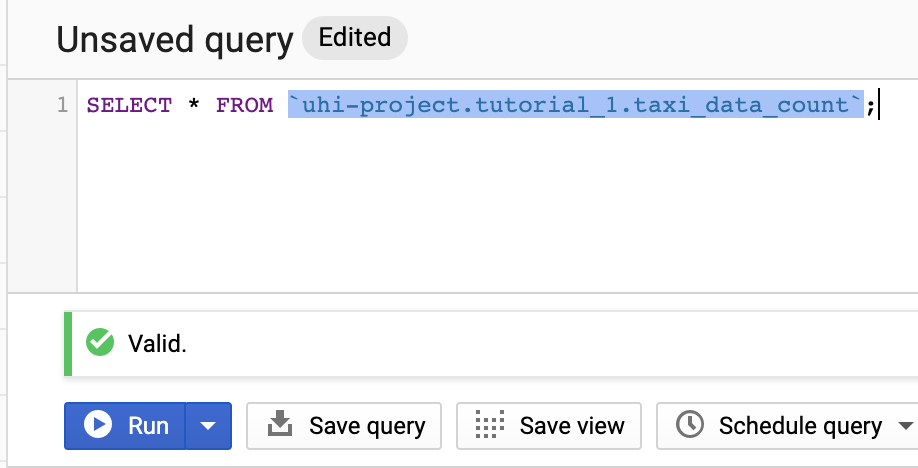


What this will do is create a view with the name “taxi\_data\_count” that will change the date in the “tlc\_yellow\_trips\_2009” to a DATE format rather than a timestamp. This will help us later. It will also count the number of trips for a particular day i.e. there will be lots of trips in one day, this will group them together for each date, then return the length of each of these groups.

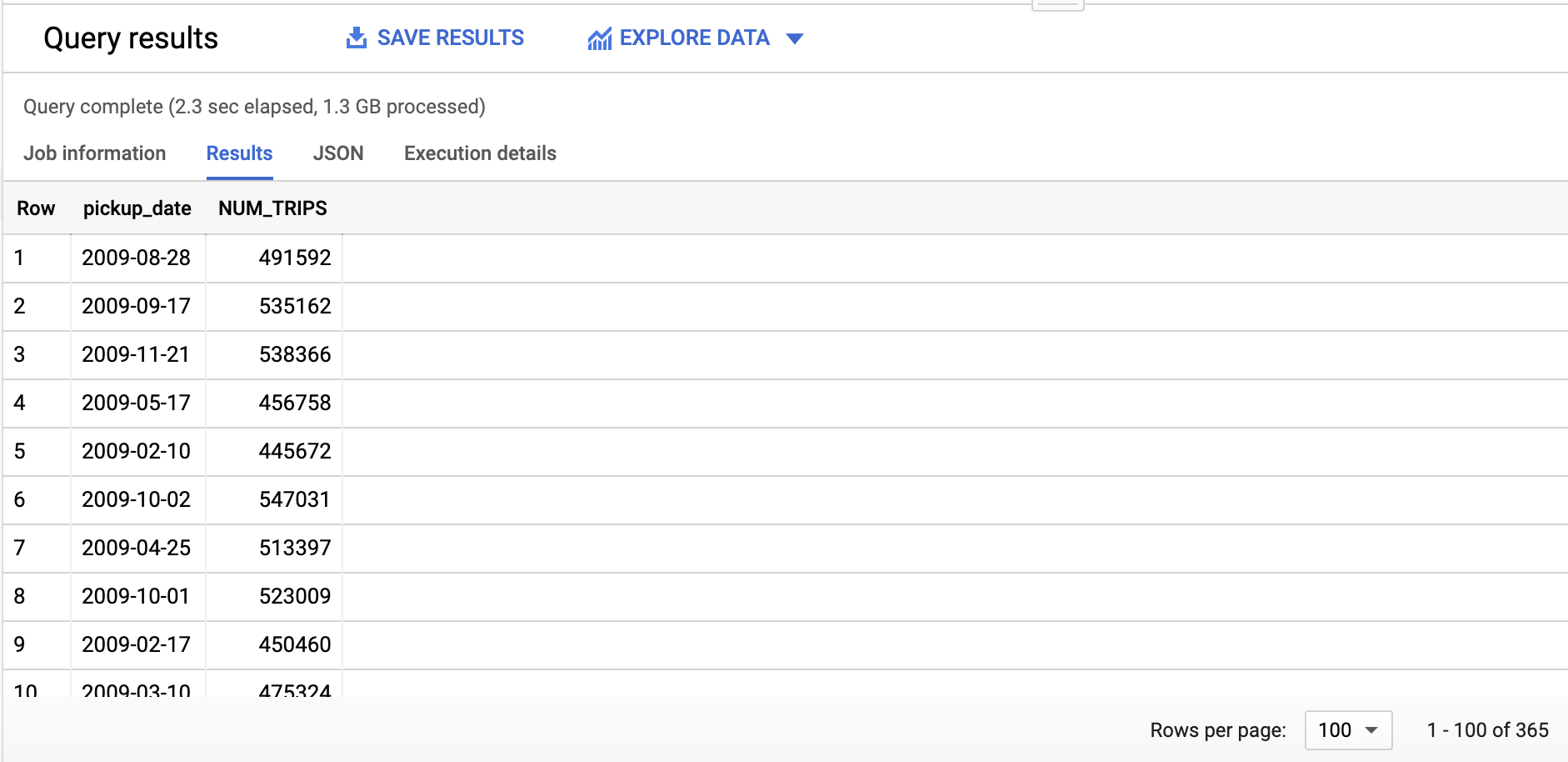
After running this query, you will see the following:



Now, let’s check out our data:

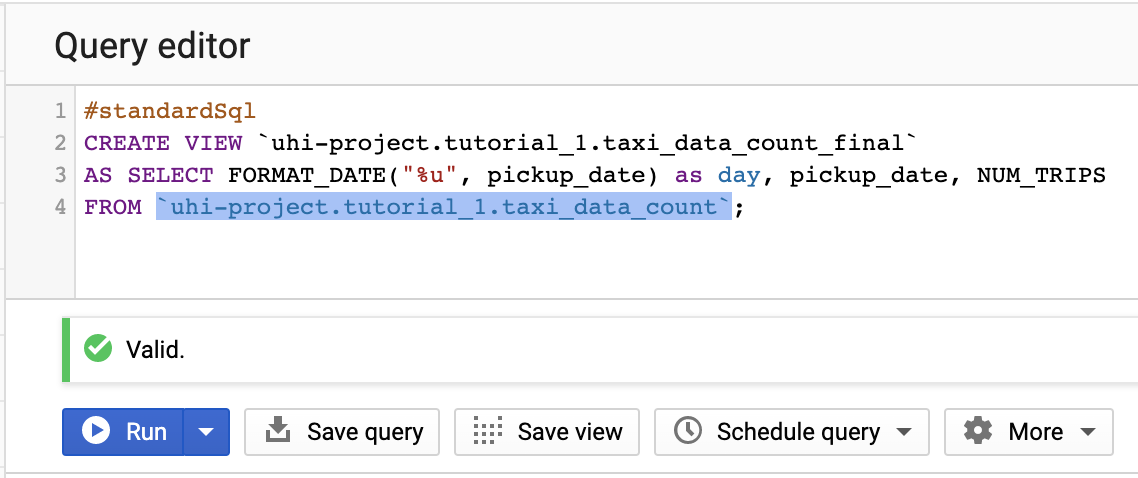


Run this query and you can see:

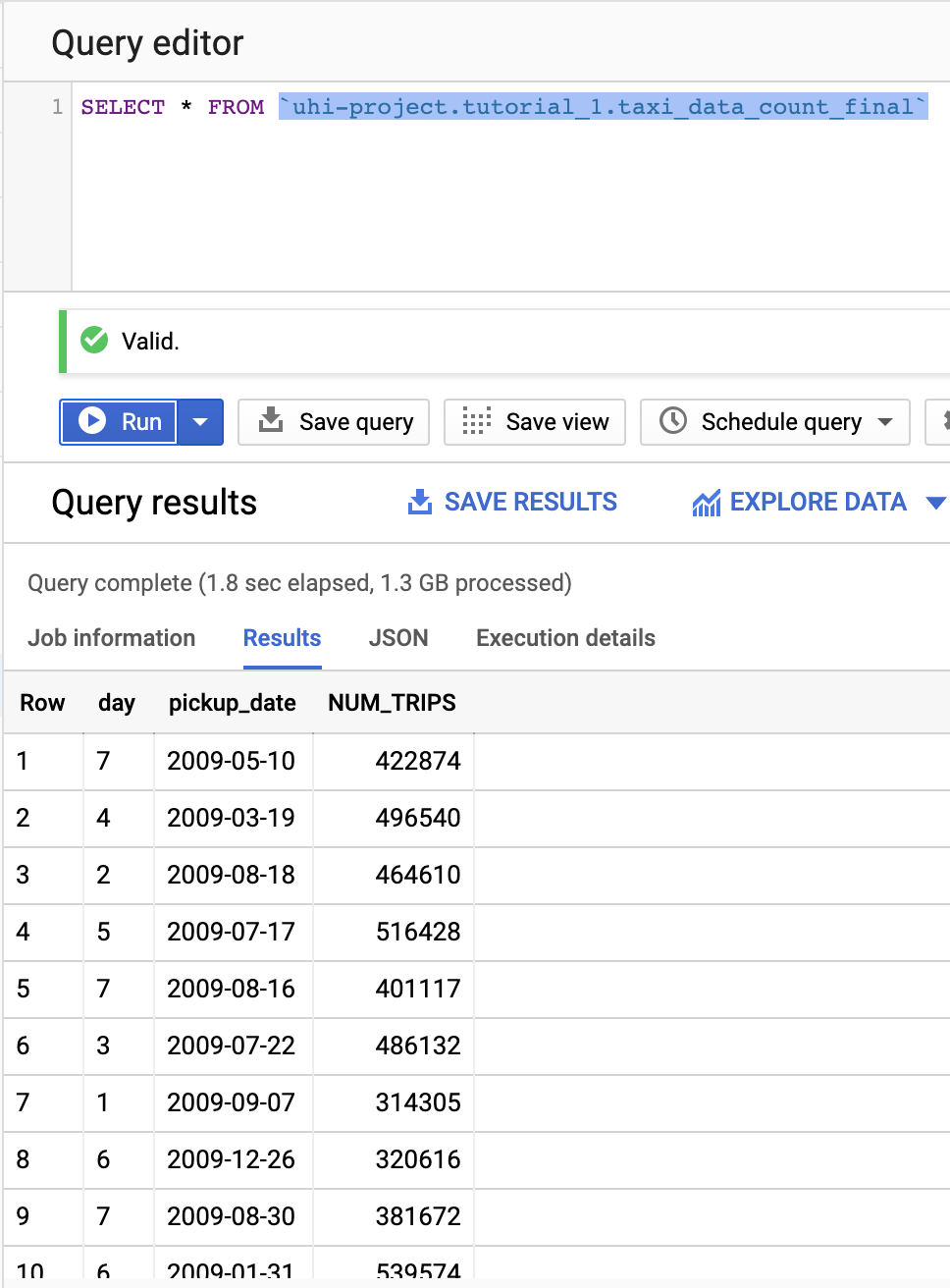


The dates aren’t ordered, but you can note there is 365 rows i.e. 1 for every day in the year and that NUM\_TRIPS shows the number of taxi journeys on that particular day.

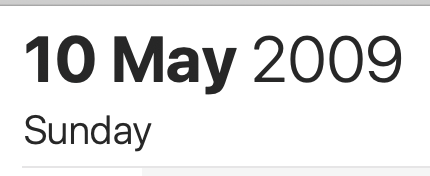
Next, we are going to add a new day of the week integer field. To do this we will use the view we just made, to make another view:



Now, let’s see the data:



As you can see, the 10th of May 2009 was a Sunday (7):



And the 7th of September is a Monday (1):

A close up of a logo

Description automatically generated

Next, we need to deal with our weather data. Firstly, create another view:

A screenshot of a cell phone

Description automatically generated

Let’s have a look at it:

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

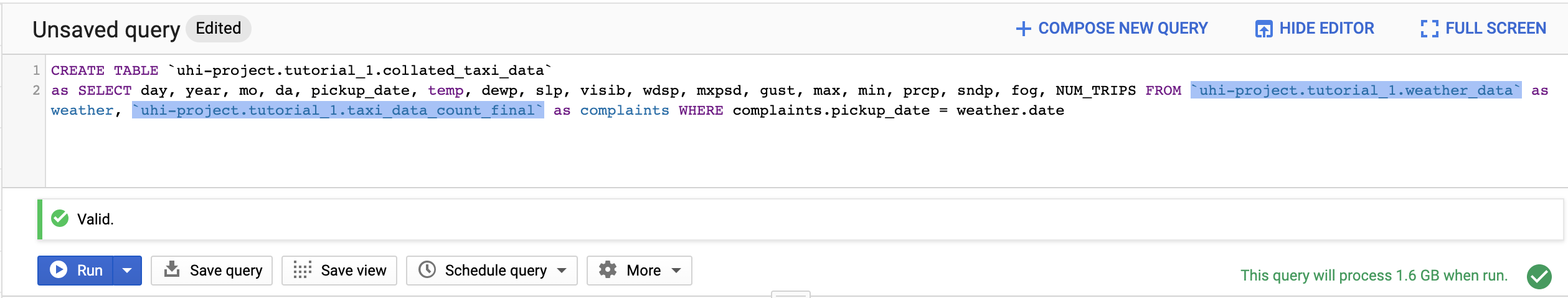
We get:

A picture containing computer

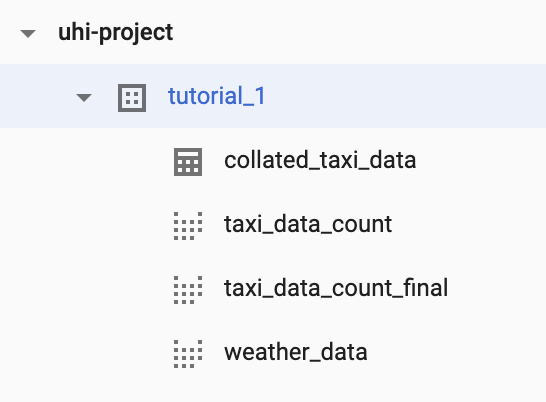
Description automatically generated

As you can see, I have created a date field that is of the type DATE. This will allow us to join with the taxi data.

Lastly, let’s create a table with our final data:



And check the data:



A screenshot of a cell phone

Description automatically generated

And we get:

A screenshot of a cell phone

Description automatically generated

1.3 Download the data as a CSV

With the returned data above, click on:

A close up of a logo

Description automatically generated

Then CSV local file:

A screenshot of a cell phone

Description automatically generated

Then Save.

A picture containing drawing

Description automatically generated

The file should be downloaded and you can check it in a text editor:

A picture containing computer

Description automatically generated

At this point, you should upload this file to your git repository that you are using for this course. If you haven’t done it already from tutorial 0:

Go to the URL: <https://github.com/>

On the left-hand column menu, click New:

A close up of a logo

Description automatically generated

Add a repository name:

A screenshot of a cell phone

Description automatically generated

As you can see, I have used my made-up student number, but you should use yours.

Add a description:

A screenshot of a cell phone

Description automatically generated

Notice that I have selected “Initialize this repository with a README”.

Click “Create repository”.

You should see something like this:

A screenshot of a social media post

Description automatically generated

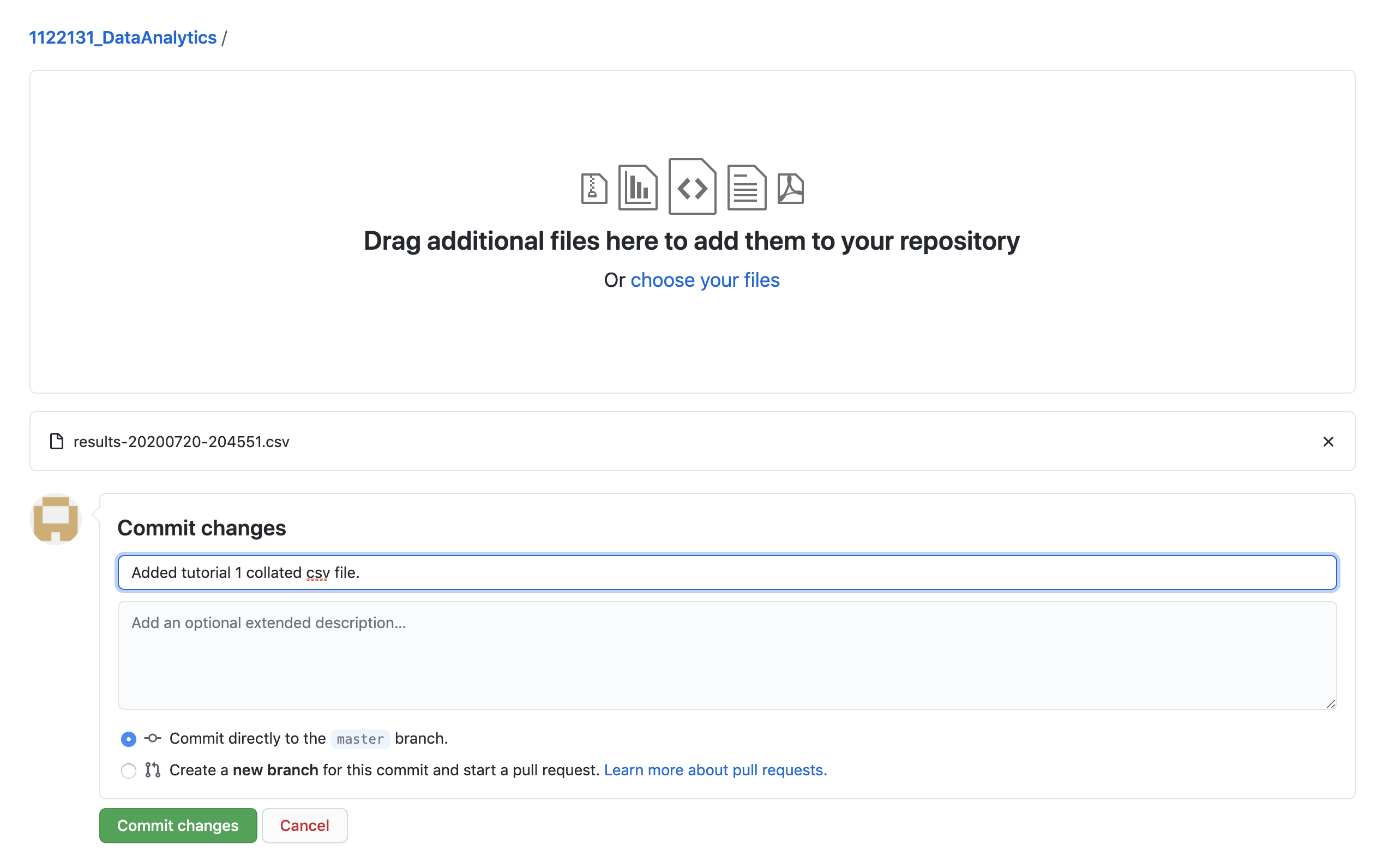
You can add the csv file in a number of different ways but we will just use the web interface. Firstly, save your finished csv file as <whateveryoulike>.csv to somewhere familiar i.e. your Desktop.

On github, on the repo page shown above, click “Add file”:

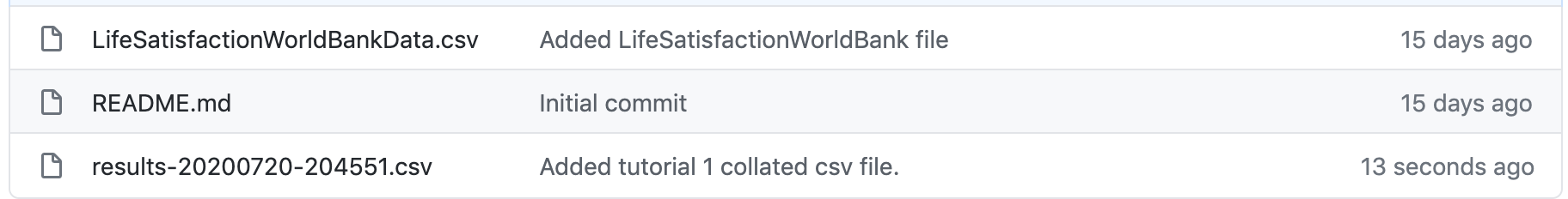
A screenshot of a cell phone

Description automatically generated

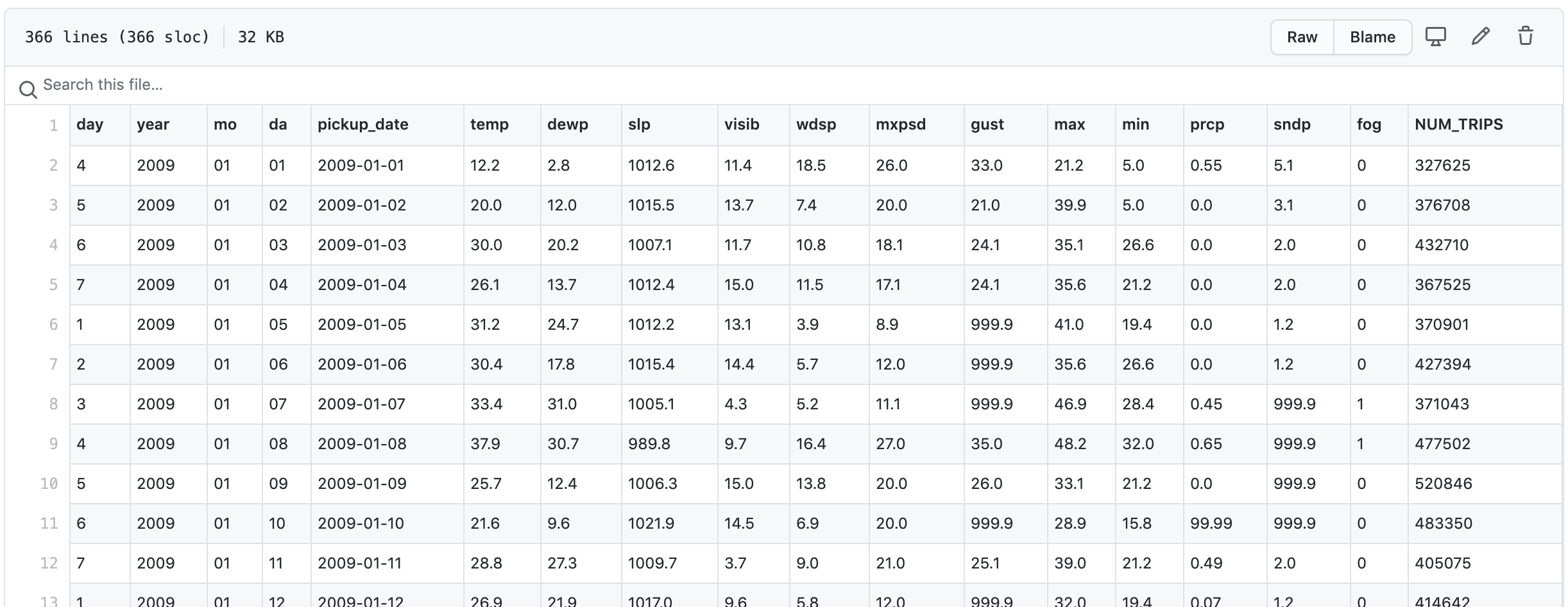
Select the file and write a commit message:



Click “Commit changes”:



You should now see your file. Click on it:



Click on Raw:

A screenshot of a computer

Description automatically generated

Now, take note of the url, copy it somewhere, we will need it.

(Here is mine just in case you are having problems: <https://raw.githubusercontent.com/1122131uhi/1122131_DataAnalytics/master/results-20200720-204551.csv>)

<https://raw.githubusercontent.com/20023167uhi/DataAnalytics/main/Tutorial_1-Weather_Taxis-results-20201105-114643.csv>

1.4 Known issues and things for the assignment:

In the assignment, you most likely want to download the full dataset that you are looking at (not the weather). This way you can do data science on it to see if there are mitigating factors or data that needs cleansed as shown in the main course document. In this case we jumped straight to collating data. The collated data will also require to be analysed and cleansed.

Also, the assignment data has a data range over a number of years. You will need to join the data for each year in the weather data, together.

Let’s explore this:

To get all of the data for 2009 you will need to use the query:

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2009` WHERE stn='725060'

As you can see, this returns 365 days. 2010 is not so simple. For example:

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2010` WHERE stn='725060'

We get 518 days. On further inspection we can see that there are three different wban numbers. In the case of 2010 we can see these results for wban 94728 (153 results) and wban 99999 (153 results) or 14704 (212)

As you can see, we can use wbans 14704 and either 94728 or 99999 i.e.

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2010` WHERE stn='725060' AND (wban='14704' OR wban='94728')

Returns 365 days.

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2011` WHERE stn='725060'

Returns 365 days

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2012` WHERE stn='725060'

Returns 366 days (leap year)

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2013` WHERE stn='725060'

Return 365

2014 requires a similar adjustment to 2010 =>

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2014` WHERE stn='725060' AND wban='14756'

Returns 365 days

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2015` WHERE stn='725060'

Returns 365 days

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2016` WHERE stn='725060'

Returns 366 days (leap year)

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2017` WHERE stn='725060'

Returns 365 days

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2018` WHERE stn='725060'

Returns 365 days

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2019` WHERE stn='725060'

Returns 365 days

SELECT \* FROM `bigquery-public-data.noaa\_gsod.gsod2020` WHERE stn='725060'

Returns 199 days (as year is not over yet) with no duplicates.

Now, I am going to show you how to merge three of these weather tables together into one table. This will be useful for the assignment and will require more than 3 unions.

A screenshot of a social media post

Description automatically generated

Running this we get:

A screenshot of a cell phone

Description automatically generated

Let’s look at the results:

A screenshot of a social media post

Description automatically generated

Run this and we get:

A screenshot of a social media post

Description automatically generated

As you can see, we get 1095 results for 3 years (no leap years) between 2009 – 2011.

This method will be useful for your assignment.