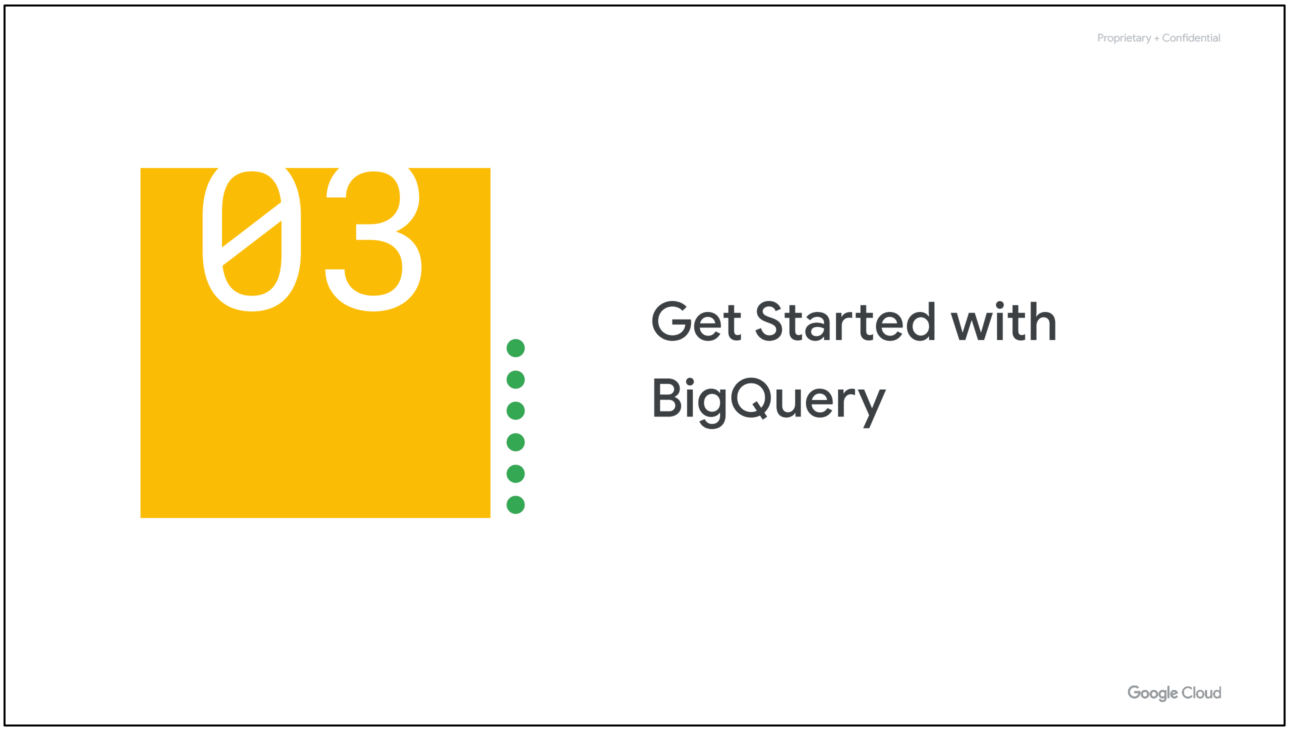
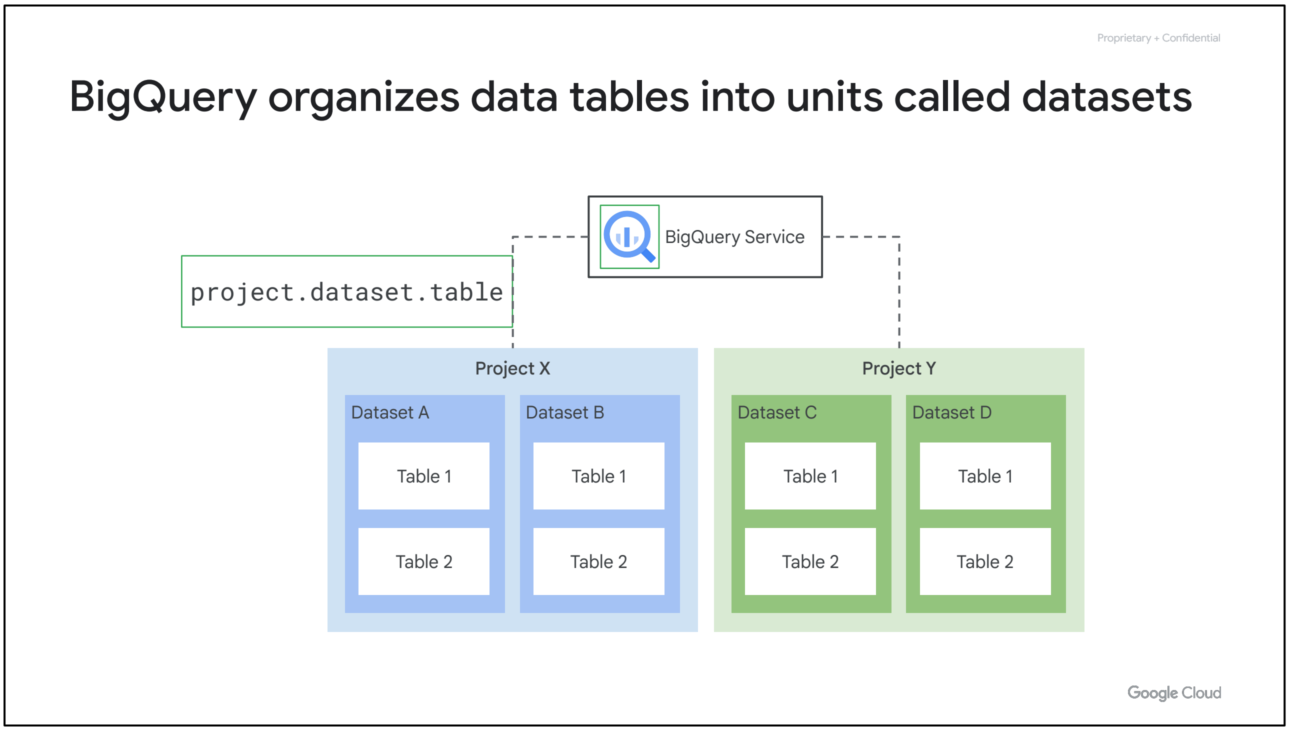


03

Get Started with BigQuery

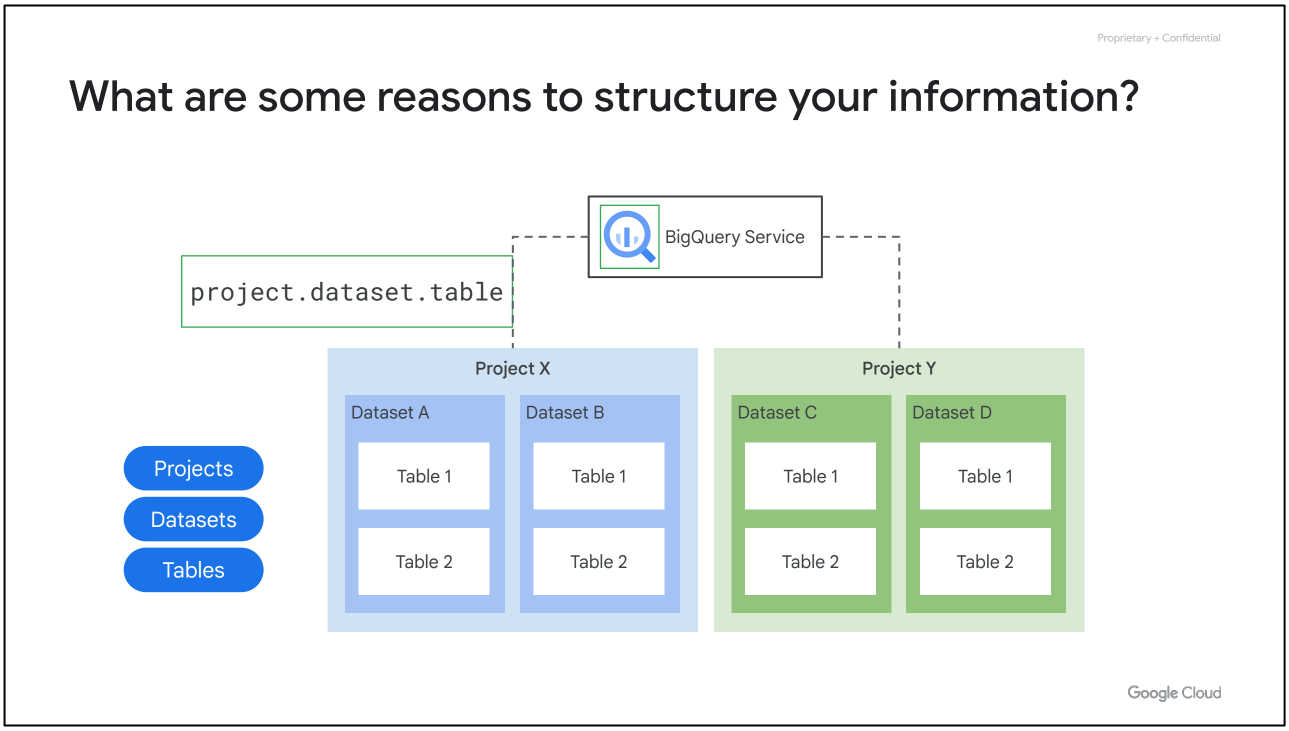
Google Cloud

Now that you're familiar with the basics of BigQuery, it's time to talk about how BigQuery organizes your data.

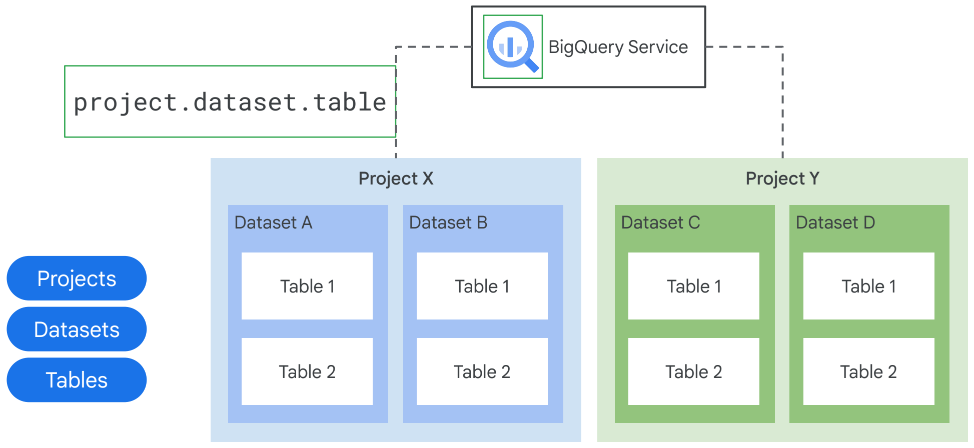


BigQuery organizes data tables into units called datasets

BigQuery organizes data tables into units called datasets. These datasets are scoped to your Google Cloud project. When you reference a table from the command line in SQL queries or in code, you refer to it by using the construct: project.dataset.table.

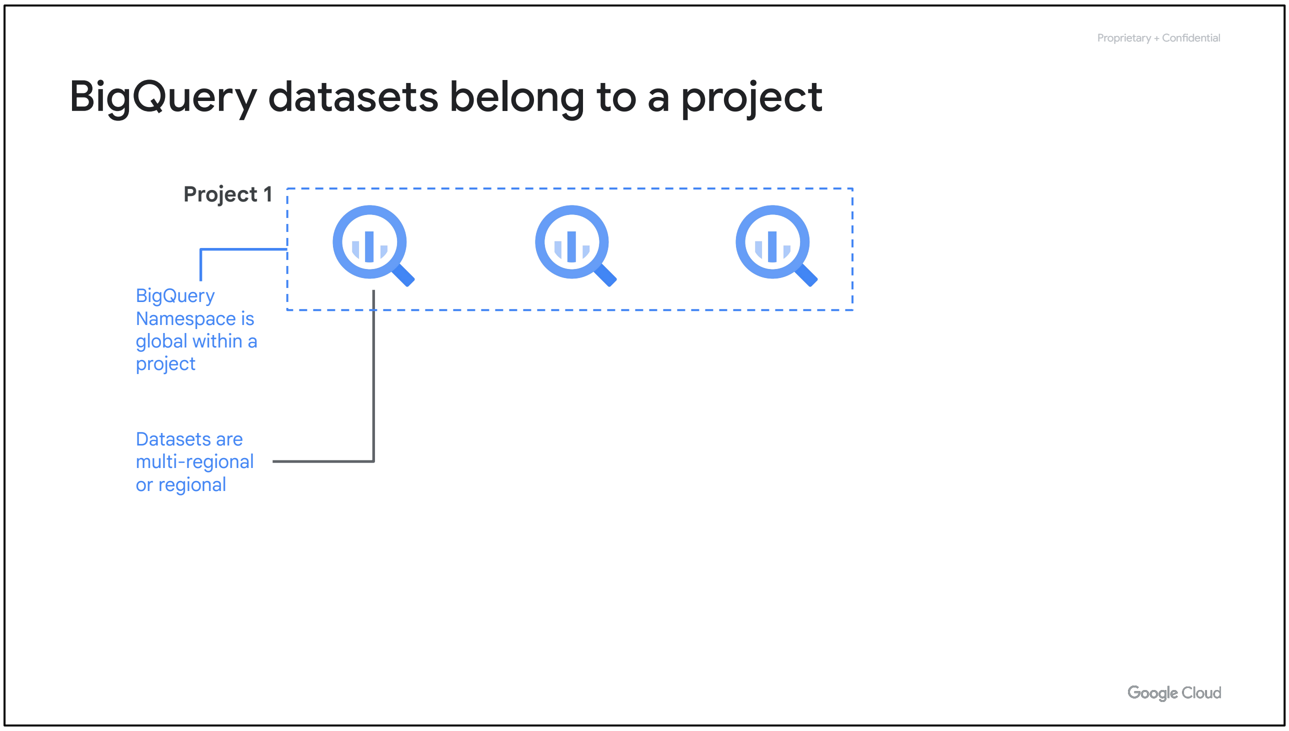


What are some reasons to structure your information?



What are some reasons to structure your information into datasets, projects, and tables? These multiple scopes—project, dataset, and table—can help you structure your information logically. You can use multiple datasets to separate tables pertaining to different analytical domains, and you can use project-level scoping to isolate datasets from each other according to your business needs.

Also, as we will discuss later, you can align projects to billing and use datasets for access control. You store data in separate tables based on logical schema considerations.



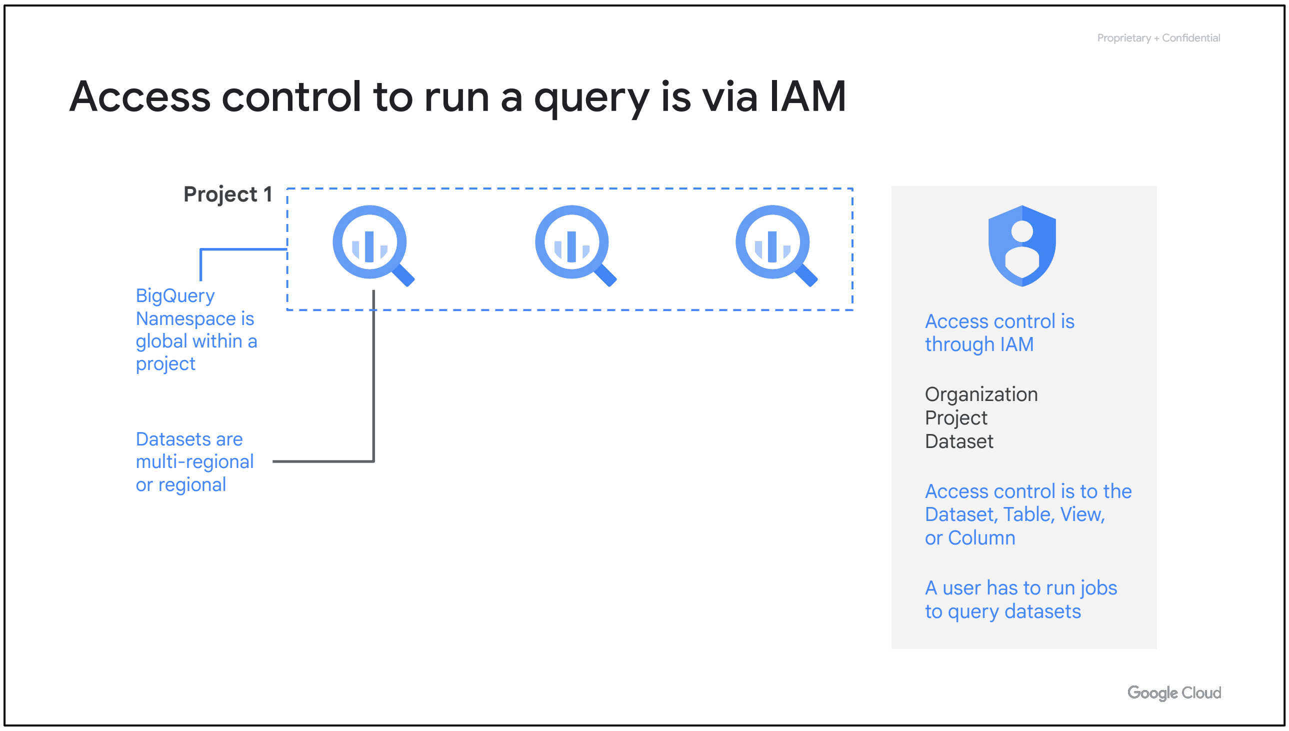
BigQuery datasets belong to a project

Project 1
BigQuery Namespace is global within a project
Datasets are multi-regional or regional

The project is what the billing is associated with. For example, if you queried a table that belong to the `bigquery-public-data` project, the storage costs are billed to that data project.

To run a query, you need to be logged in to the Cloud Console. You will run a query in your own Google Cloud project and the query charges are billed to your project, not to the public data project.

In order to run a query in a project, you need Identity Access Management (IAM) permission to submit a job. Remember that running a query means that you must be able to submit a query job to the service.



Access control to run a query is via IAM

Project 1
BigQuery Namespace is global within a project
Datasets are multi-regional or regional

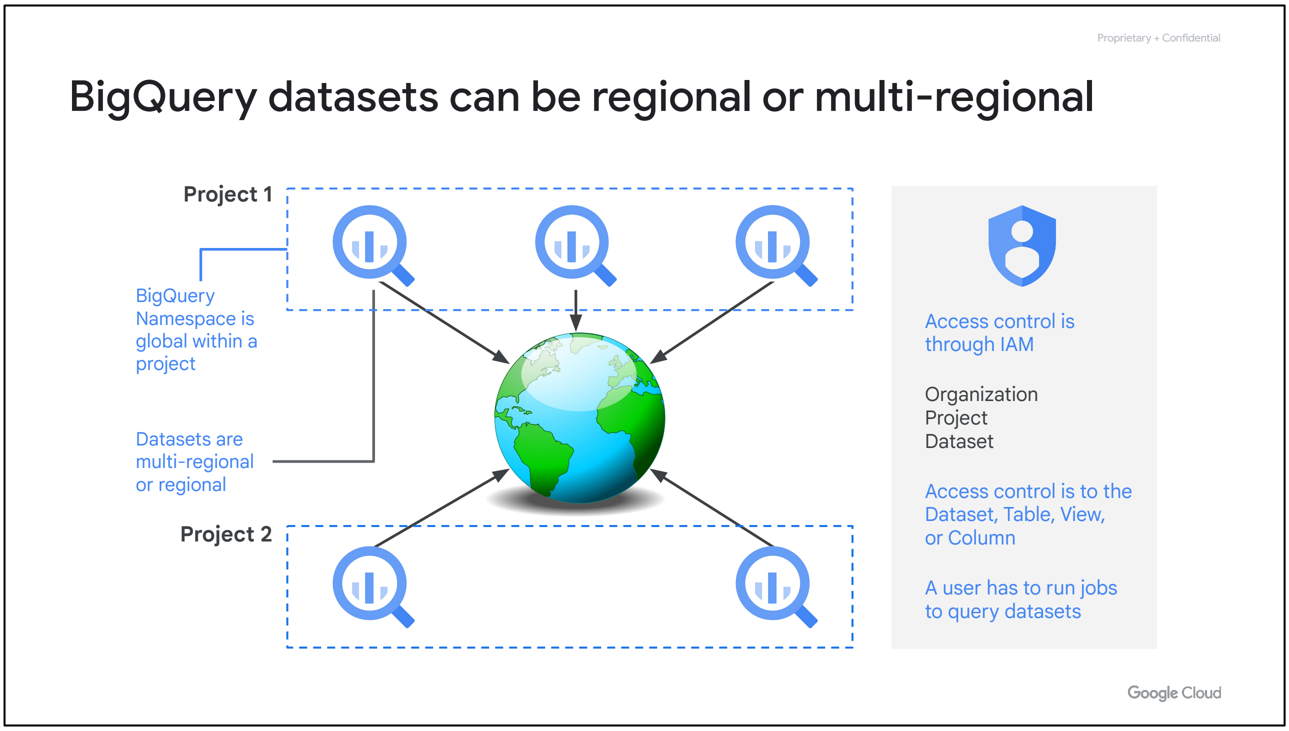
Access control is through IAM

Organization
Project
Dataset

Access control is to the Dataset, Table, View, or Column

A user has to run jobs to query datasets

Access control is through IAM and is at the dataset, table/view, or column level. In order to query data in a table or view, you need at least read permissions on the table or view.



BigQuery datasets can be regional or multi-regional

Project 1
BigQuery Namespace is global within a project
Datasets are multi-regional or regional
Project 2

Access control is through IAM

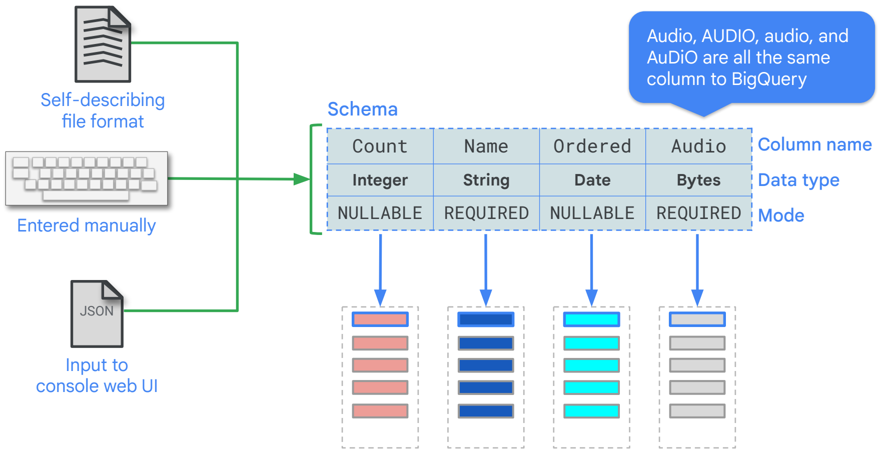
Organization
Project
Dataset

Access control is to the Dataset, Table, View, or Column

A user has to run jobs to query datasets

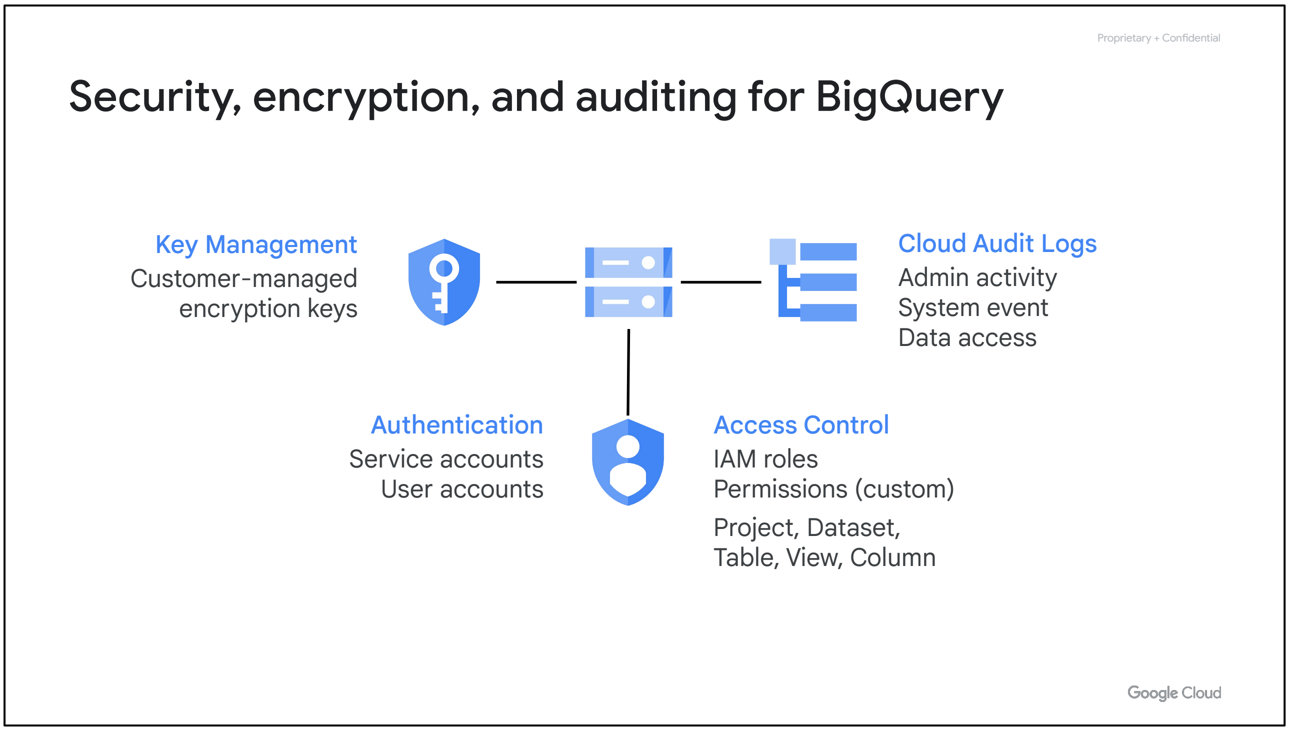
Like Cloud Storage, BigQuery datasets can be regional or multi-regional. Regional datasets are replicated across multiple zones in the region. Multi-regional means replication among multiple regions.

The table schema provides structure to the data


Self-describing file format
Entered manually
JSON
Input to console web UI
Audio, AUDIO, audio, and AuDiO are all the same column to BigQuery
Schema
Column name
Data type
Mode

Count	Name	Ordered	Audio
Integer	**String**	**Date**	**Bytes**
NULLABLE	REQUIRED	NULLABLE	REQUIRED

Every table has a schema. You can enter the schema manually through the Cloud Console, or by supplying a JSON file.



Proprietary + Confidential

Security, encryption, and auditing for BigQuery

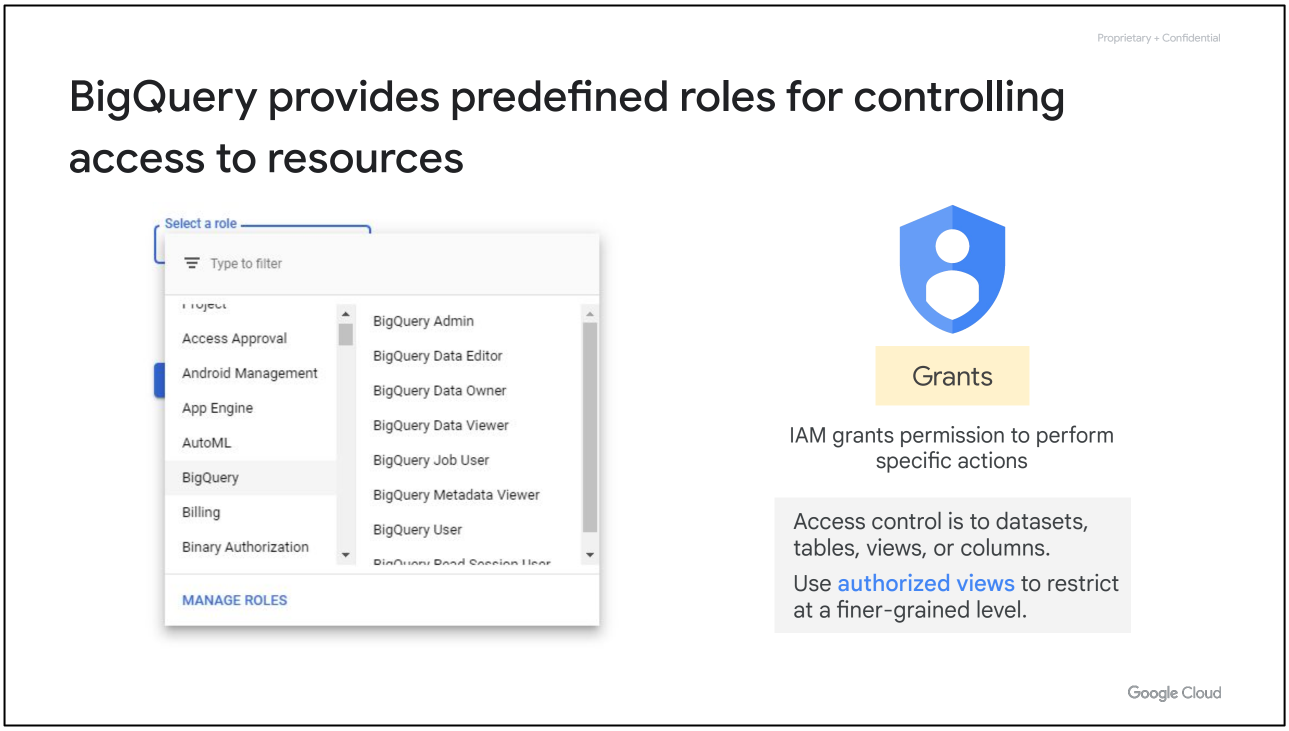
Key Management
As with Cloud Storage, BigQuery storage encrypts data at rest and over the wire using Google-managed encryption keys. It's also possible to use customer-managed encryption keys.

Authentication
Authentication is through IAM, and so it's possible to use Gmail addresses or Google Workspace accounts for this task.

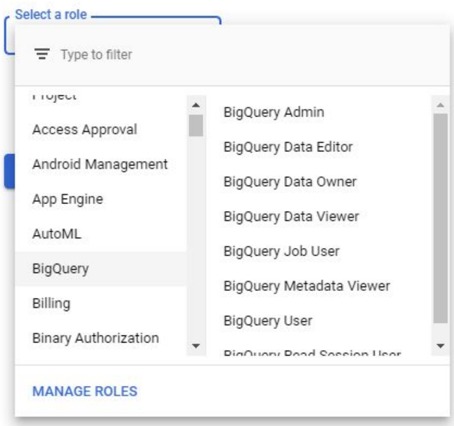
Access control
Access control is through IAM roles and involves giving permissions. We discussed two of those in read access and the ability to submit query jobs. However, many other permissions are possible.
Remember that access control is at the level of datasets, tables, views, or columns. When you provide access to a dataset, either read or write, you provide access to all the tables in that dataset.

Cloud Audit Logs
Logs in BigQuery are immutable and are available to be exported to Cloud Operations. Admin activities and system events are all logged. An example of a system event is table expiration.

If, when creating a table, you configure it to expire in 30 days, at the end of 30 days a system event will be generated and logged. You will also get immutable logs of every access that happens to a dataset under your project.



BigQuery provides predefined roles for controlling access to resources



Grants

IAM grants permission to perform specific actions

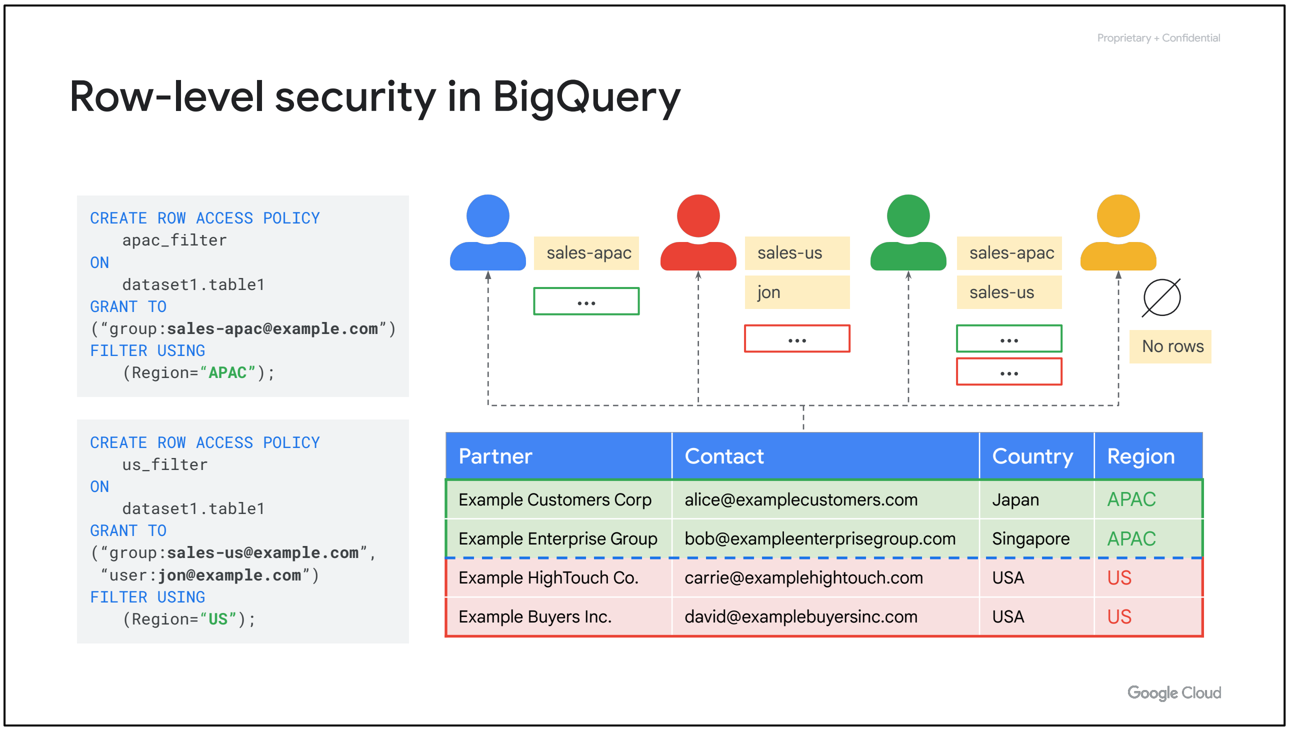
Access control is to datasets, tables, views, or columns.

Use authorized views to restrict at a finer-grained level.

Google Cloud

BigQuery provides predefined roles for controlling access to resources. You can also create custom IAM roles consisting of your defined set of permissions, and then assign those roles to users or groups. You can assign a role to a Google email address or to a Google Workspace Group.

An important aspect of operating a data warehouse is allowing shared but controlled access against the same data to different groups of users. For example, finance, HR, and marketing departments all access the same tables, but their levels of access differ. Traditional data warehousing tools make this possible by enforcing row-level security. You can achieve the same results in BigQuery with access control to datasets, tables, views, or columns, or by defining authorized views and row-level permissions.



Row-level security in BigQuery

```
CREATE ROW ACCESS POLICY
   apac_filter
ON
   dataset1.table1
GRANT TO
("group:sales-apac@example.com")
FILTER USING
   (Region="APAC");
```

```
CREATE ROW ACCESS POLICY
   us_filter
ON
   dataset1.table1
GRANT TO
("group:sales-us@example.com",
 "user:jon@example.com")
FILTER USING
   (Region="US");
```

Partner	Contact	Country	Region
Example Customers Corp	alice@examplecustomers.com	Japan	APAC
Example Enterprise Group	bob@exampleenterprisegroup.com	Singapore	APAC
Example HighTouch Co.	carrie@examplehightouch.com	USA	US
Example Buyers Inc.	david@examplebuyersinc.com	USA	US

Google Cloud

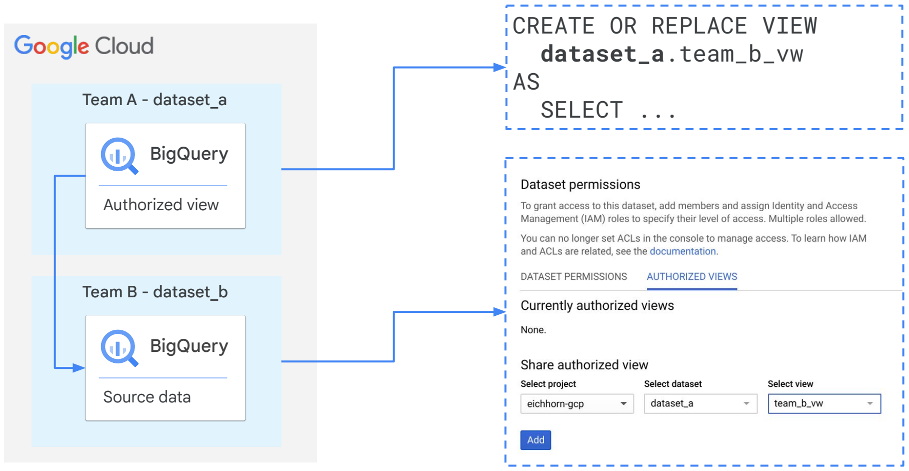
In BigQuery, row-level security involves the creation of row-level access policies on a target BigQuery table. This policy then acts as a filter to hide or display certain rows of data, depending on whether a user or group is in an allowed list.

An authorized user, with the IAM roles BigQuery Admin or BigQuery DataOwner, can create row-level access policies on a BigQuery table.

When you create a row-level access policy, you specify the table by name, and which users or groups (called the grantee-list) should have access to certain row data. The policy includes the data on which you wish to filter, called the filter_expression. The filter_expression functions like a WHERE clause in a typical query.

In this example, users in the group:apac can only see partners from the APAC region.

Creating an authorized view



```
CREATE OR REPLACE VIEW
  dataset_a.team_b_vw
AS
  SELECT ...
```

Dataset permissions

To grant access to this dataset, add members and assign Identity and Access Management (IAM) roles to specify their level of access. Multiple roles allowed.

You can no longer set ACLs in the console to manage access. To learn how IAM and ACLs are related, see the documentation.

Currently authorized views

None.

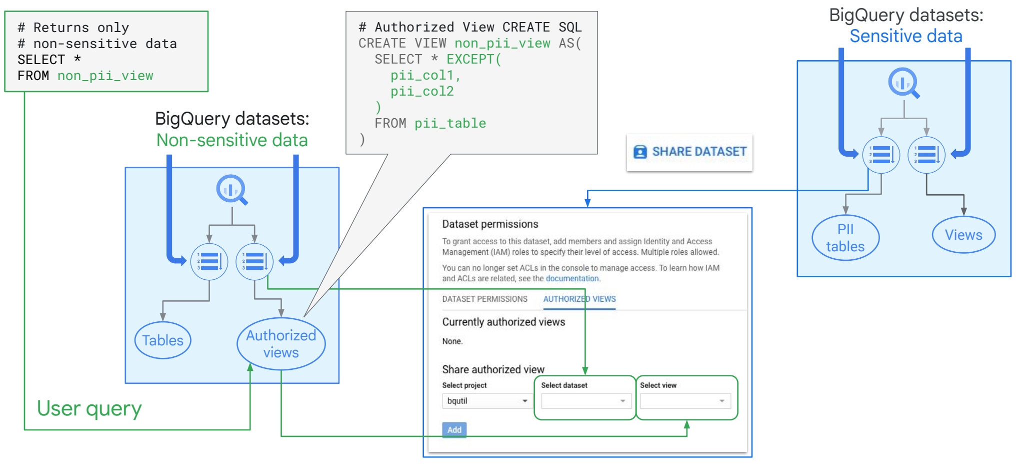
Share authorized view

Select project	Select dataset	Select view
eichhorn-gcp	dataset_a	team_b_vw

Giving view access to a dataset is also known as creating an authorized view in BigQuery. An authorized view allows you to share query results with particular users and groups without giving them access to the underlying source data.

[Creating authorized views: https://cloud.google.com/bigquery/docs/authorized-views]

Protect columns with authorized views



```
# Returns only
# non-sensitive data
SELECT *
FROM non_pii_view
```

```
# Authorized View CREATE SQL
CREATE VIEW non_pii_view AS(
  SELECT * EXCEPT(
    pii_col1,
    pii_col2
  )
  FROM pii_table
)
```

Dataset permissions

To grant access to this dataset, add members and assign Identity and Access Management (IAM) roles to specify their level of access. Multiple roles allowed.

You can no longer set ACLs in the console to manage access. To learn how IAM and ACLs are related, see the documentation.

DATASET PERMISSIONS | AUTHORIZED VIEWS

Currently authorized views

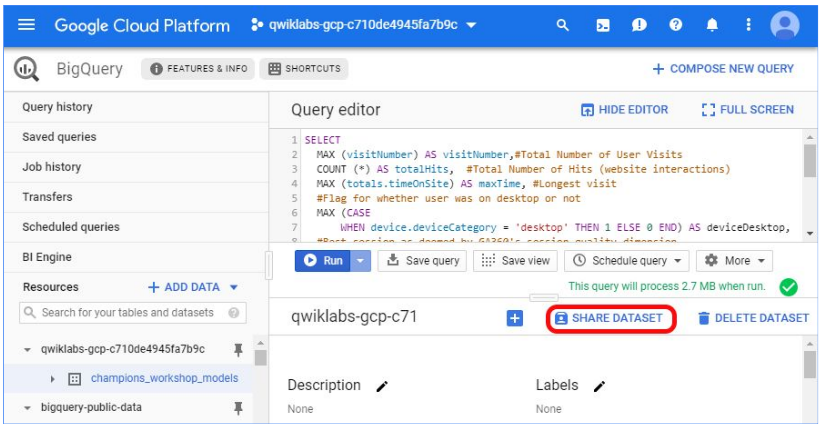
None.

Share authorized view

Select project: bqutil
Select dataset:
Select view:

You can also use the view's SQL query to restrict the columns (fields) the users are able to query.

It’s easy to share access to datasets with other analysts



```
SELECT
  MAX (visitNumber) AS visitNumber,#Total Number of User Visits
  COUNT (*) AS totalHits,  #Total Number of Hits (website interactions)
  MAX (totals.timeOnSite) AS maxTime, #Longest visit
  #Flag for whether user was on desktop or not
  MAX (CASE
      WHEN device.deviceCategory = 'desktop' THEN 1 ELSE 0 END) AS deviceDesktop,
```

This query will process 2.7 MB when run.

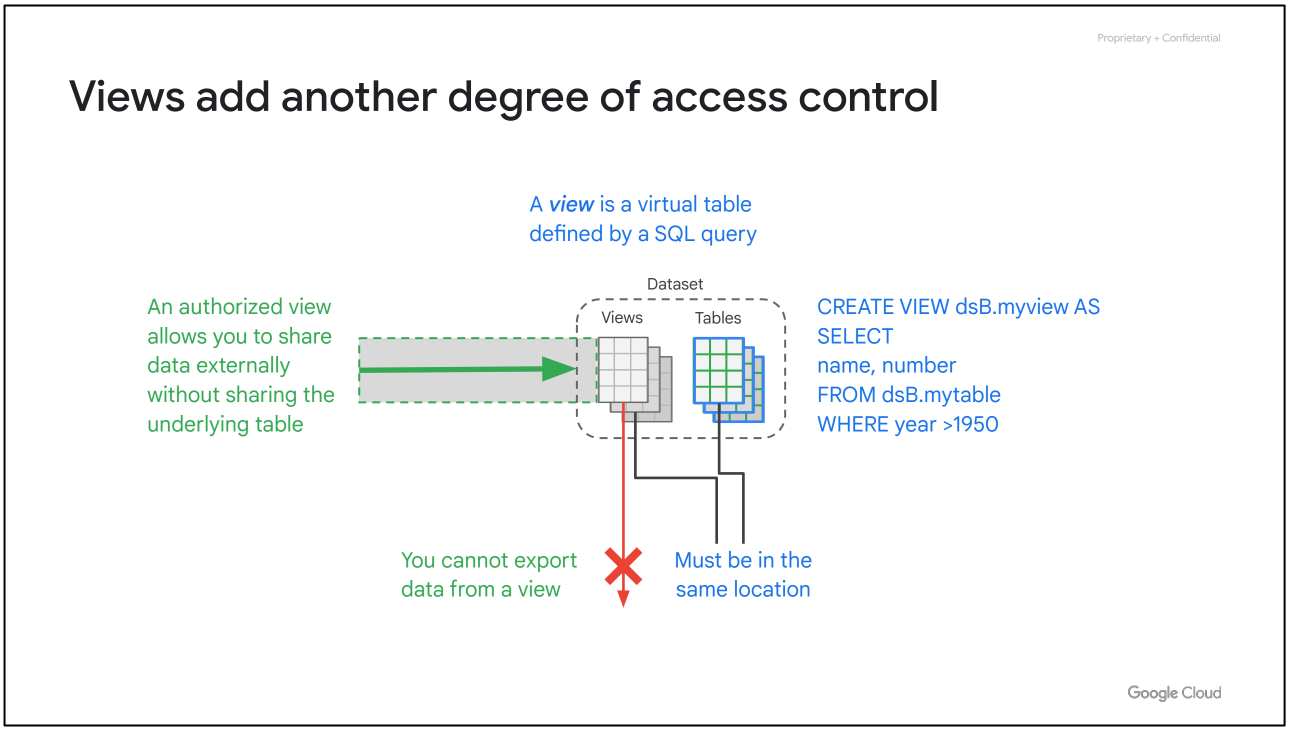
Sharing access to datasets is easy.

Traditionally, onboarding new data analysts involved significant lead time. To enable analysts to run simple queries, you had to show them where data sources resided and set up ODBC connections and tools and access rights. Using Google Cloud, you can greatly accelerate an analyst's time to productivity.

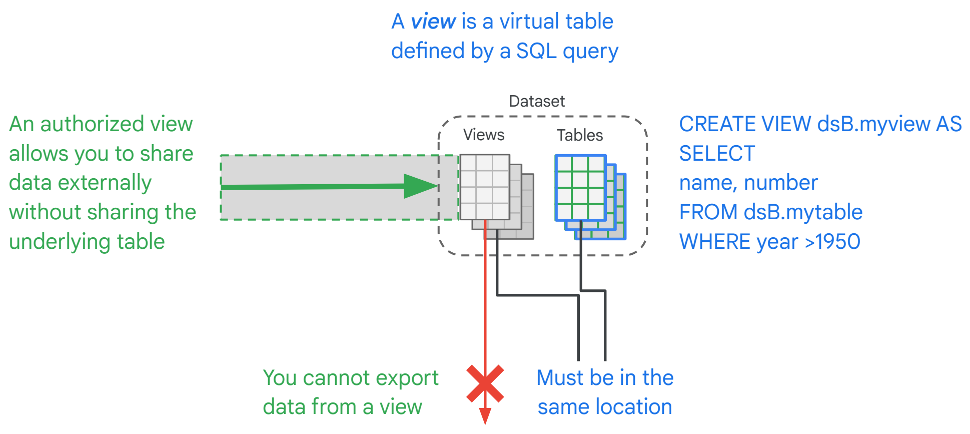
To onboard an analyst on Google Cloud, you grant access to relevant project(s), introduce them to the Cloud Console and BigQuery web UI, and share some queries to help them get acquainted with the data.

The Cloud Console provides a centralized view of all assets in your Google Cloud environment. The most relevant asset to data analysts might be Cloud Storage buckets, where they can collaborate on files.

The BigQuery web UI presents the list of datasets that the analyst has access to. Analysts can perform tasks in the Cloud Console according to the role you grant them, such as viewing metadata, previewing data, executing, and saving and sharing queries.



Views add another degree of access control

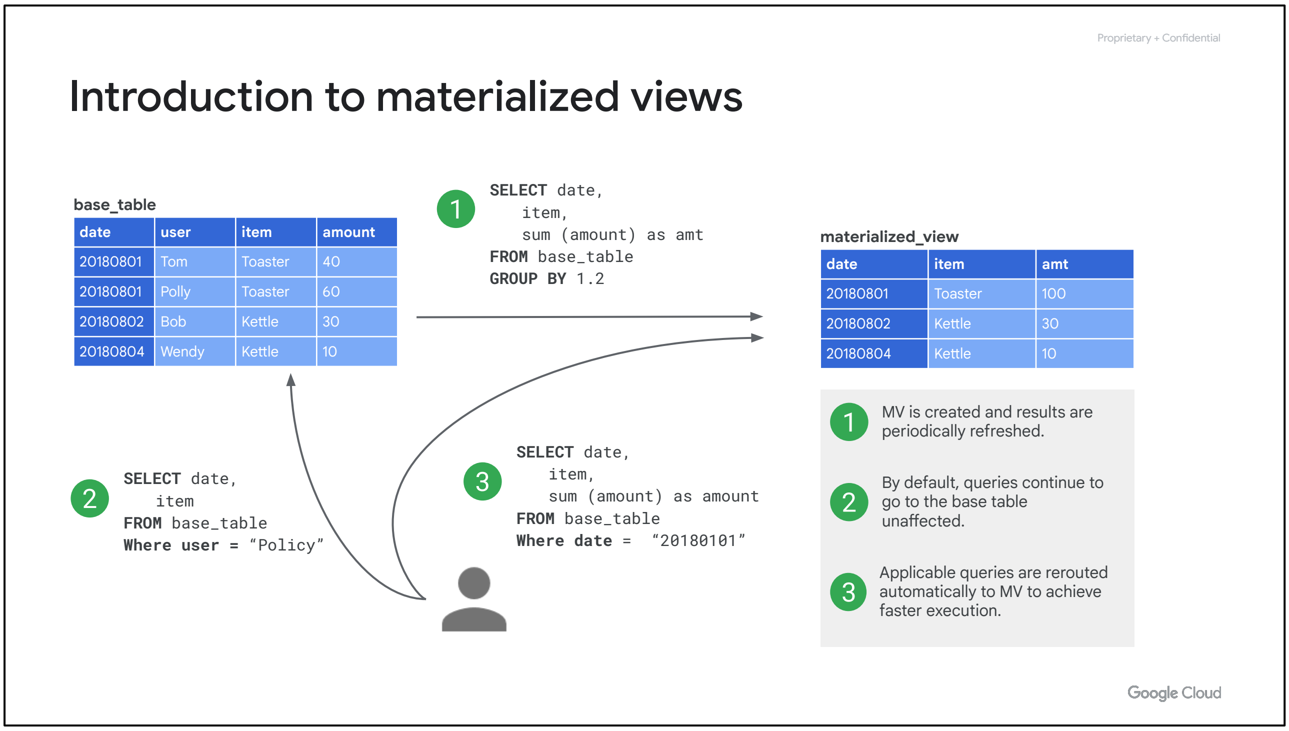

A *view* is a virtual table defined by a SQL query
Dataset
Views
Tables
An authorized view allows you to share data externally without sharing the underlying table
You cannot export data from a view
Must be in the same location

```
CREATE VIEW dsB.myview AS
SELECT
name, number
FROM dsB.mytable
WHERE year >1950
```

When you provide read access to a dataset to a user, every table in that dataset is readable by that user.

What if you want more fine-grained control? In addition to access controls at the table or column level, you can use views. In this example, we are creating a view in Dataset B, and the view is a subset of the table data in Dataset A. Now, by providing users with access to Dataset B, we are creating an authorized view that is only a subset of the original data. Note that you cannot export data from a view, and dataset B has to be in the same region or multi-region as dataset A.

A view is a SQL query that looks like and has properties similar to a table. You can query a view just like you query a table. BigQuery supports materialized views as well. These are views that are persisted so that the table does not need to be queried every time the view is used. BigQuery will keep the materialized view refreshed and up to date with the contents of the source table.



Proprietary + Confidential

Introduction to materialized views

base_table

date	user	item	amount
20180801	Tom	Toaster	40
20180801	Polly	Toaster	60
20180802	Bob	Kettle	30
20180804	Wendy	Kettle	10

1

```
SELECT date,
   item,
   sum (amount) as amt
FROM base_table
GROUP BY 1.2
```

materialized_view

date	item	amt
20180801	Toaster	100
20180802	Kettle	30
20180804	Kettle	10

2

```
SELECT date,
   item
FROM base_table
Where user = “Policy”
```

3

```
SELECT date,
   item,
   sum (amount) as amount
FROM base_table
Where date =  “20180101”
```

1 MV is created and results are periodically refreshed.

2 By default, queries continue to go to the base table unaffected.

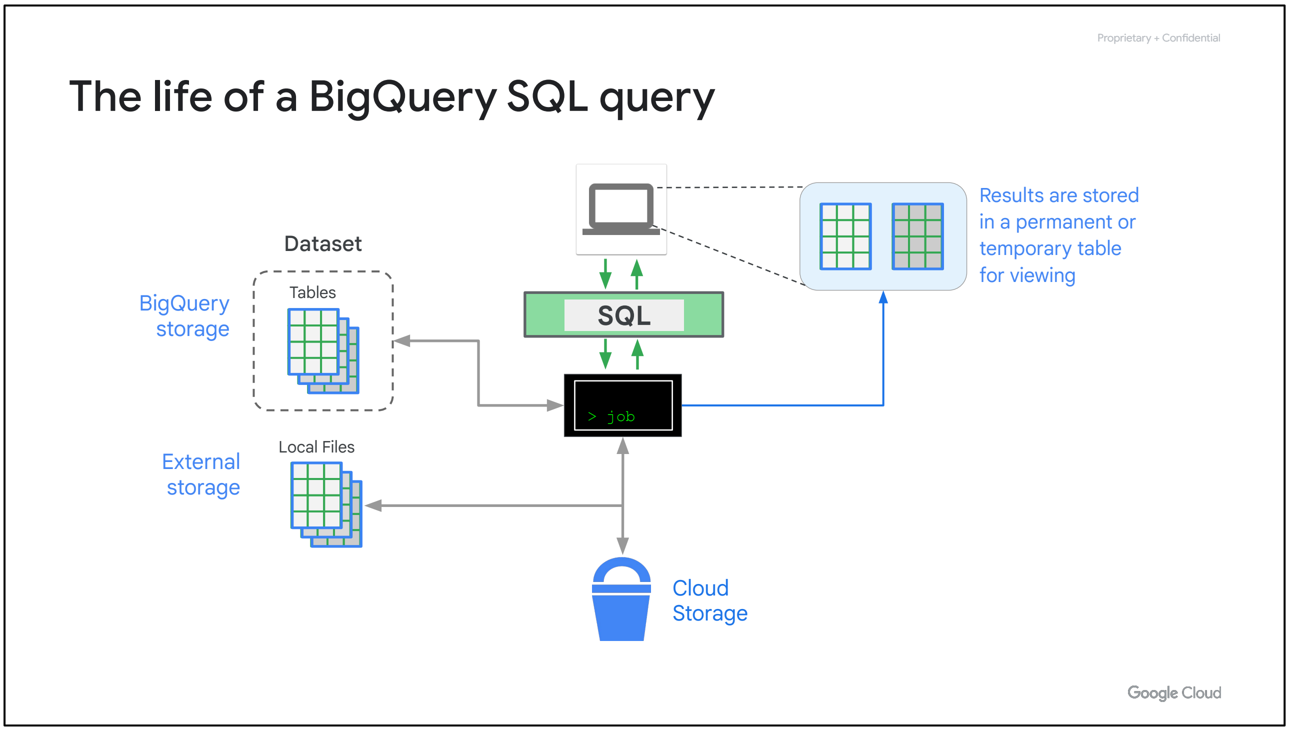
3 Applicable queries are rerouted automatically to MV to achieve faster execution.

Google Cloud

In BigQuery, materialized views periodically cache the results of a query for increased performance and efficiency. BigQuery leverages precomputed results from materialized views and whenever possible reads only delta changes from the base table to compute up-to-date results. Materialized views can be queried directly or can be used by the BigQuery optimizer to process queries to the base tables.

Queries that use materialized views are generally faster and consume fewer resources than queries that retrieve the same data only from the base table. Materialized views can significantly improve the performance of workloads that have the characteristic of common and repeated queries.

[Materialized views: https://cloud.google.com/bigquery/docs/materialized-views-intro]



The life of a BigQuery SQL query

In the queries we saw earlier, we wrote the query in SQL and selected Run on the UI. What this did was to submit a QueryJob to the BigQuery service.

The BigQuery query service is separate from the BigQuery storage service. However, they are designed to collaborate and be used together. In this case, we were querying native tables in the bigquery-public-data project. Querying native tables is the most common case, and is the most performant way to use BigQuery.

BigQuery is most efficient when working with data contained in its own storage service. The storage service and the query service work together to internally organize the data to make queries efficient over huge datasets of Terabytes and Petabytes in size.

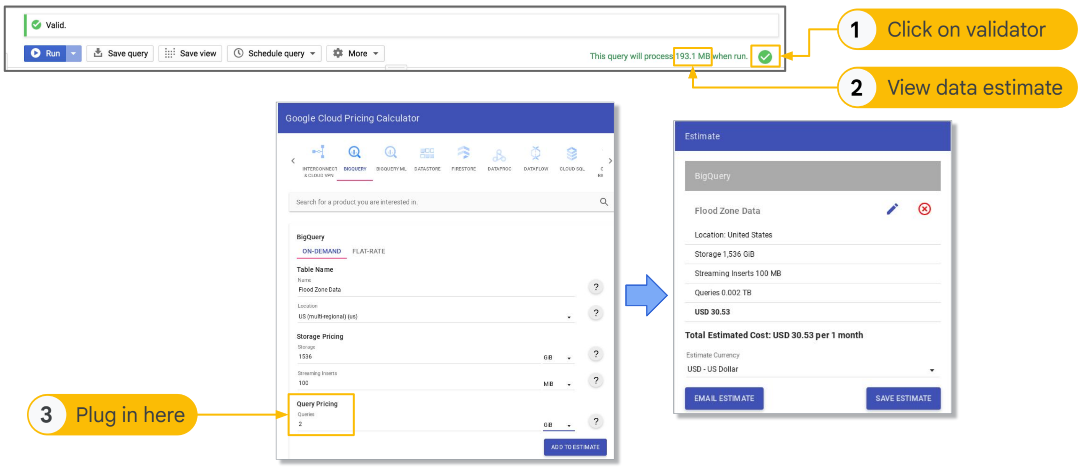
The query service can also run query jobs on data contained in other locations, such as tables in CSV files hosted in Cloud Storage.

So you can query data in external tables or from external sources without loading it into BigQuery. These are called federated queries.

In either case, the query service puts the results into a temporary table and the user interface pulls and displays the data in the temporary table. This temporary table is stored for 24 hours, so if you run the exact same query again, and if the results would not be different, then BigQuery will simply return a pointer to the cached results. Queries that can be served from the cache do not incur any charges.

It is also possible to request that the query job write to a destination table. In that case, you get to control when the table is deleted. Because the destination table is permanent, and not temporary, you will get charged for the storage of the results.

Use query validator with pricing calculator for estimates



Valid.

This query will process 193.1 MB when run.

BigQuery

Table Name
Name: Flood Zone Data
Location: US (multi-regional) (us)

Storage Pricing
Storage: 1536 GiB
Streaming Inserts: 100 MiB

Query Pricing
Queries: 2 GiB

BigQuery

Flood Zone Data
Location: United States
Storage 1,536 GiB
Streaming Inserts 100 MB
Queries 0.002 TB
USD 30.53

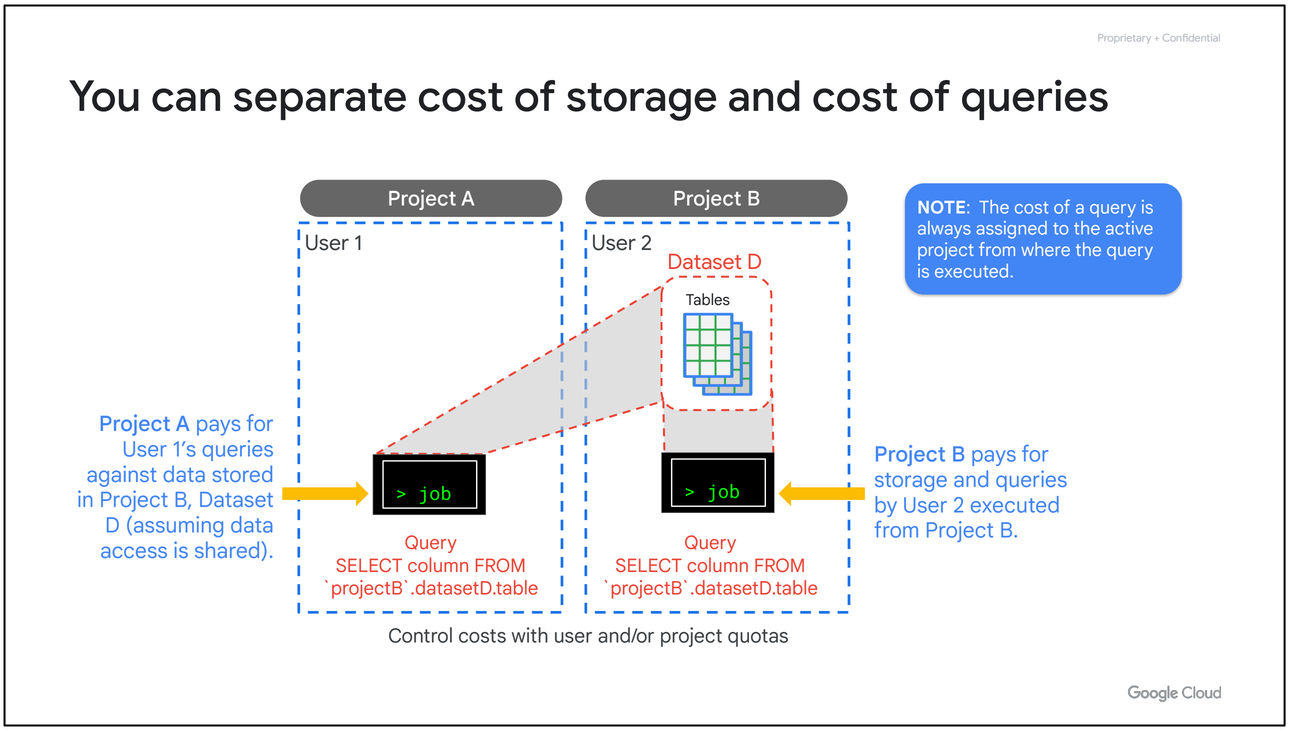
Total Estimated Cost: USD 30.53 per 1 month

Estimate Currency: USD - US Dollar

To calculate pricing, you can use BigQuery's query validator in combination with the pricing calculator for estimates.

The query validator provides an estimate of the size of data that will be processed during a query. You can plug this into the calculator to find an estimate of how much running the query will cost.

This is valid if you are using an on-demand plan where you pay for each query based on how much data is processed by that query. Your company might have opted for a flat-rate plan. In that case, your company will be paying a fixed price, and so the cost is really how many slots your query uses.



You can separate cost of storage and cost of queries

NOTE: The cost of a query is always assigned to the active project from where the query is executed.

Project A pays for User 1's queries against data stored in Project B, Dataset D (assuming data access is shared).

Project B pays for storage and queries by User 2 executed from Project B.

Query
SELECT column FROM
`projectB`.datasetD.table

Query
SELECT column FROM
`projectB`.datasetD.table

Control costs with user and/or project quotas

Google Cloud

You can separate cost of storage and cost of queries.

By separating projects A and B, it's possible to share data without giving access to run jobs. In this diagram, Users 1 and 2 have access to run jobs and access the datasets in their own respective Projects. If they run a query, that job is billed to their own project.

What if User 1 needs the ability to access Dataset D in Project B? The person who owns Project B can allow User 1 to query Project B Dataset D and the charges will go to Project A when executed from Project A.

The public dataset project owner granted all authenticated users access to use their data. The special setting allAuthenticatedUsers makes a dataset public. Authenticated users must use BigQuery within their own project and have access to run BigQuery jobs so that they can query the Public Dataset. The billing for the query goes to their project, even though the query is using public or shared data.

In summary, the cost of a query is always assigned to the active project from where the query is executed. The active project for a user is displayed at the top of the Cloud Console or set by an environmental variable in the Cloud Shell or client tools.

NOTE: BigQuery offers 1 TB of querying for free every month, so public datasets are an easy way to try out BigQuery.