BigQuery Tutorial

MSBA7024/MACC7020

HKU

What will you learn in this tutorial?

- What is BigQuery
- Why use BigQuery
- BigQuery data organization
- Basic use of BigQuery
  - Create a free account
  - Create a project/dataset/table
  - Run SQL queries
  - Save results/query/view
  - Access BigQuery public datasets
- BigQuery exercises

What is Google BigQuery?

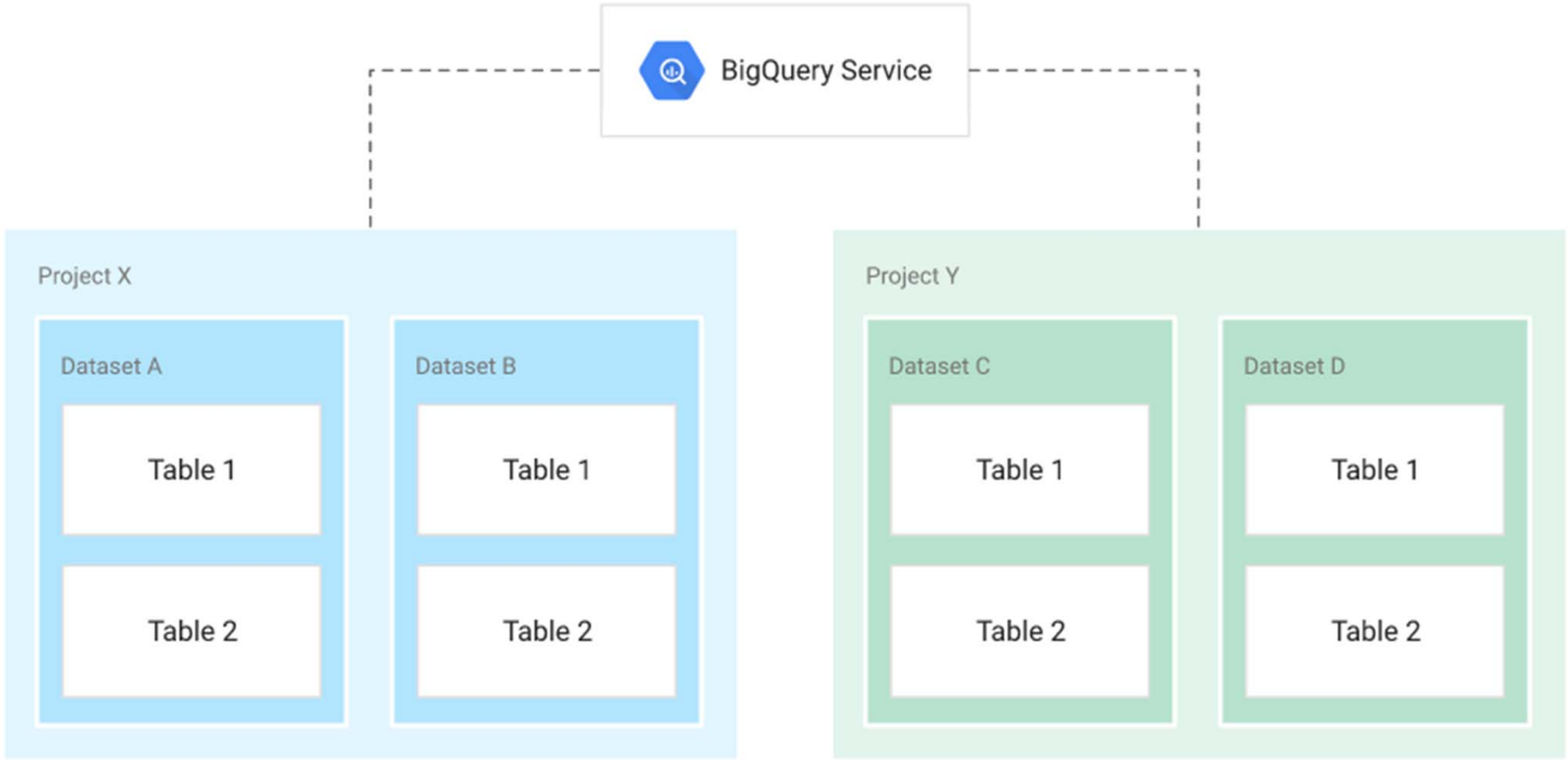
- BigQuery is Google Cloud's fully managed and completely serverless enterprise data warehouse.

- It allows for super-fast queries at petabyte scale using the processing power of Google's infrastructure.

Why use BigQuery?

- Real-time analysis of massive datasets at high speed
- Cloud service
- Web-based interface
- Standard SQL
- Public datasets
- Built-in machine learning
- High security
- Flexible pricing model

BigQuery Data Organization

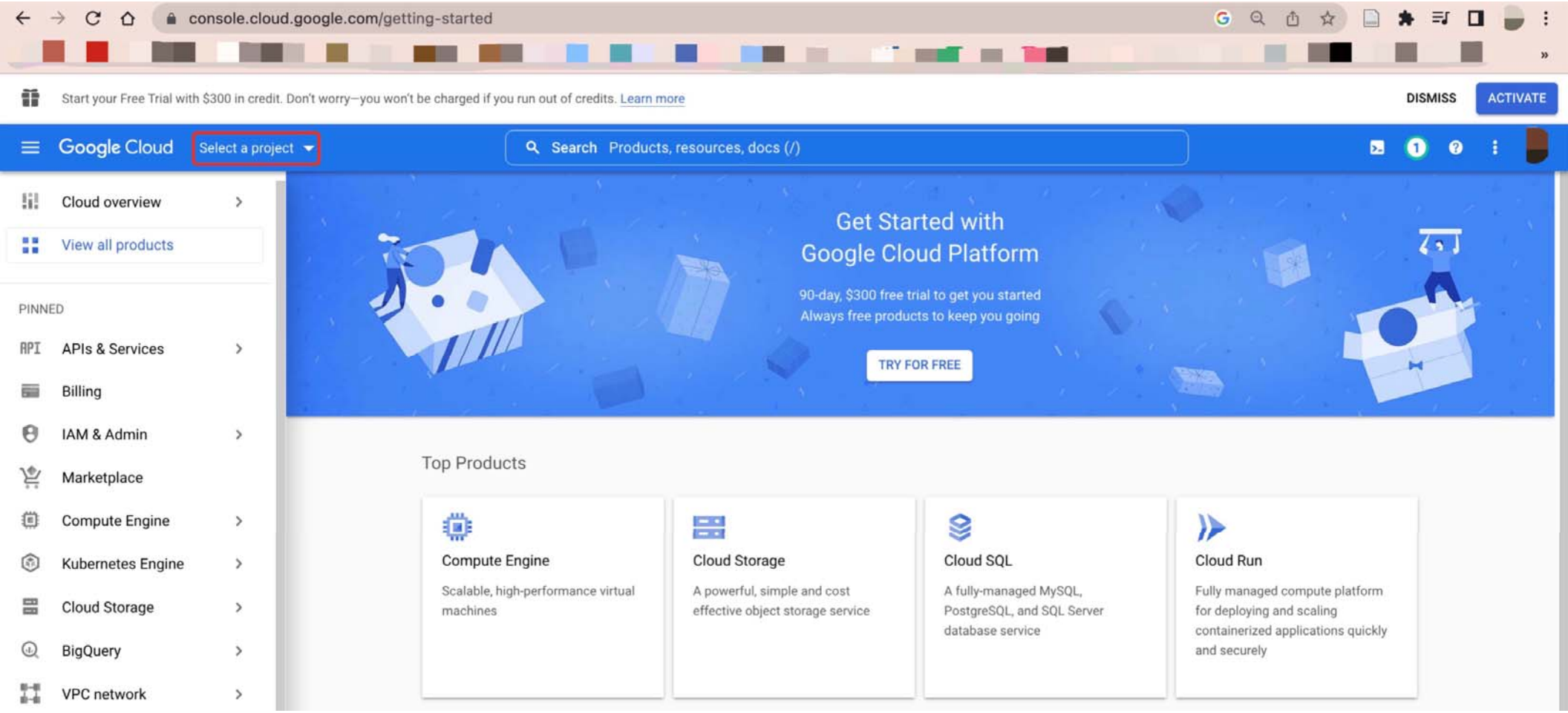


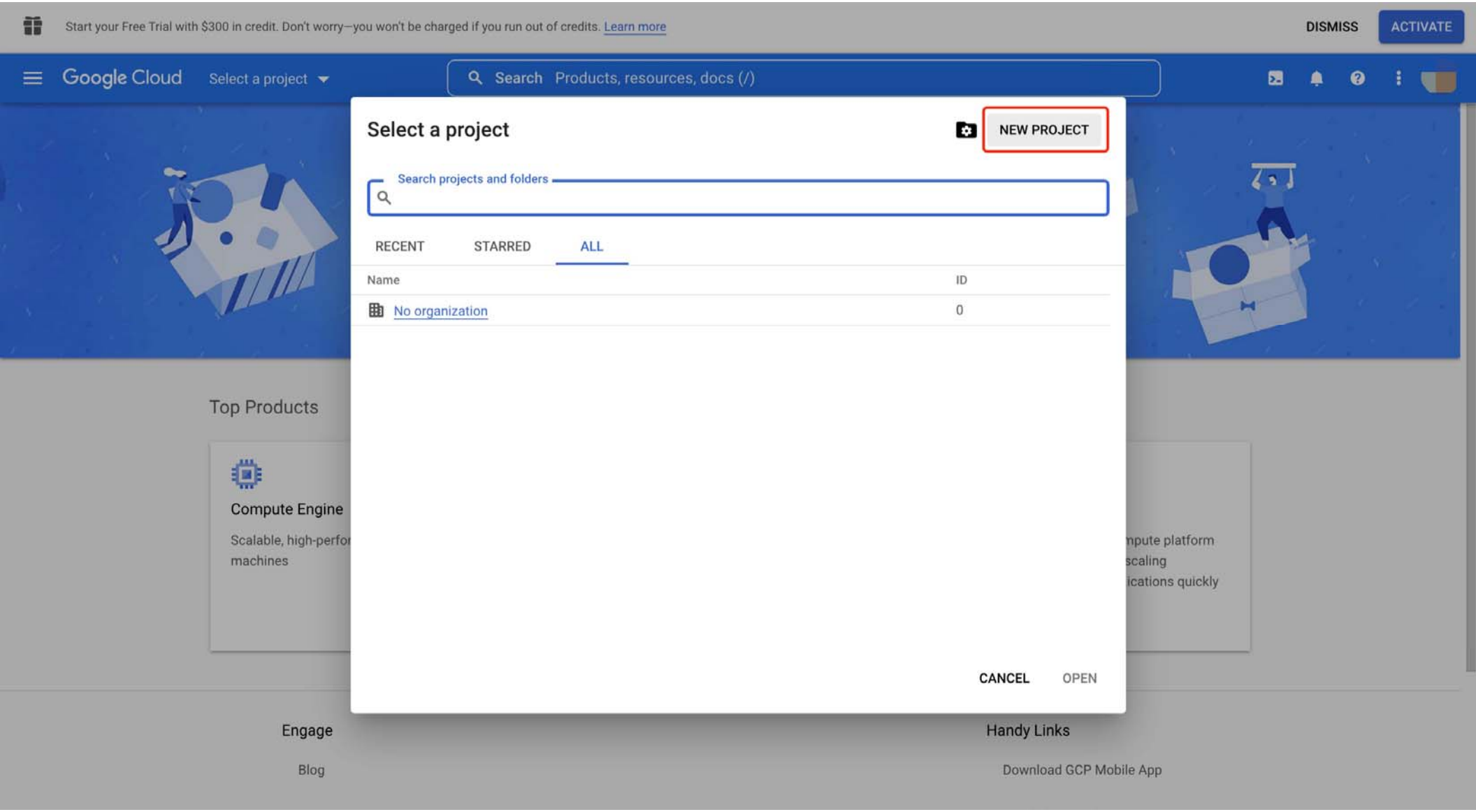
Source: https://medium.com/google-cloud/bigquery-explained-storage-overview-70cac32251fa

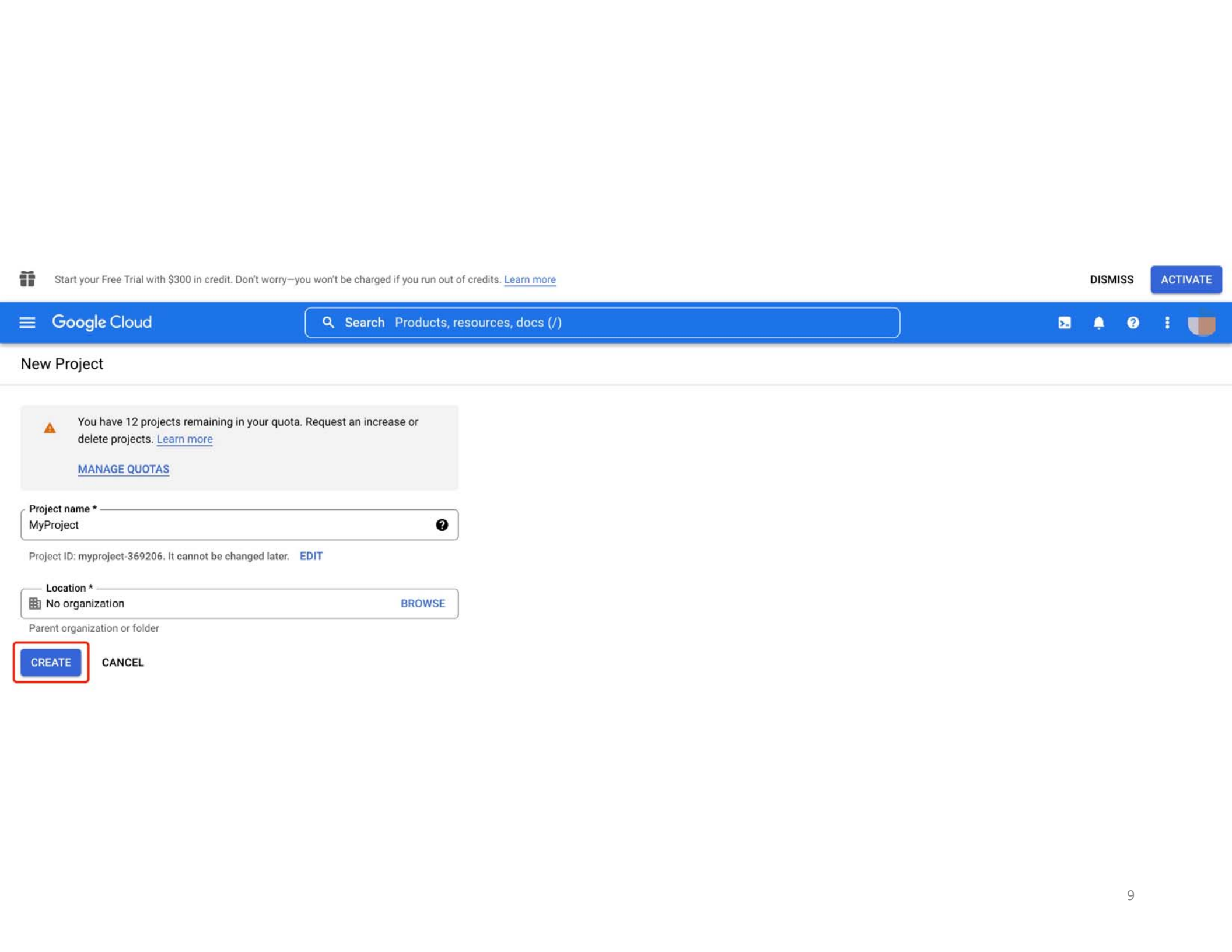
BigQuery Setup Guide

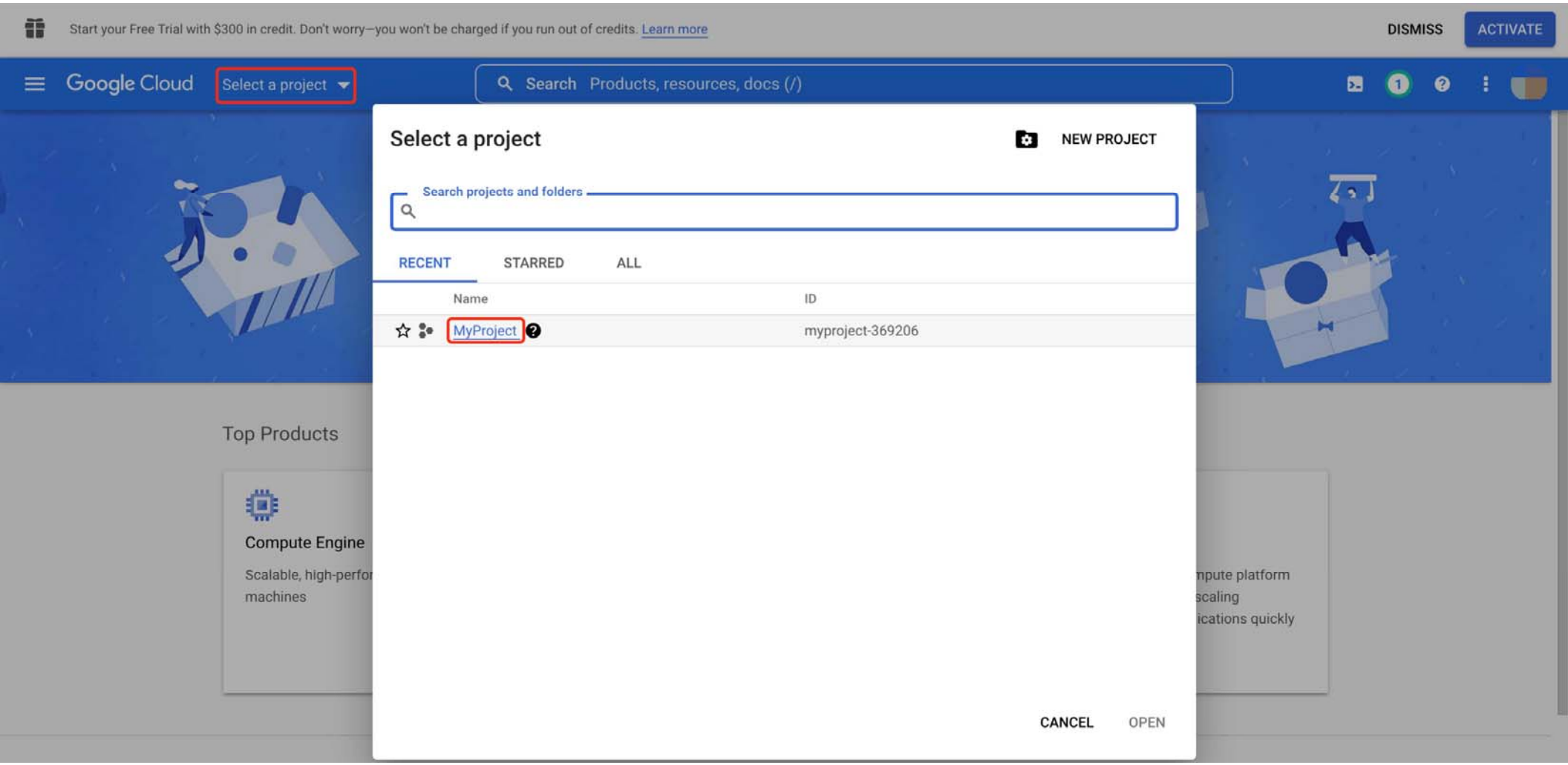
- https://console.cloud.google.com/

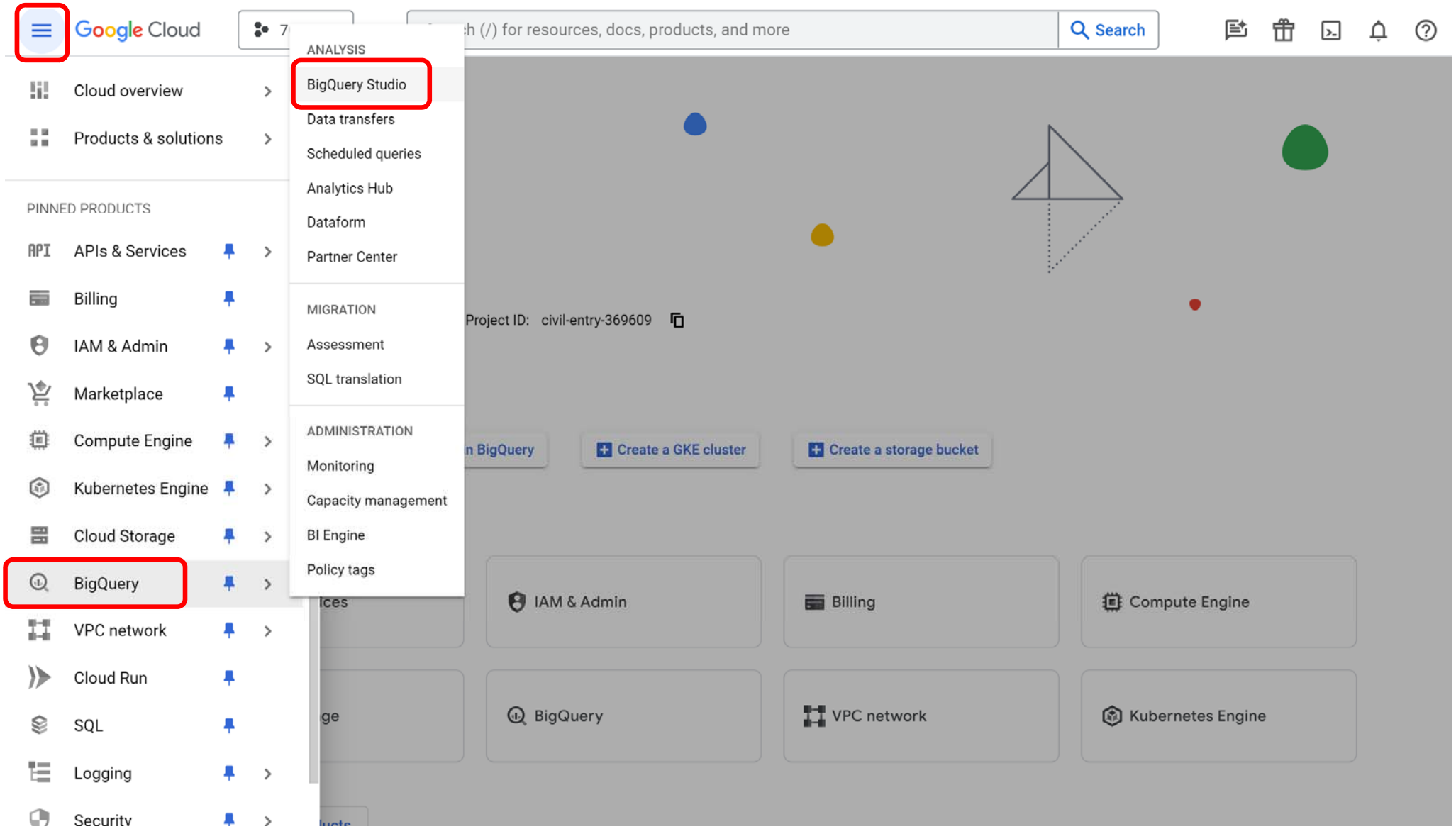
- If it is your first visit, you will need to select your country and agree to the Terms of Service.


console.cloud.google.com/getting-started
Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more
DISMISS
ACTIVATE
Google Cloud
Select a project
Search Products, resources, docs (/)
Cloud overview
View all products
PINNED
APIs & Services
Billing
IAM & Admin
Marketplace
Compute Engine
Kubernetes Engine
Cloud Storage
BigQuery
VPC network
Get Started with
Google Cloud Platform
90-day, $300 free trial to get you started
Always free products to keep you going
TRY FOR FREE
Top Products
Compute Engine
Scalable, high-performance virtual machines
Cloud Storage
A powerful, simple and cost effective object storage service
Cloud SQL
A fully-managed MySQL, PostgreSQL, and SQL Server database service
Cloud Run
Fully managed compute platform for deploying and scaling containerized applications quickly and securely


Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more
DISMISS
ACTIVATE
Google Cloud
Select a project
Search Products, resources, docs (/)
Select a project
NEW PROJECT
Search projects and folders
RECENT
STARRED
ALL
Name
ID
No organization
0
CANCEL
OPEN
Top Products
Compute Engine
Scalable, high-perfor
machines
mpute platform
scaling
ications quickly
Engage
Blog
Handy Links
Download GCP Mobile App


Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more
DISMISS
ACTIVATE
Google Cloud
Search Products, resources, docs (/)
New Project
You have 12 projects remaining in your quota. Request an increase or delete projects. Learn more
MANAGE QUOTAS
Project name *
MyProject
Project ID: myproject-369206. It cannot be changed later. EDIT
Location *
No organization
BROWSE
Parent organization or folder
CREATE
CANCEL

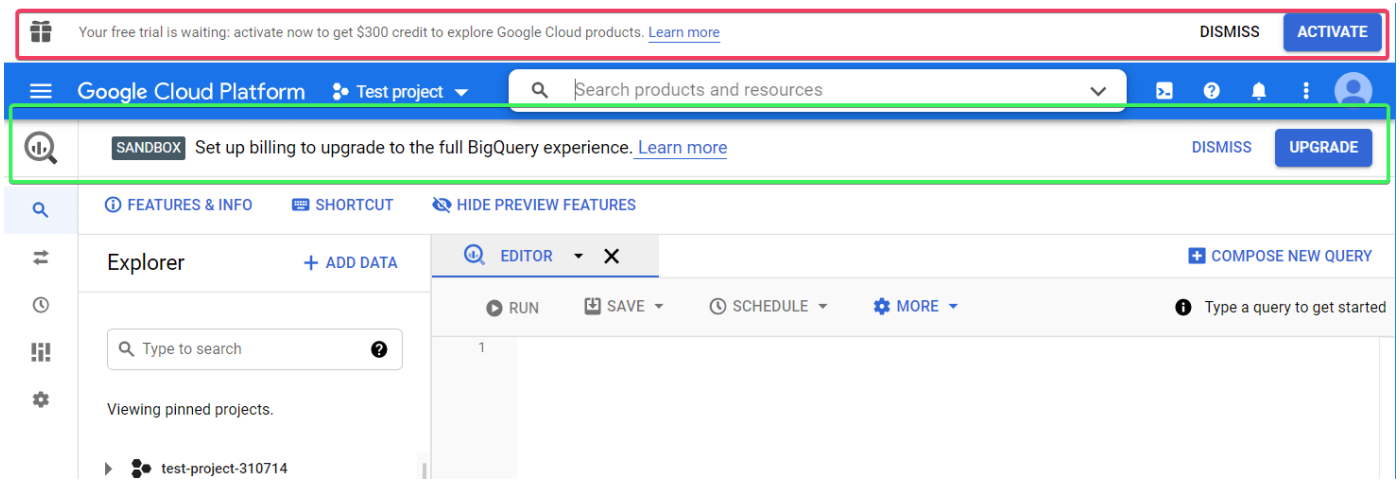

Start your Free Trial with $300 in credit. Don't worry—you won't be charged if you run out of credits. Learn more
DISMISS
ACTIVATE
Google Cloud
Select a project
Search Products, resources, docs (/)
Select a project
NEW PROJECT
Search projects and folders
RECENT
STARRED
ALL
Name
ID
MyProject
myproject-369206
CANCEL
OPEN
Top Products
Compute Engine
Scalable, high-perfor
machines
mpute platform
scaling
ications quickly



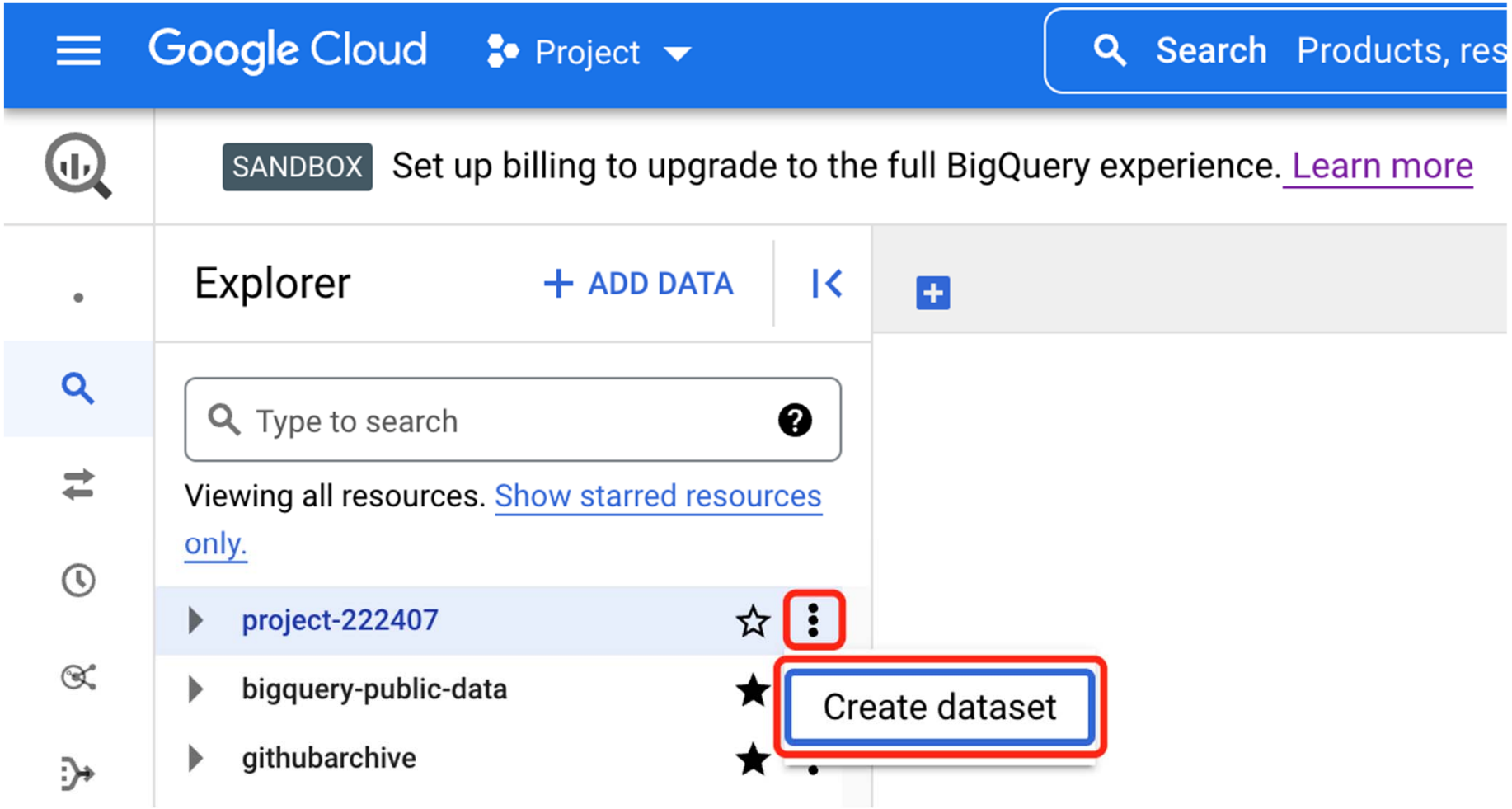
Or go here directly: https://console.cloud.google.com/bigquery

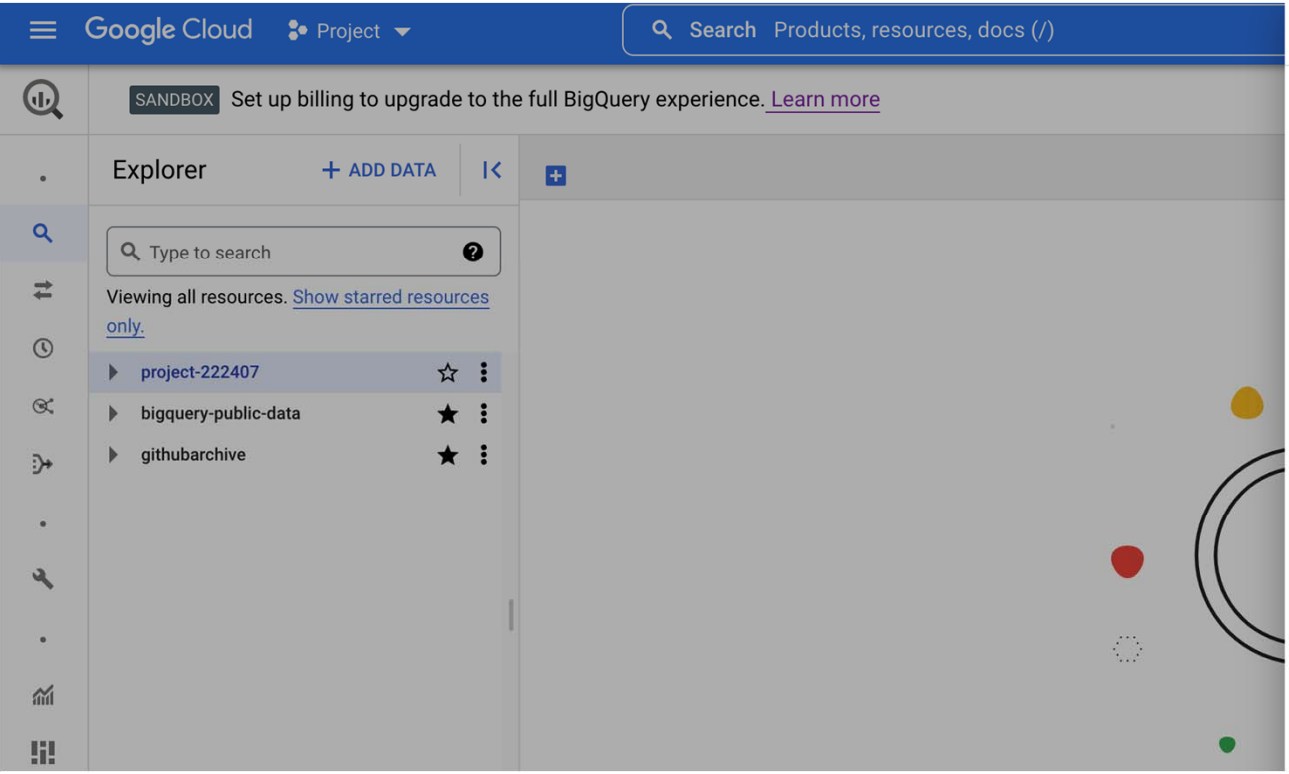
What is the BigQuery Sandbox?

- Sandbox is a credit-card free path to enable users to experiment with BigQuery at no cost.
  - Same compute power as paying users.
- Limitations
  - 10 GB of active storage and 1 TB of processed query data per month.
  - Your tables will expire in 60 days.



Create a Dataset





Create dataset

Project ID
project-222407
CHANGE

Dataset ID *
d1

Letters, numbers, and underscores allowed

Data location

Default table expiration

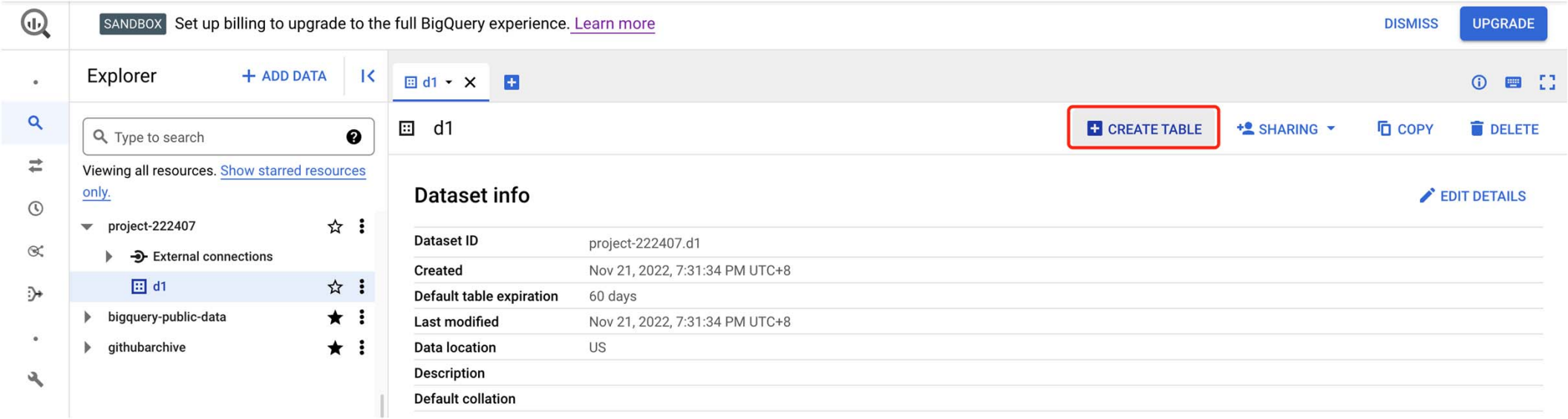
Enable table expiration

Default maximum table age
Days

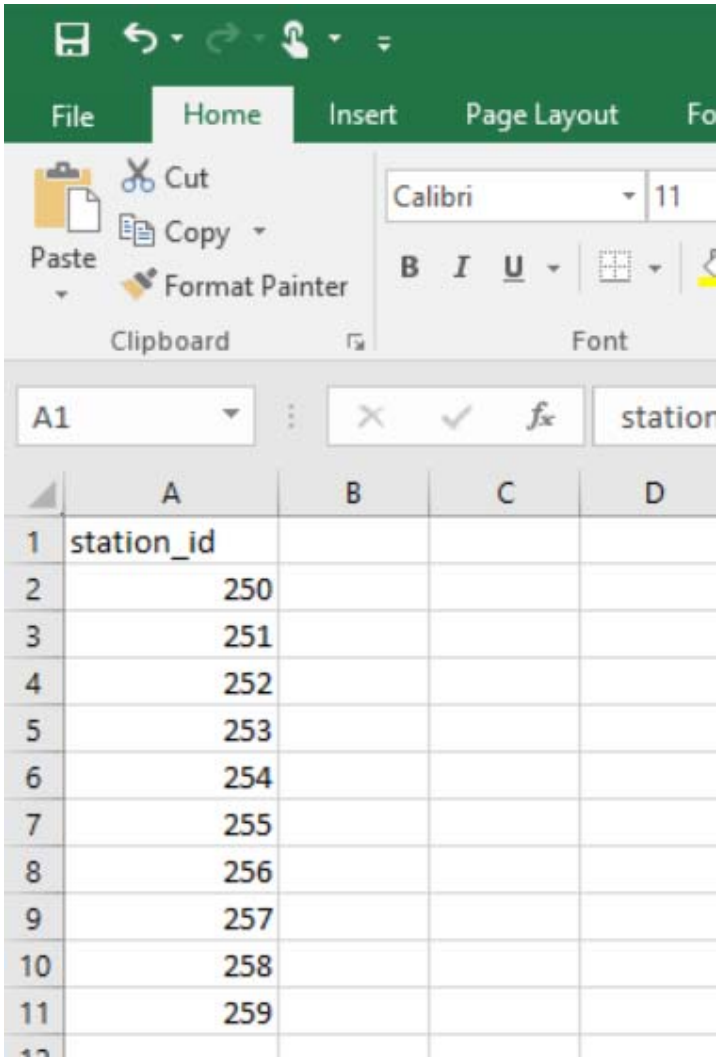
Advanced options

CREATE DATASET
CANCEL

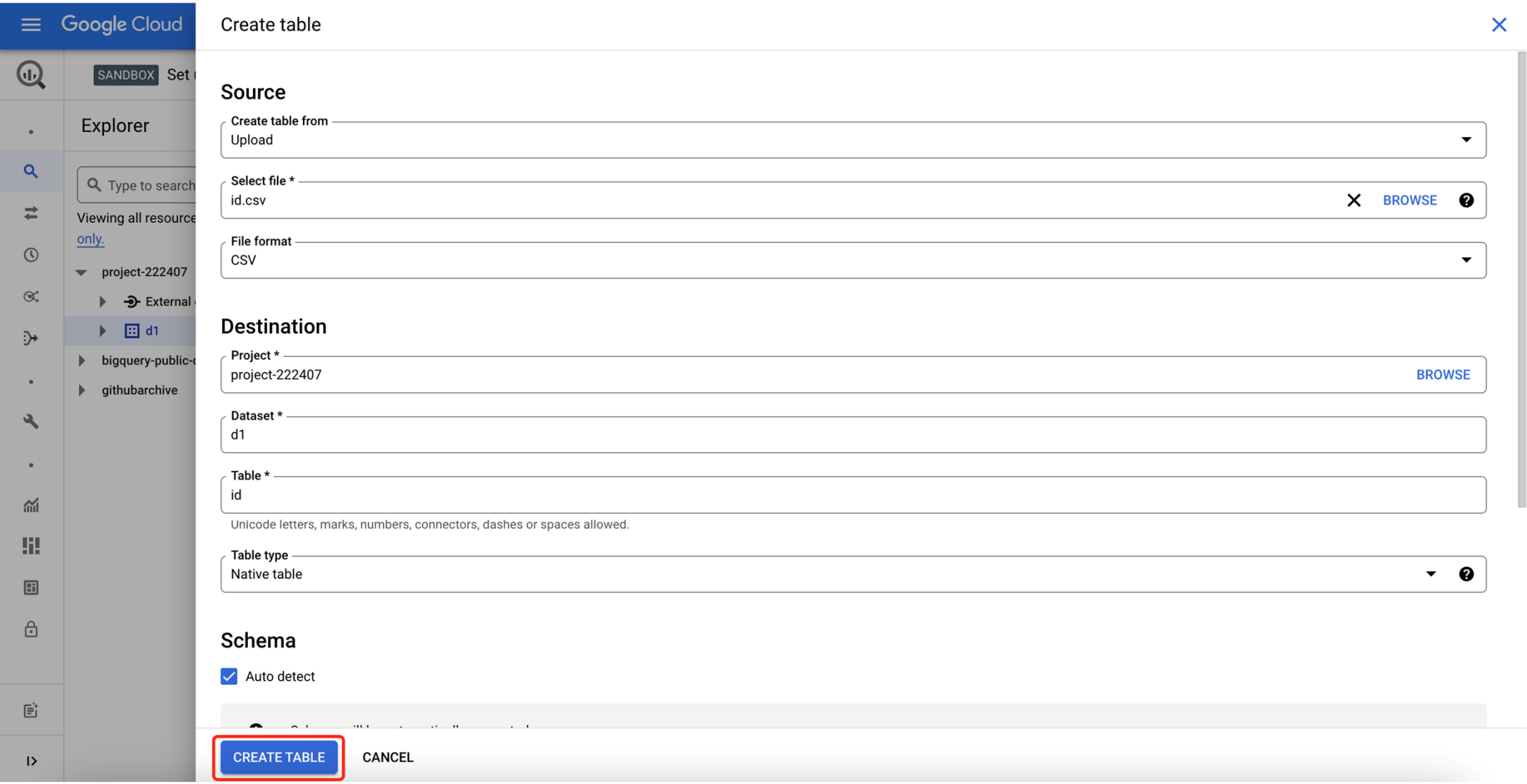
Create a Table



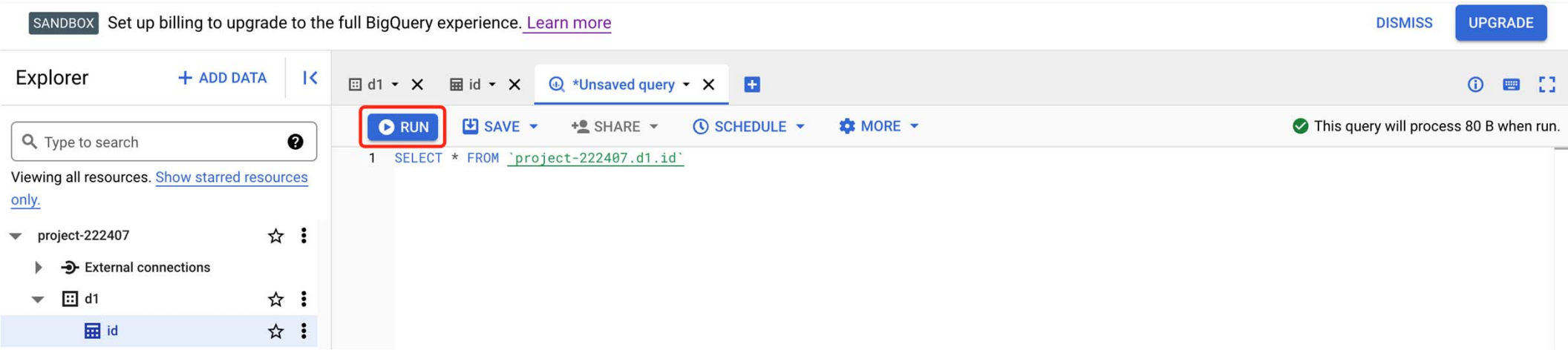
First, create a simple file in Excel



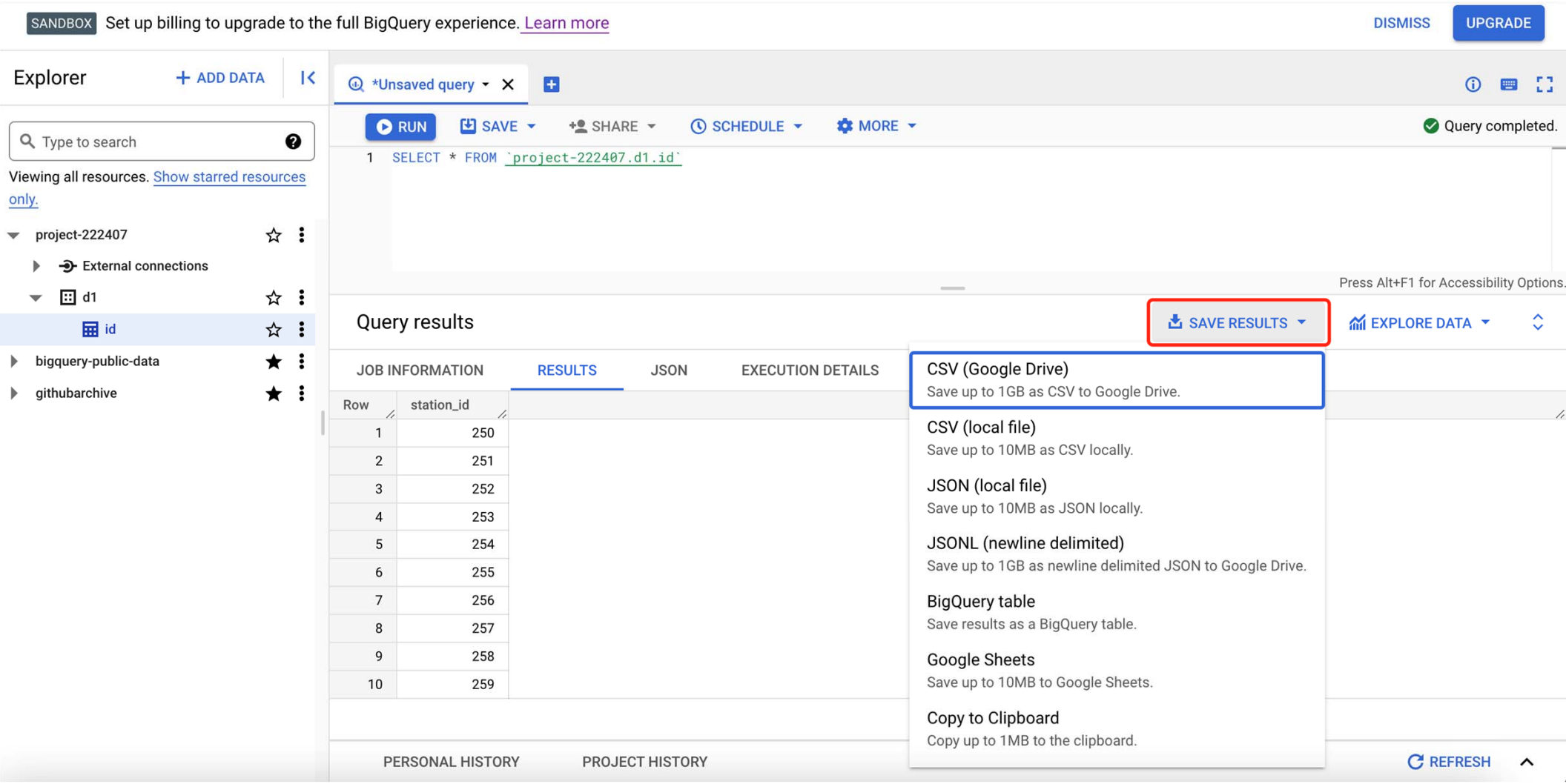
Then upload the file



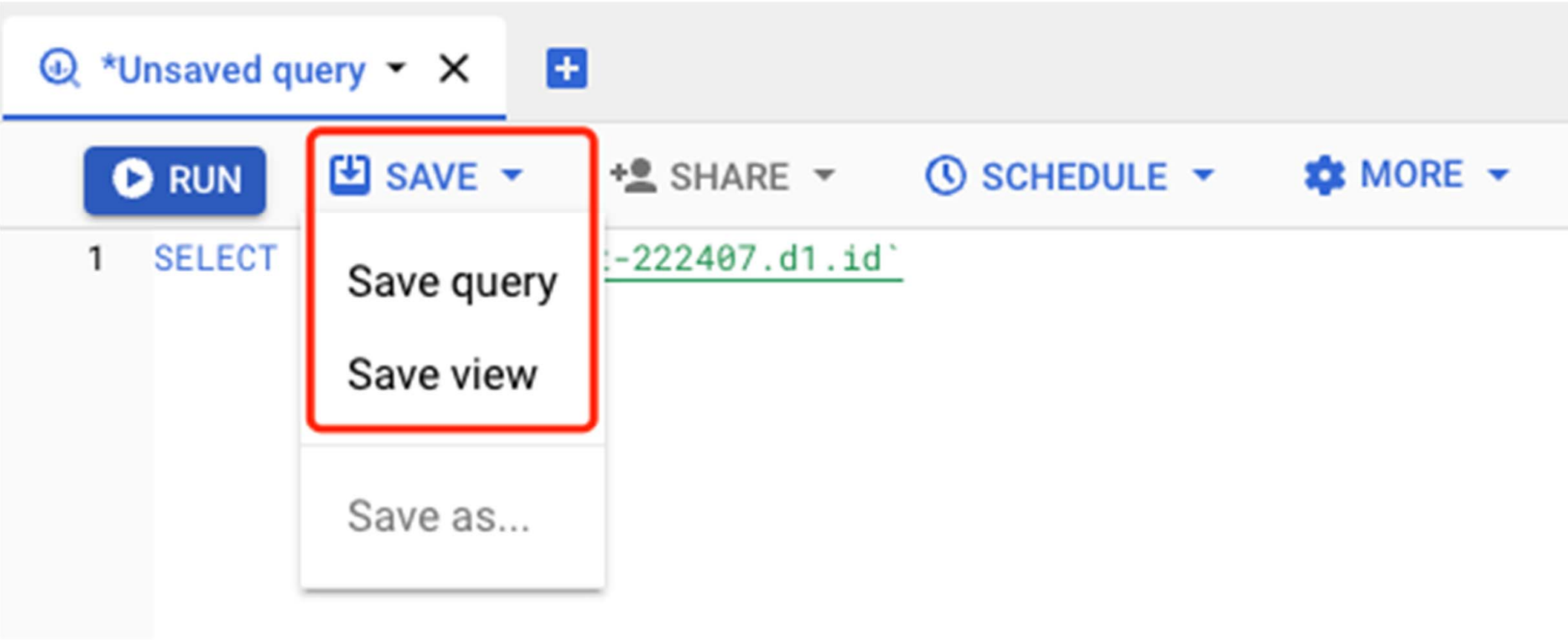
Query Tables



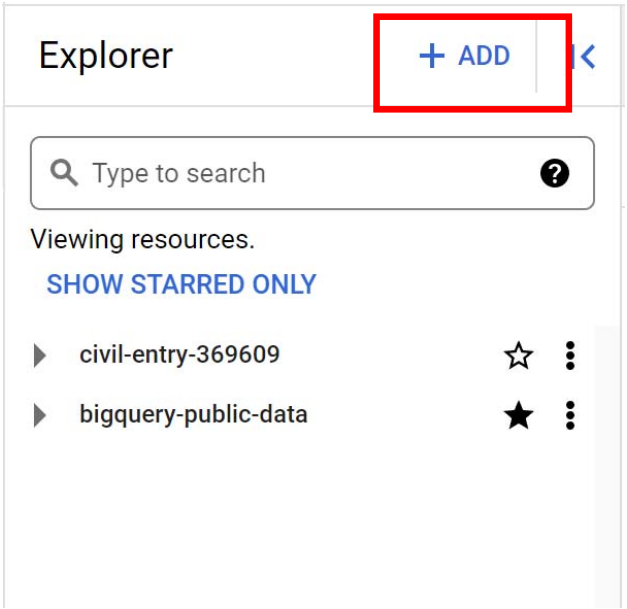
Save Results

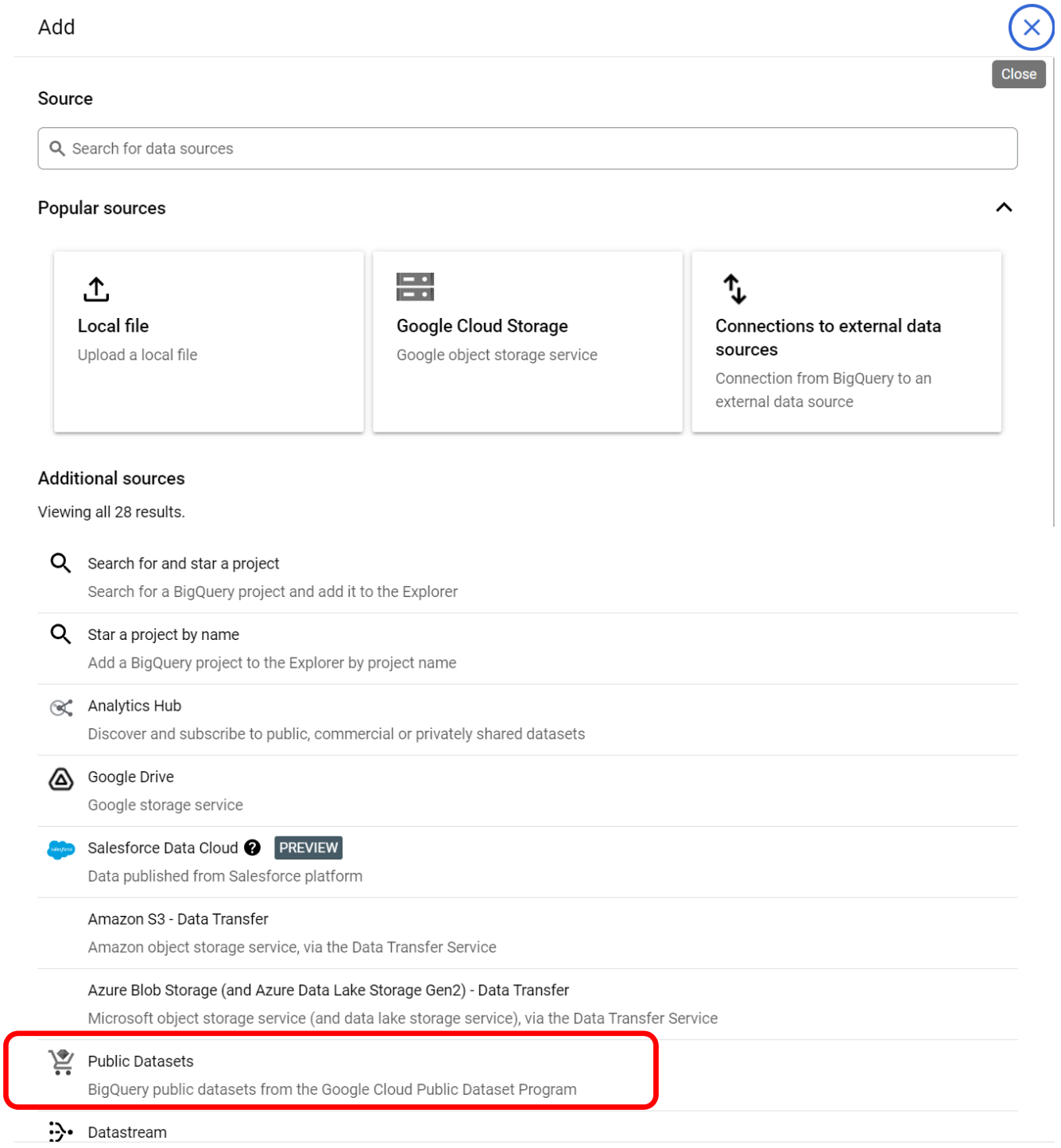


Save Query/View

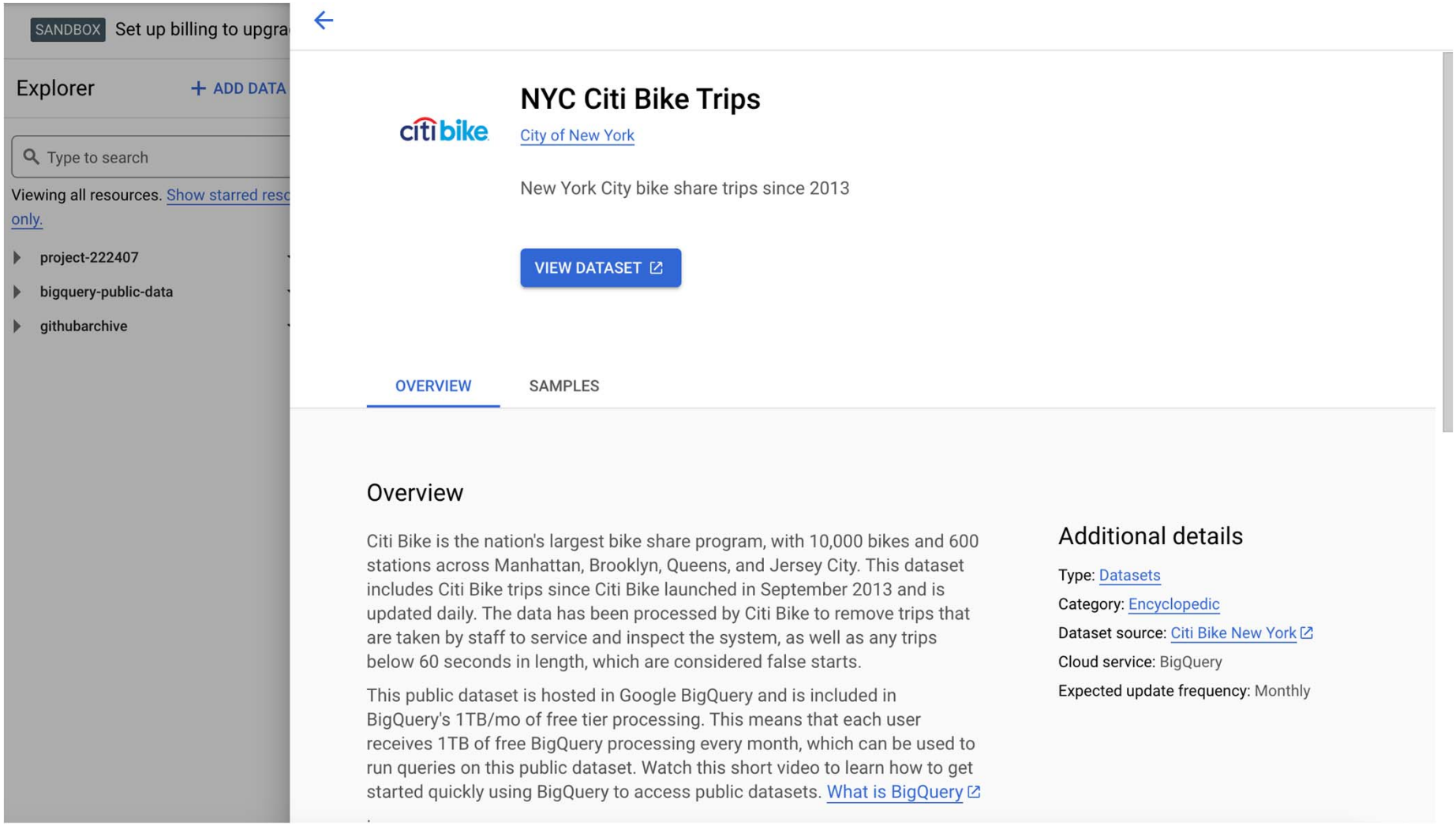


BigQuery Public Data

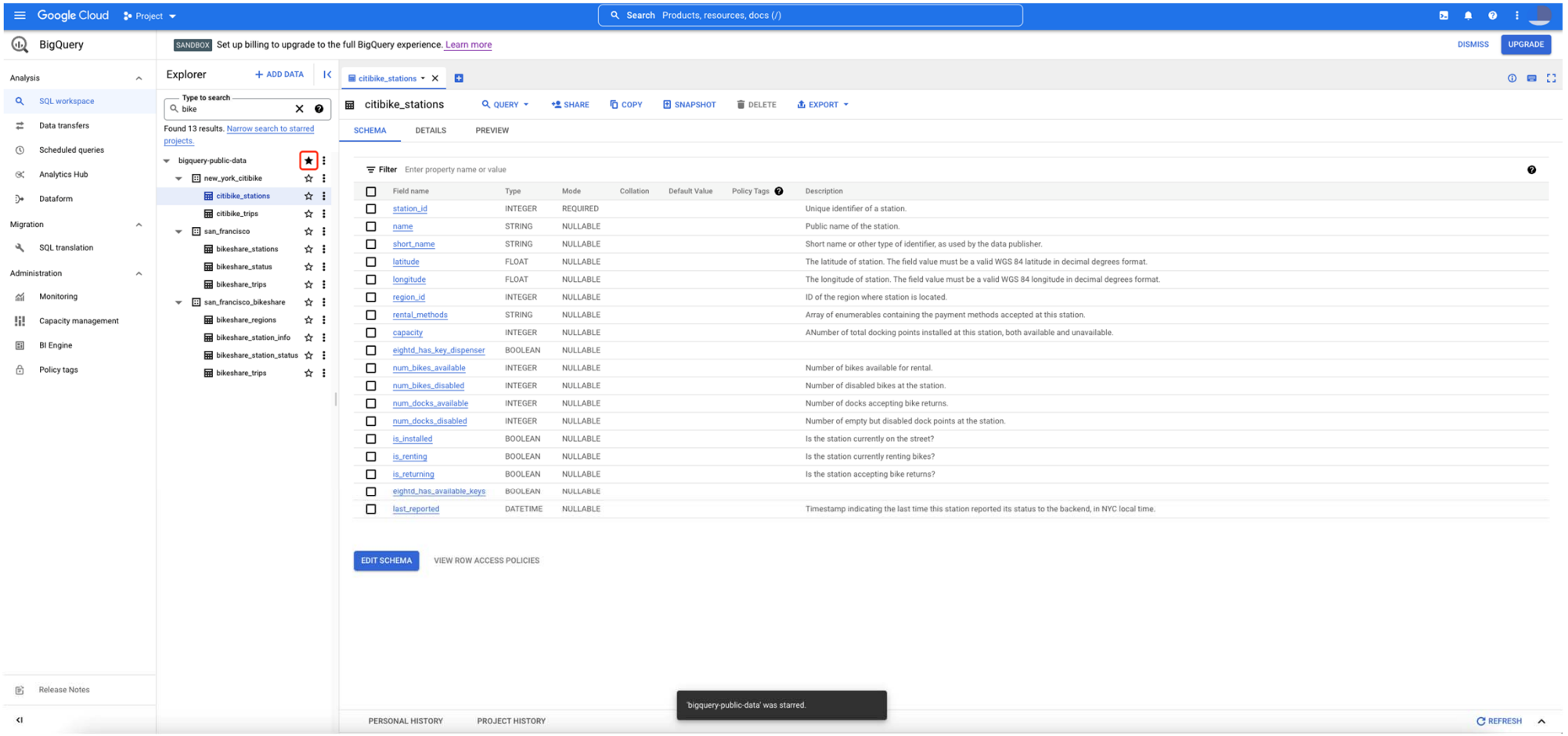




Search for the NYC Citi Bike Trips Data



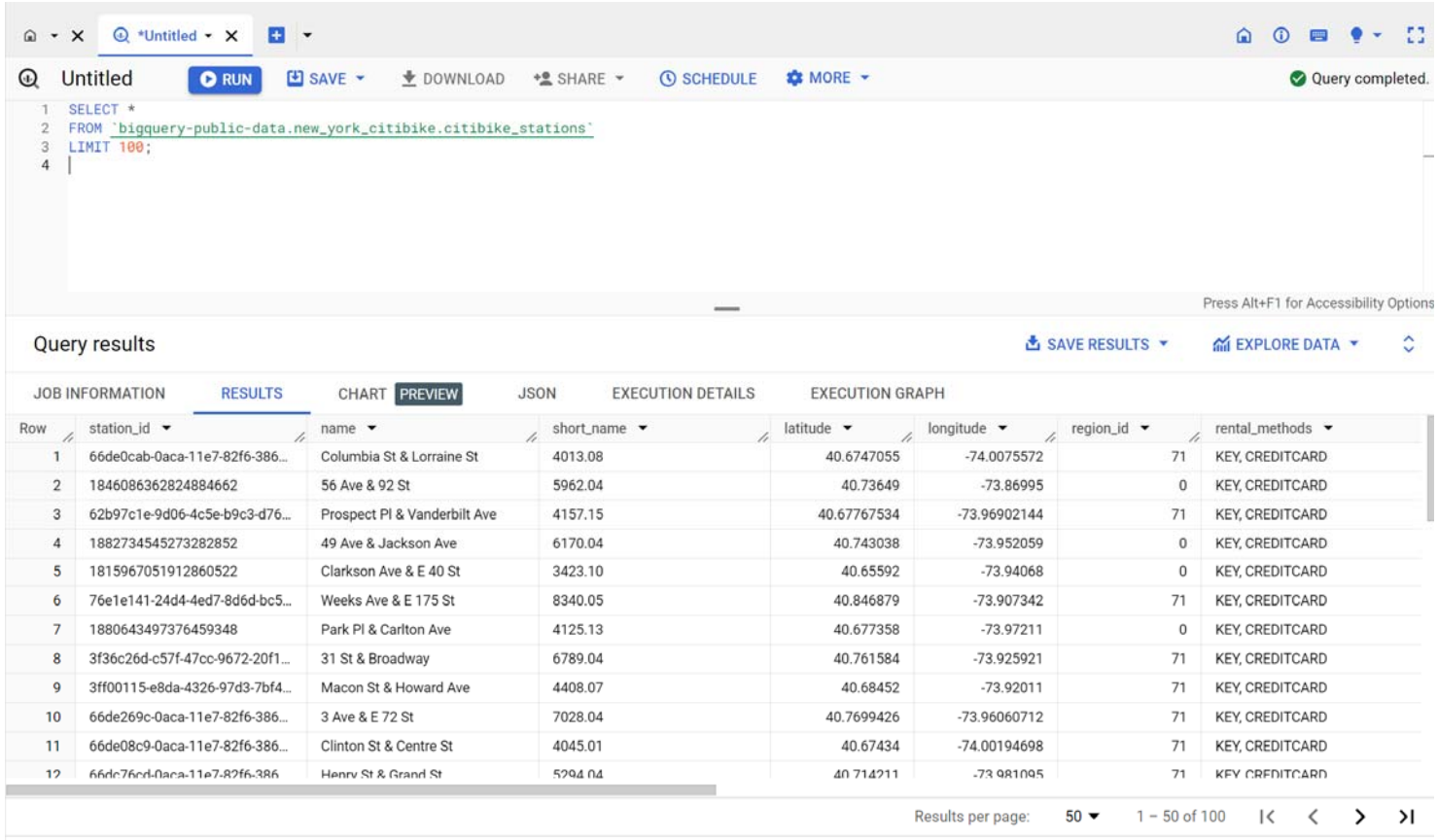
Under Explorer, look for bigquery-public-data and look for new_york_citibike



Simple query to look at the data

- The database is: "`bigquery-public-data.new_york_citibike`"

```
SELECT *
FROM `bigquery-public-data.new_york_citibike.citibike_stations`
LIMIT 100;
```



Row	station_id	name	short_name	latitude	longitude	region_id	rental_methods
1	66de0cab-0aca-11e7-82f6-386...	Columbia St & Lorraine St	4013.08	40.6747055	-74.0075572	71	KEY, CREDITCARD
2	184608362824884662	56 Ave & 92 St	5962.04	40.73649	-73.86995	0	KEY, CREDITCARD
3	62b97c1e-9d06-4c5e-b9c3-d76...	Prospect Pl & Vanderbilt Ave	4157.15	40.67767534	-73.96902144	71	KEY, CREDITCARD
4	1882734545273282852	49 Ave & Jackson Ave	6170.04	40.743038	-73.952059	0	KEY, CREDITCARD
5	1815967051912860522	Clarkson Ave & E 40 St	3423.10	40.65592	-73.94068	0	KEY, CREDITCARD
6	76e1e141-24d4-4ed7-8d6d-bc5...	Weeks Ave & E 175 St	8340.05	40.846879	-73.907342	71	KEY, CREDITCARD
7	1880643497376459348	Park Pl & Carlton Ave	4125.13	40.677358	-73.97211	0	KEY, CREDITCARD
8	3f36c26d-c57f-47cc-9672-20f1...	31 St & Broadway	6789.04	40.761584	-73.925921	71	KEY, CREDITCARD
9	3ff00115-e8da-4326-97d3-7bf4...	Macon St & Howard Ave	4408.07	40.68452	-73.92011	71	KEY, CREDITCARD
10	66de269c-0aca-11e7-82f6-386...	3 Ave & E 72 St	7028.04	40.7699426	-73.96060712	71	KEY, CREDITCARD
11	66de08c9-0aca-11e7-82f6-386...	Clinton St & Centre St	4045.01	40.67434	-74.00194698	71	KEY, CREDITCARD
12	66dc76cd-0aca-11e7-82f6-386...	Henry St & Grand St	5294.04	40.714211	-73.981095	71	KEY, CREDITCARD

Exercises

Q1. What is the total trip duration (in hours) of all male users? Use data from the `bigquery-public-data.new_york_citibike.citibike_trips` table.

Row	f0_
1	7962530.36...

Q2. What is the longest trip duration (in hours) from the station with ID 250?

Row	f0_
1	602.221944...

Q3. What are the 5 most frequently used bikes? Show the bike ID and how many times the bike has been used.

Row	bikeid	trip_count
1	18104	7222
2	15731	7146
3	19455	7076
4	17526	7030
5	16158	7025

Q4. What are the 5 most popular routes? Show the start station ID, end station ID, and how many times the route has been selected. (Note: the start and end stations should be different.)

Row	start_station_id	end_station_id	route_count
1	514	426	18667
2	435	509	17509
3	519	492	16228
4	435	462	15120
5	426	514	14353

Q5. What is the average trip duration in each month in 2017? Display the results in ascending order by month. Round the result to 3 decimal places. (Hint: use the `EXTRACT(part FROM date_expression)` function and `ROUND` function.)

Row	month	avg_duration
1	4	1085.863
2	5	983.285
3	6	1072.827
4	7	1060.143
5	8	1156.788
6	9	997.089
7	10	1067.82
8	11	892.714
9	12	764.997

Q6. Retrieve the start station name, end station name, and the number of trips between each station pair for the month of February 2018. Filter the results to only include station pairs that have at least 100 trips. Sort the results in descending order of the number of trips and returns the top 10 station pairs.

Row	start_station_name	end_station_name	num_trips
1	E 7 St & Avenue A	Cooper Square & Astor Pl	565
2	Columbus Ave & W 72 St	Central Park West & W 72 St	338
3	S 4 St & Wythe Ave	N 6 St & Bedford Ave	335
4	W 63 St & Broadway	Broadway & W 60 St	328
5	Greenwich Ave & Charles St	Greenwich Ave & Charles St	319
6	N 6 St & Bedford Ave	S 4 St & Wythe Ave	292
7	Willoughby St & Fleet St	Adelphi St & Myrtle Ave	282
8	Bedford Ave & Nassau Ave	N 8 St & Driggs Ave	281
9	W 21 St & 6 Ave	9 Ave & W 22 St	281
10	Pershing Square North	E 24 St & Park Ave S	279

Using the `With` clause

- The SQL WITH clause allows you to give a sub-query block a name (a process also called sub-query refactoring), which can be referenced in several places within the main SQL query.
- The temporary table is a Common Table Expression (CTE).
- Example syntax:

```
WITH tempTable (averageValue) AS
(SELECT AVG(Attr1)
 FROM Table1)
SELECT Attr1
FROM Table1, tempTable
WHERE Table1.Attr1 > tempTable.averageValue;
```

Q7. Rewrite Q6 using the WITH clause. Call the temporary table “popular_stations”.

Row	start_station_name	end_station_name	num_trips
1	E 7 St & Avenue A	Cooper Square & Astor Pl	565
2	Columbus Ave & W 72 St	Central Park West & W 72 St	338
3	S 4 St & Wythe Ave	N 6 St & Bedford Ave	335
4	W 63 St & Broadway	Broadway & W 60 St	328
5	Greenwich Ave & Charles St	Greenwich Ave & Charles St	319
6	N 6 St & Bedford Ave	S 4 St & Wythe Ave	292
7	Willoughby St & Fleet St	Adelphi St & Myrtle Ave	282
8	Bedford Ave & Nassau Ave	N 8 St & Driggs Ave	281
9	W 21 St & 6 Ave	9 Ave & W 22 St	281
10	Pershing Square North	E 24 St & Park Ave S	279

Q8. Find the average age of all female users for each of the top 10 most popular starting stations (based on the number of trips). Sort the results in descending order of the average age.

Row	start_station_name	avg_age
1	8 Ave & W 31 St	46.27247602015...
2	West St & Chambers St	45.46278860448...
3	Pershing Square North	45.31437529691...
4	E 17 St & Broadway	45.25354647576...
5	W 21 St & 6 Ave	44.93504657760...
6	Broadway & E 22 St	44.70652029676...
7	8 Ave & W 33 St	44.322588517171
8	Lafayette St & E 8 St	44.25149033889...
9	Broadway & E 14 St	43.95138193405...
10	Cleveland Pl & Spring St	42.95379687775...

Q9. Write a query to retrieve the number of trips between every pair of the top 5 starting stations and the top 5 ending stations. Sort the result in the number of trips in descending order. Exclude empty station names. (Hint: use WITH to first create two temporary tables for the popular start stations and popular end stations, respectively.)

Row	start_station_name	end_station_name	num_trips
1	West St & Chambers St	West St & Chambers St	14165
2	E 17 St & Broadway	W 21 St & 6 Ave	6563
3	Pershing Square North	Broadway & E 22 St	5999
4	Pershing Square North	E 17 St & Broadway	5878
5	E 17 St & Broadway	Pershing Square North	5490
6	E 17 St & Broadway	Broadway & E 22 St	4649
7	E 17 St & Broadway	E 17 St & Broadway	4618
8	W 21 St & 6 Ave	E 17 St & Broadway	3774
9	Pershing Square North	W 21 St & 6 Ave	3771
10	W 21 St & 6 Ave	Pershing Square North	3508
11	8 Ave & W 31 St	Pershing Square North	3433
12	8 Ave & W 31 St	W 21 St & 6 Ave	3424
13	W 21 St & 6 Ave	W 21 St & 6 Ave	3362
14	W 21 St & 6 Ave	Broadway & E 22 St	3279
15	8 Ave & W 31 St	E 17 St & Broadway	3103
16	8 Ave & W 31 St	Broadway & E 22 St	2937
17	8 Ave & W 31 St	West St & Chambers St	2873
18	Pershing Square North	Pershing Square North	2346
19	W 21 St & 6 Ave	West St & Chambers St	2201
20	West St & Chambers St	W 21 St & 6 Ave	1720
21	West St & Chambers St	Broadway & E 22 St	1448
22	E 17 St & Broadway	West St & Chambers St	1278
23	Pershing Square North	West St & Chambers St	994
24	West St & Chambers St	Pershing Square North	746
25	West St & Chambers St	E 17 St & Broadway	620

Q10. For each of the top 5 starting stations in terms of the most number of trips, retrieve the top 2 ending stations. List the start station name, end station name, number of trips for each pair, and the rank of the end station. (Hint: use RANK() OVER (PARTITION BY start_station_name ORDER BY COUNT(num_trips) DESC) to get the ranking of each end station for each start station.)

Row	start_station_name	end_station_name	total_trips	end_station_rank
1	8 Ave & W 31 St	11 Ave & W 27 St	8786	1
2	8 Ave & W 31 St	9 Ave & W 18 St	6909	2
3	E 17 St & Broadway	W 17 St & 8 Ave	8106	1
4	E 17 St & Broadway	W 21 St & 6 Ave	6563	2
5	Pershing Square North	W 33 St & 7 Ave	12831	1
6	Pershing Square North	E 24 St & Park Ave S	11969	2
7	W 21 St & 6 Ave	9 Ave & W 22 St	17509	1
8	W 21 St & 6 Ave	W 22 St & 10 Ave	15120	2
9	West St & Chambers St	12 Ave & W 40 St	14353	1
10	West St & Chambers St	West St & Chambers St	14165	2

Q11. Find the maximum, minimum, total, average, standard deviation, and variance of the capacity of all stations from the citibike_stations table. Round all decimal numbers to 3 decimal places.

Row	max	min	total	avg	sd	var
1	123	0	68660	31.167	17.169	294.788

Q12. Find all the station names with the pattern “nnn St & xxx Ave” from the citibike_stations table, where nnn is any number and xxx is any text starting with the letters from A to P, e.g., “56 St & Arnold Ave”, “2 St & Park Ave”. (Hint: use the `REGEXP_CONTAINS` function. Regex, or Regular Expressions, is a sequence of characters, used to search and locate specific sequences of characters that match a pattern.)

Basic regular expressions:

	Description	Example	Example matches
`.`	Any character	`.`	a, b, .
`*`	Zero or more of the preceding group	`.*`	a, ab, abab, ‘’ (empty string)
`^`	Beginning of string	`^b.*`	b, baaaa
`$`	End of string	`b.*b$`	bb, baaaab, abab
`[ ]`	Match any one in a set of characters	`[a-cz]`	a, b, c, z
`[^ ]`	Set of characters	`[^a]`	b, c, 1, 2
`( )`	Captured subexpression	`(a.*)`	a, abb
`{m, n}`	Match at least m and at most n of preceding group	`a{2,4}`	aa, aaa, aaaa
`|`	Or, alternation, either one or the other	`a|b`	a, b
`+`	One or more of the proceeding group	`a+`	a, aa, aaa
`?`	Zero or one	`a?`	‘’ (empty string), a

Row	name
1	2 St & Park Ave
2	48 St & Barnett Ave
3	50 St & Barnett Ave
4	44 St & Greenpoint Ave
5	56 St & Arnold Ave
6	53 St & Flushing Ave
7	57 St & Grand Ave
8	61 St & Borden Ave
9	103 St & Martense Ave
10	100 St & Lewis Ave
11	63 St & Borden Ave
12	31 St & Newtown Ave

Q13. Find the top 10 pairs of stations that are closest to each other from the citibike_stations table. (Hint: use the `ST_GEOGPOINT` and `ST_DISTANCE` functions, and use the `CROSS JOIN` operation for a self-join.)

Row	station1	station2	distance
1	9 Ave & W 18 St	W 18 St & 9 Ave	39.95704484248...
2	E 58 St & 1 Ave (NE Corner)	E 58 St & 1 Ave (NW Corner)	40.13894509487...
3	Eastern Pkwy & Franklin Ave (N...	Eastern Pkwy & Franklin Ave (S...	53.36064528832...
4	Degraw St & 3 Ave	Douglass St & 3 Ave	58.47543382234...
5	Broadway & W 37 St	W 37 St & Broadway	62.23820670180...
6	President St & 4 Ave	Union St & 4 Ave	68.55058381580...
7	Ave A & E 14 St	E 13 St & Ave A	73.65765678833...
8	2 Ave & 37 St	2 Ave & 39 St	74.44908430018...
9	Gansevoort St & Hudson St	Hudson St & W 13 St	75.93700620361...
10	Clinton Ave & Flushing Ave	Flushing Ave & Vanderbilt Ave	76.48228274423...