



# Storing and Exporting Data

Evan Jones

Welcome back. You're now in course number two of the data analyst course series.

Here's where we get to look at fun concepts like bringing new data into BigQuery and visualizing it.

## Storing and exporting data

01 Compare permanent vs temporary tables

02 Save and export query results

03 Performance preview: Query cache



Now, one of the core building blocks of data analysis is creating and running your SQL queries on raw data sources, and then saving those results into new tables that you can access later.

And here, we'll cover the difference between permanent and temporary data tables, and how to store the results of your queries.

## How to create a new permanent table

1. Write SQL query
2. Click **More > Query Settings**
3. Specify the **destination table** (can be existing)
4. Choose **Write Preference** (if table already exists)
5. Run query

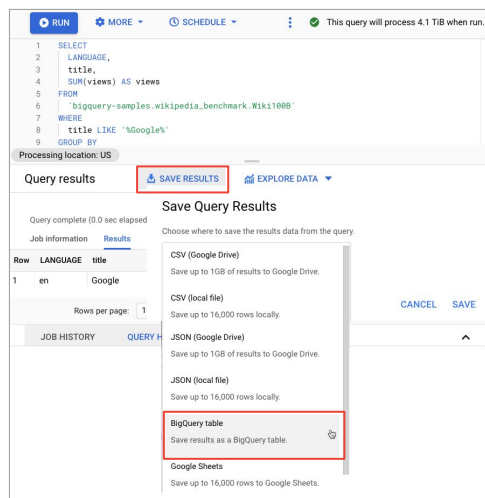
### If the destination table exists:

- Write if empty
- Append records
- Overwrite table

Okay so, so far you've queried data sets that have already existed out in the public space in BigQuery. Now it's time to create your own and store some new tables permanently for you to access later. And the benefit here again is, if you had a SQL query and you have these amazing insights and results and you want to store it, you needed to do so in a permanent table or some of the other options that we're going to show you.

## Forgot to specify a table? Store query results after running

- Use **SAVE RESULTS > BigQuery table**.
- **All query results are stored in tables** (regardless if you save as table)
- If you don't save as a permanent table, a temporary one is automatically created and saved for 24 hours
- Re-running the same query will likely hit the cached temporary table



Now here's the really kind of mind blowing part. Even if you don't explicitly save a query as a permanent table by clicking on that Save button or specifying a destination table. Once you run that query, any and all query, right? It actually, BigQuery behind the scenes, will store the results of that query in its own anonymous, temporary table. And the reason why it does that is so we can actually rerun that same query and get the benefit of pulling from query cache. Which as you saw in the Wikipedia example in the prior course, and a few of the other queries that we've been executing, is extremely fast.

## All query results are saved to a table

- 01 All query results are saved to either a temporary or permanent table
- 02 If you specify a destination table then that table becomes permanent otherwise it's a new temporary table
- 03 Temporary tables are the basis of query cached results
- 04 Temporary tables last 24 hours only

Okay, so here's the key lesson. All query results, again, are stored as a temporary or permanent table. And it's your choice whether or not, before the query runs or after the query runs, if you want to store it into that permanent table. But regardless, those results are going to be stored in a temporary table, and temporary tables last for about 24 hours.

Running the same queries will pull from cache

Cache = Faster Results

Let's talk a little bit more about query cache.

This is this magical concept of creating those really fast queries.

## Running the same queries will pull from cache

Cache is **not used** when:

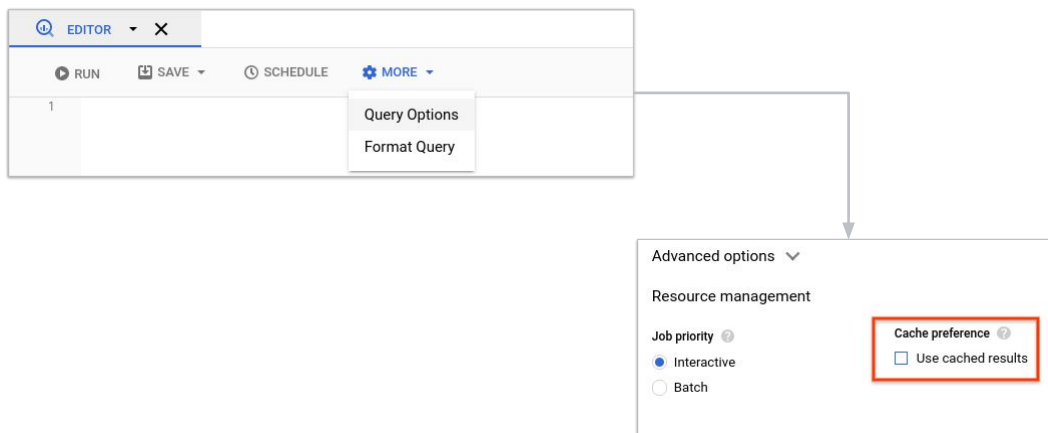
- ✗ Deterministic queries used (like `CURRENT_TIMESTAMP()`)
- ✗ Underlying table(s) updated

Now if you have what's called a deterministic query, if at runtime you don't know what the answer is, like if you're pulling things like the current timestamp for example, you can't rely on cache, right? Because that's something that's going to be changing depending upon when you're running it and then naturally that can't be cached.

Now naturally, cache is already what we call broken or busted. Cache is busted when the underlying tables themselves have been updated.

So if there was a data change in the table, what BigQuery will do is it will compare the timestamps from when you ran that last query, what was that table last modified date? If that has since changed, then it will automatically be smart enough to not rely on the query cache and will invalidate that cache.

## Disable the Used cache results option

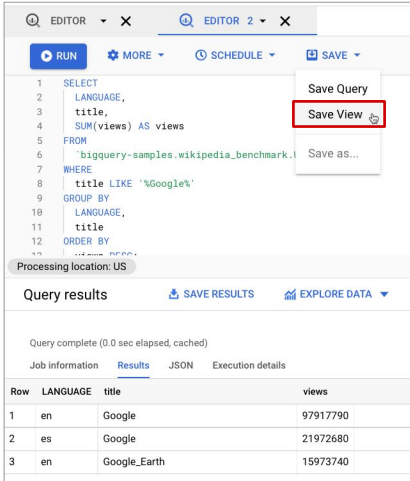


But you can manually do it yourself inside of the Options should you want. Say, if you're timing a query for performance reasons, you don't necessarily want to rely on cache as it might not always be there.



## Storing results in a View

- View = Saved SQL query (a virtual table)
- The underlying query is executed each time the view is accessed



The screenshot shows the Google Cloud BigQuery console. At the top, there are tabs for 'EDITOR' and 'EDITOR 2'. Below the tabs are buttons for 'RUN', 'MORE', 'SCHEDULE', and 'SAVE'. The 'SAVE' button is open, showing a dropdown menu with options: 'Save Query', 'Save View' (highlighted with a red box), and 'Save as...'. The SQL query in the editor is as follows:

```

1 SELECT
2   LANGUAGE,
3   title,
4   SUM(vIEWS) AS views
5 FROM
6   `bigquery-samples.wikipedia_benchmark.`
7 WHERE
8   title LIKE '%Google%'
9 GROUP BY
10  LANGUAGE,
11  title
12 ORDER BY

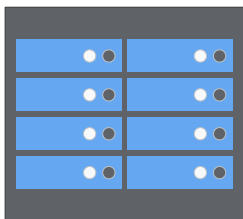
```

Below the query editor, it says 'Processing location: US'. Under 'Query results', there are links for 'SAVE RESULTS' and 'EXPLORE DATA'. The query is complete (0.0 sec elapsed, cached). Below the 'Results' tab, there is a table with the following data:

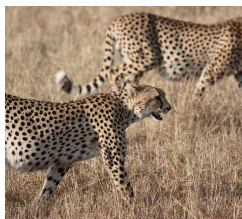
Row	LANGUAGE	title	views
1	en	Google	97917790
2	es	Google	21972680
3	en	Google_Earth	15973740

So we talked about permanent tables, temporary tables, and query cache were the last topics that we want to cover. It's actually storing your query as a View. Now a View, all it is, is you can take your query, and you can save it. Almost like a table, but it reruns that same query that you have on the underlying data source.

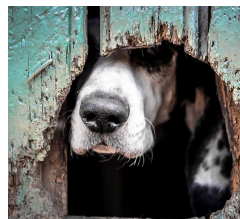
## Summary: Save and store query results in BigQuery



All query results are stored as tables (temporary or permanent)



Pulling from cached results is fastest but not always possible

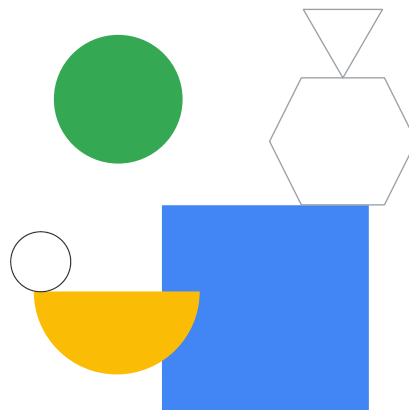


Views in BigQuery are logical and are often used to restrict row-level access

Now in this session, we covered how you can permanently store the results of your queries into tables. You can then query those tables later. One of the key things to remember is that BigQuery will store the results of all your queries into temporary tables even if you don't specify anything. And these temp tables are around for about 24 hours. And they form the basis of queried cache. Lastly, we covered views, which are saved queries on top of your existing tables. And you can modify the sequel behind the view at any time. Let's get some practice saving queries into tables and views in our next lab.

## Lab Intro

Creating Permanent Tables and  
Access-Controlled Views in  
BigQuery



All right, it's time to create some new permanent tables and views.

Now that you're familiar with working with your public dataset, it's time to create a dataset of your very own.

Now in this lab, you'll create a dataset, add in a few tables and views, and permanently store those results of the queries that you've run.

Let's jump right in.