Module 6

Storing and Exporting Data

In this module we will:

- Compare Permanent vs Temporary Tables
- Save and Export Query Results
- Performance Preview: Query Cache

Google Cloud

One of the building blocks of data analysis is creating SQL queries on raw data sources and then saving your results into new reporting tables that you can then query from. 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 **Show Options**
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

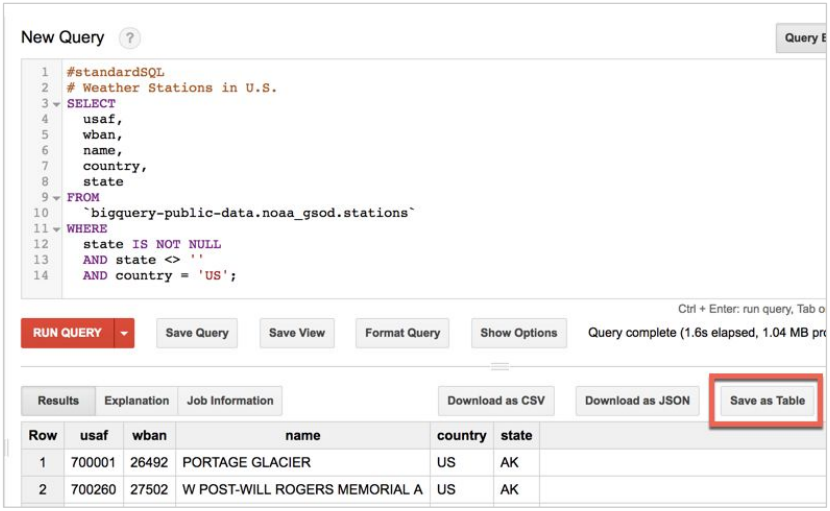
Google Cloud

Query results are always saved to either a [temporary or permanent table](). You can choose whether to append or overwrite data in an existing table and whether to create a new table if none exists by that name.

Also: Alternately, if you forget to specify a destination table before running your query, you can copy the temporary table to a permanent table by clicking the Save as Table button in the results window.

More details:
https://cloud.google.com/bigquery/querying-data#queries

Forgot to Specify a Table? **Store Query Results After Running**



- Use **Save as 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

Google Cloud

All query results end up in a table (even if you dont explicitly create one)

If you don't create one, a temp one is created (per user, per project) and is saved for 24 hours. Same queries will use cached results when available assuming:

- Underlying table not modified
- No CURRENT_DATE() or similar function is used

See if it's used after execution time (will say cached result)

Can be disabled in Show Options

All Query Results are Saved to a Table

- All Query Results are saved to either a Temporary or Permanent Table
- If you specify a Destination Table then that table becomes Permanent otherwise it's a new Temporary Table
- Temporary Tables are the basis of Query Cached Results
- Temporary Tables last 24 hours only

Google Cloud

Alternately, if you forget to specify a destination table before running your query, you can copy the temporary table to a permanent table by clicking the Save as Table button in the results window.

Running the Same Queries will Pull from **Cache**

```
#standardSQL
# Weather Stations in U.S.
SELECT
  usaf,
  wban,
  name,
  country,
  state
FROM
  `bigquery-public-data.noaa_gsod.stations`
WHERE
  state IS NOT NULL
  AND state <> ''
  AND country = 'US';
```

Ctrl + Enter: run query, Tab or Ctrl +

RUN QUERY | Save Query | Save View | Format Query | Show Options | Query complete (0.8s elapsed, cached)

Cache = Faster Results

*Cache is **not used** when:*

- Underlying table(s) updated
- Deterministic queries used (like CURRENT_TIMESTAMP( )
- Cache disabled in Show Options

Google Cloud

https://cloud.google.com/bigquery/querying-data#query-caching

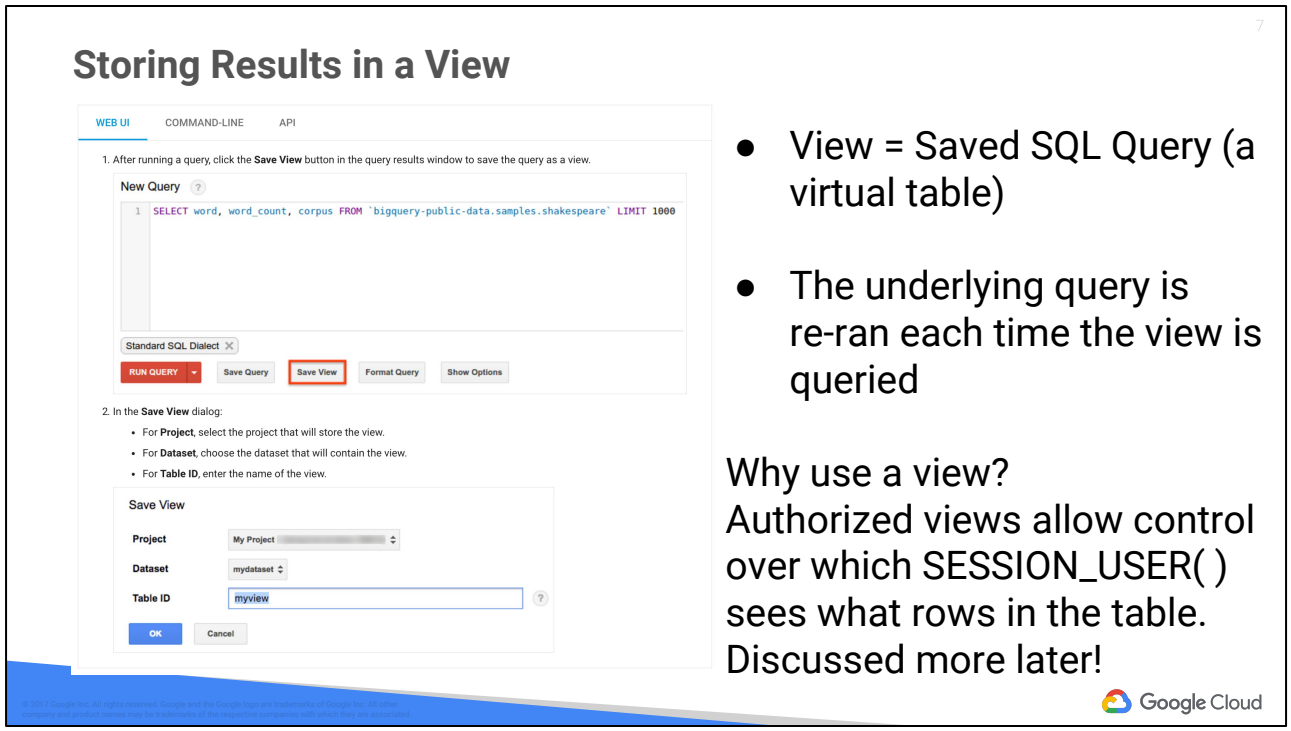
BigQuery writes all query results to a table. The table is either explicitly identified by the user (a destination table), or it is a temporary, cached results table. Temporary, cached results tables are maintained per-user, per- project.

When you run a duplicate query, BigQuery attempts to reuse cached results. When query results are retrieved from a cached results table, the job statistics property statistics.query.cacheHit returns as true, and you are not charged for the query. Though you are not charged for queries that use cached results, the queries are subject to the BigQuery quota policies. In addition to reducing costs, queries that use cached results are significantly faster because BigQuery does not need to compute the result set.

All query results, including both interactive and batch queries, are cached in temporary tables for approximately 24 hours with some exceptions. Query results are not cached:

- When a destination table is specified in the job configuration, the web UI, the command line, or the API
- If any of the referenced tables or logical views have changed since the results were previously cached
- When any of the tables referenced by the query have recently received streaming inserts (a streaming buffer is attached to the table) even if no new rows have arrived

- If the query uses non-deterministic functions; for example, date and time functions such as `CURRENT_TIMESTAMP()` and `NOW()`, and other functions such as `CURRENT_USER()` return different values depending on when a query is executed
- If you are querying multiple tables using a wildcard
- If the cached results have expired; typical cache lifetime is 24 hours, but the cached results are best-effort and may be invalidated sooner



A view is a virtual table defined by a SQL query. You can query views in BigQuery using the web UI, the command-line tool, or the API. You can also use a view as a data source for a visualization tool such as Google Data Studio.

BigQuery's views are logical views, not materialized views. Because views are not materialized, the query that defines the view is run each time the view is queried. Queries are billed according to the total amount of data in all table fields referenced directly or indirectly by the top-level query. For more information, see query pricing.

SQL queries used to define views are subject to the standard query quotas.

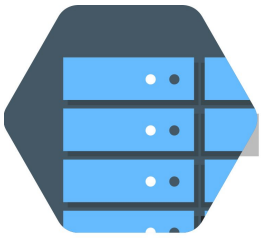
Limitations

BigQuery views are subject to the following limitations:

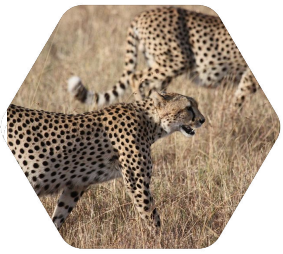
- You cannot run a BigQuery job that exports data from a view.
- You cannot use the TableDataList JSON API method to retrieve data from a view. For more information, seeTabledata: list.
- You cannot mix standard SQL and legacy SQL queries when using views. A standard SQL query cannot reference a view defined using legacy SQL syntax.
- The schemas of the underlying tables are stored with the view when the view is created. If columns are added, deleted, and so on after the view is created,

- You cannot update a legacy SQL view to standard SQL in the BigQuery web UI. You can change the SQL language using the command-line tool `bq update --view` command or by using the update or patch API methods.
- You cannot include a user-defined function in the SQL query that defines a view.
- You cannot reference a view in a wildcard table query.
- BigQuery supports up to four levels of nested views in legacy SQL. If there are more than four levels, an `INVALID_INPUT` error returns. In standard SQL, you are limited to 100 levels of nested views.
- You are limited to 1,000 authorized views per dataset.

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

Google Cloud

In this short section we covered how you can permanently store the results of your queries in tables to retrieve them later. One of the key things to remember is that BigQuery will store the results of all queries into temporary tables even if you don't specify anything. These temp tables are around for 24 hours and are the basis of query cache.

Lastly, we covered views which are like saved queries on top of existing tables.

Let's get some practice saving queries into tables and views in our next lab.

Image (data blocks) cc0: https://cloud.google.com/data-transfer/
Image (cheetah) cc0: https://pixabay.com/en/africa-south-africa-wild-nature-171315/
Image (dog) cc0: https://pixabay.com/en/animals-dogs-domesticated-pets-2607753/

Lab 5
Creating New Permanent Tables and Views

Google Cloud

Lab 5 in Qwiklabs

Creating new Permanent Tables and Views

Core to BigQuery are the concepts of datasets, tables, and views. Learn how to create your own datasets, tables, and how to store and export query results.

Google Cloud

Image (data blocks) cc0: https://cloud.google.com/data-transfer/