GCP Certification Series: 4.4 Managing data solutions

Prashanta Paudel
Nov 12, 2018 · 25 min read

This blog discusses managing data solutions including many GCP products and solutions.

As we have already discussed basic stuff in previous blogs we will dig directly into course content.

=====================================
===

Executing queries to retrieve data from data instances (e.g., Cloud SQL, BigQuery, Cloud Spanner, Cloud Datastore, Cloud Bigtable, Cloud Dataproc)

Queries help display the data into the format of the instance. Traditionally we use queries in the database to show the data satisfying the fields we mentioned in the query.

let's go to different data solution's queries.

CLOUD SQL

After the database is up and running you have two options to query for data

1. from console
2. from Cloud Shell
3. Using cloud SDK or third party apps.

we will see console and Shell as these covers all types of method

> *From Console*

We can only perform these kinds of actions in Cloud SQL.

- Create a database
- Export/import a database
- duplicate/delete a database

Querying is done from Cloud Shell, VM or other tools.

> *From Cloud Shell*

First, connect to the instance from shell

```
$ gcloud sql connect mydata --user=root --quiet
Whitelisting your IP for incoming connection for 5
minutes...done.
Connecting to database with SQL user [root].Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 263
Server version: 5.7.14-google-log (Google)

Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and
others.

Type 'help;' or '\h' for help. Type '\c' to clear the
current input statement.

MySQL [(none)]>
```

Then we can do various operations in the database in MySQL commands.

Viewing instance summary information

You can view summary information about your Cloud SQL instances

```
gcloud sql instances describe [INSTANCE_NAME]
```

Create the instance

```
gcloud sql instances create [INSTANCE_NAME] --tier=
[MACHINE_TYPE] --region=[REGION]


$ gcloud sql instances create pop1011 --tier=db-n1-standard-
1 --region=us-central1
Creating Cloud SQL instance...done.
Created
[https://www.googleapis.com/sql/v1beta4/projects/fourpointfo
ur-222308/instances/pop1011].
NAME     DATABASE_VERSION  LOCATION       TIER
PRIMARY_ADDRESS  PRIVATE_ADDRESS  STATUS
pop1011  MYSQL_5_7         us-central1-a  db-n1-standard-1
35.225.244.102   -                RUNNABLE
```

Do not include sensitive or personally identifiable information in your instance name; it is externally visible.
You do not need to include the project ID in the instance name. This is done automatically where appropriate (for example, in the log files).

MACHINE_TYPE is one of the values from the previous step that starts with `db-` .

For example, the following command creates a Second Generation instance called instance1 with the machine type of `db-n1-standard-2` in the London region:

```
gcloud sql instances create instance1 --tier=db-n1-standard-
2 --region=europe-west2
```

Clone Instance

```
gcloud sql instances clone [SOURCE_INSTANCE_NAME]
[TARGET_INSTANCE_NAME]


$ gcloud sql instances clone pop1011 mypop
Cloning Cloud SQL instance...done.
Created
[https://www.googleapis.com/sql/v1beta4/projects/fourpointfo
ur-222308/instances/mypop].
NAME   DATABASE_VERSION  LOCATION       TIER
PRIMARY_ADDRESS  PRIVATE_ADDRESS  STATUS
```

```
mypop  MYSQL_5_7         us-central1-a  db-n1-standard-1
35.232.220.175   -                RUNNABLE
```

Deleting Instance

```
gcloud sql instances delete [INSTANCE_NAME]


$ gcloud sql instances delete mypop
All of the instance data will be lost when the instance is
deleted.

Do you want to continue (Y/n)?  y

Deleting Cloud SQL instance...done.
Deleted
[https://www.googleapis.com/sql/v1beta4/projects/fourpointfo
ur-222308/instances/mypop].
```

QUERIES in MySQL

```
$ gcloud sql connect mydata --user=root --quiet
Whitelisting your IP for incoming connection for 5
minutes...done.
Connecting to database with SQL user [root].Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 263
Server version: 5.7.14-google-log (Google)

Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and
others.

Type 'help;' or '\h' for help. Type '\c' to clear the
current input statement.

MySQL [(none)]>
OR from VM
#mysql


SHOW DATABASES                             -- Show all the
databases in this server
MySQL [(none)]> SHOW DATABASES
    -> ;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| classicmodels      |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.04 sec)


USE databaseName                           -- Set the
default (current) database


MySQL [(none)]> use classicmodels
Reading table information for completion of table and column
names
You can turn off this feature to get a quicker startup with
-A

Database changed


-- Database-Level
DROP DATABASE databaseName                 -- Delete the
database (irrecoverable!)
DROP DATABASE IF EXISTS databaseName       -- Delete if it
exists
CREATE DATABASE databaseName               -- Create a new
database
CREATE DATABASE IF NOT EXISTS databaseName -- Create only if
it does not exists

SELECT DATABASE()                          -- Show the
default database


MySQL [classicmodels]> select database()
    -> ;
+---------------+
| database()    |
+---------------+
| classicmodels |
+---------------+
1 row in set (0.04 sec)


SHOW CREATE DATABASE databaseName          -- Show the
CREATE DATABASE statement

-- Table-Level
DROP TABLE [IF EXISTS] tableName, ...
CREATE TABLE [IF NOT EXISTS] tableName (
   columnName columnType columnAttribute, ...
   PRIMARY KEY(columnName),
   FOREIGN KEY (columnNmae) REFERENCES tableName
(columnNmae)
)
SHOW TABLES                -- Show all the tables in the
default database
DESCRIBE|DESC tableName    -- Describe the details for a
table
ALTER TABLE tableName ...  -- Modify a table, e.g., ADD
COLUMN and DROP COLUMN
ALTER TABLE tableName ADD columnDefinition
ALTER TABLE tableName DROP columnName
ALTER TABLE tableName ADD FOREIGN KEY (columnNmae)
REFERENCES tableName (columnNmae)
ALTER TABLE tableName DROP FOREIGN KEY constraintName
SHOW CREATE TABLE tableName        -- Show the CREATE TABLE
statement for this tableName

-- Row-Level
INSERT INTO tableName
   VALUES (column1Value, column2Value,...)               --
Insert on all Columns
INSERT INTO tableName
   VALUES (column1Value, column2Value,...), ...          --
Insert multiple rows
INSERT INTO tableName (column1Name, ..., columnNName)
   VALUES (column1Value, ..., columnNValue)              --
Insert on selected Columns
DELETE FROM tableName WHERE criteria
UPDATE tableName SET columnName = expr, ... WHERE criteria
SELECT * | column1Name AS alias1, ..., columnNName AS aliasN
   FROM tableName
   WHERE criteria
   GROUP BY columnName
   ORDER BY columnName ASC|DESC, ...
   HAVING groupConstraints
   LIMIT count | offset count
 MySQL [classicmodels]> select * from customers where
postalcode=41101;
+----------------+--------------------------+------------
----+-----------------+----------------+--------------+---
-----------+---------+-------+-----------+---------+------
-----------------+------------+
| customerNumber | customerName               |
contactLastName | contactFirstName | phone          |
addressLine1  | addressLine2 | city    | state | postalCode
| country | salesRepEmployeeNumber | creditLimit |
+----------------+--------------------------+------------
----+-----------------+----------------+--------------+---
-----------+---------+-------+-----------+---------+------
-----------------+------------+
|            484 | Iberia Gift Imports, Corp. | Roel
| José Pedro       | (95) 555 82 82 | C/ Romero, 33 | NULL
| Sevilla | NULL  | 41101      | Spain   |
1702 |    65700.00 |
+----------------+--------------------------+------------
----+-----------------+----------------+--------------+---
-----------+---------+-------+-----------+---------+------
-----------------+------------+
1 row in set, 20 warnings (0.04 sec)
-- Others
SHOW WARNINGS;   -- Show the warnings of the previous
statement
```

=====================================
===

BigQuery

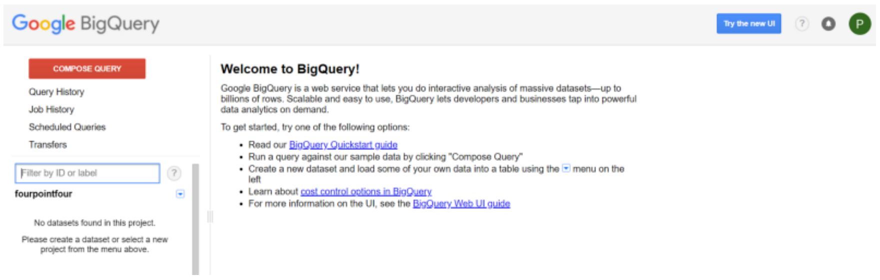
Bigquery is the platform where we can ***import huge database in terrabytes and run a query on it.***

It is basically a query running platform rather than storing or editing databases.

We can run a query in public database directly from the platform as well as by importing database to BigQuery and then running the query.

The big difference will be on the size of the database that needs to be stored in cloud storage.

First, we need to set up a database.



No datasets found in this project.

Please create a dataset or select a new project from the menu above.

Welcome to BigQuery!

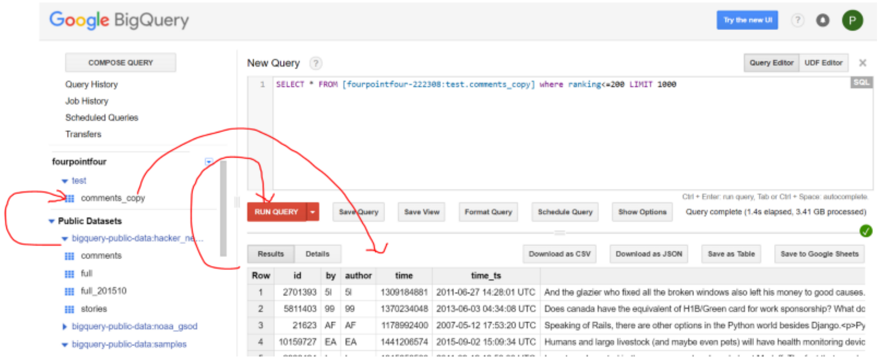
Google BigQuery is a web service that lets you do interactive analysis of massive datasets—up to billions of rows. Scalable and easy to use, BigQuery lets developers and businesses tap into powerful data analytics on demand.

To get started, try one of the following options:

- Read our BigQuery Quickstart guide
- Run a query against our sample data by clicking "Compose Query"
- Create a new dataset and load some of your own data into a table using the menu on the left
- Learn about cost control options in BigQuery
- For more information on the UI, see the BigQuery Web UI guide

BigQuery Dashboard

Three steps to run a query by importing data



```
SELECT * FROM [fourpointfour-222308:test.comments_copy] where ranking<=200 LIMIT 1000
```

Ctrl + Enter: run query, Tab or Ctrl + Space: autocomplete

Query complete (1.4s elapsed, 3.41 GB processed)

Row	id	by	author	time	time_ts	
1	2701393	5l	5l	1309184881	2011-06-27 14:28:01 UTC	And the glazier who fixed all the broken windows also left his money to good causes.
2	5811403	99	99	1370234048	2013-06-03 04:34:08 UTC	Does canada have the equivalent of H1B/Green card for work sponsorship? What dc
3	21623	AF	AF	1178992400	2007-05-12 17:53:20 UTC	Speaking of Rails, there are other options in the Python world besides Django.<p>Py
4	10159727	EA	EA	1441206574	2015-09-02 15:09:34 UTC	Humans and large livestock (and maybe even pets) will have health monitoring devic

three steps for public data query

The main target is to create a query job and retrieve the result through API or export result to Data Studio for further analysis and representation.

> *In Shell*

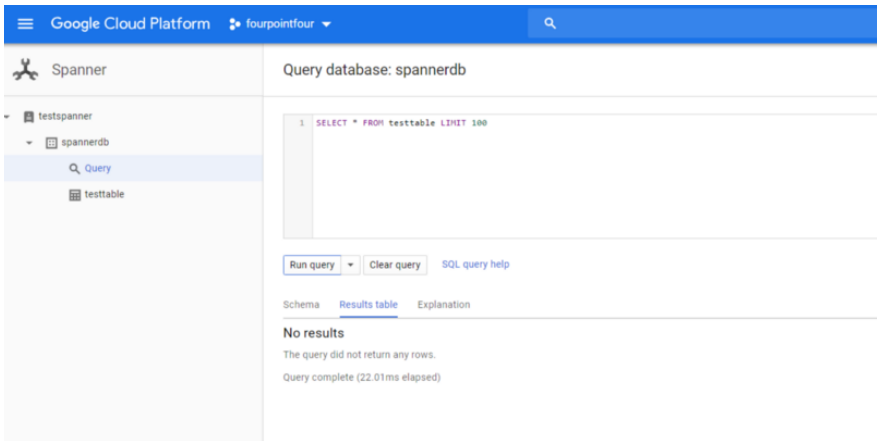
```
prashantapaudel_7@cloudshell:~ (fourpointfour-222308)$ bq
shell
Welcome to BigQuery! (Type help for more information.)
fourpointfour-222308>


fourpointfour-222308> SELECT * FROM [bigquery-public-
data.samples.shakespeare]
Waiting on bqjob_r517d95cd678a9617_00000167078ee625_1 ...
(1s) Current status: DONE
+--------------+------------+---------+-------------+
|     word     | word_count | corpus  | corpus_date |
+--------------+------------+---------+-------------+
| LVII         |          1 | sonnets |           0 |
| augurs       |          1 | sonnets |           0 |


| wailing      |          1 | sonnets |           0 |
| whereupon    |          1 | sonnets |           0 |
| clears       |          1 | sonnets |           0 |
| moiety       |          1 | sonnets |           0 |
| oaths'       |          1 | sonnets |           0 |
| alien        |          1 | sonnets |           0 |
| warning      |          1 | sonnets |           0 |
| Weeds        |          1 | sonnets |           0 |
| duteous      |          1 | sonnets |           0 |
| April's      |          1 | sonnets |           0 |
| season'd     |          1 | sonnets |           0 |
| exceeds      |          1 | sonnets |           0 |
| miss'd       |          1 | sonnets |           0 |
+--------------+------------+---------+-------------+
fourpointfour-222308>
```

=======================================
===

Cloud Spanner

Cloud spanner quer is similar to BigQuery and SQL as all of these follow SQL format.

In cloud spanner, we have to create a relational database or create a pipeline where databases are created via Dataflow instance. then we can query on the data as similar as earlier



Query database: spannerdb

```
SELECT * FROM testtable LIMIT 100
```

No results

The query did not return any rows.

Query complete (22.01ms elapsed)

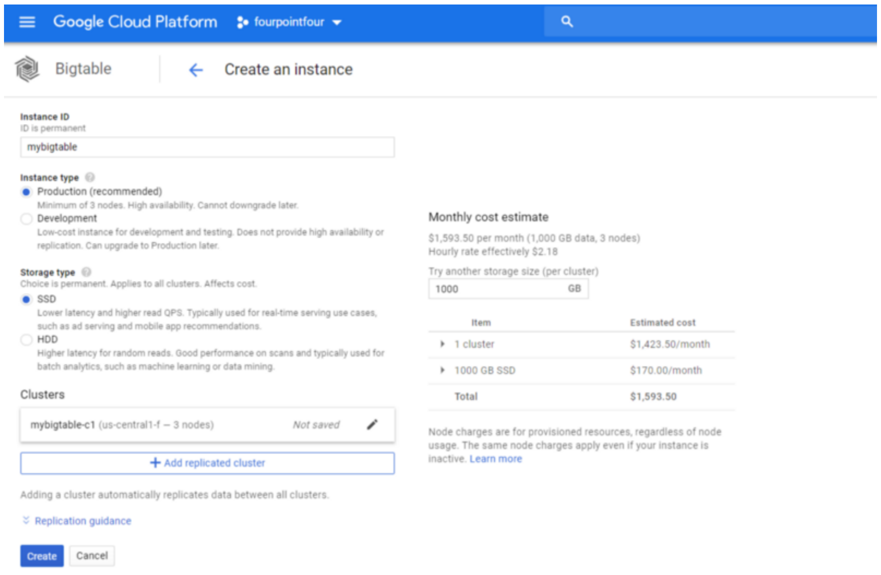
Cloud Spanner

=====================================

===

Cloud Bigtable

Cloud Bigtable is a NoSQL database management system. It stores data in large tabular format.



Instance ID
ID is permanent
mybigtable

Instance type
[✓ Production (recommended)]
Minimum of 3 nodes. High availability. Cannot downgrade later.
Development
Low-cost instance for development and testing. Does not provide high availability or replication. Can upgrade to Production later.

Storage type
Choice is permanent. Applies to all clusters. Affects cost.
[✓ SSD]
Lower latency and higher read QPS. Typically used for real-time serving use cases, such as ad serving and mobile app recommendations.
HDD
Higher latency for random reads. Good performance on scans and typically used for batch analytics, such as machine learning or data mining.

Clusters
mybigtable-c1 (us-central1-f – 3 nodes) *Not saved*

Adding a cluster automatically replicates data between all clusters.

Monthly cost estimate
$1,593.50 per month (1,000 GB data, 3 nodes)
Hourly rate effectively $2.18
Try another storage size (per cluster)
1000 GB

Item	Estimated cost
1 cluster	$1,423.50/month
1000 GB SSD	$170.00/month
Total	$1,593.50

Node charges are for provisioned resources, regardless of node usage. The same node charges apply even if your instance is inactive. Learn more

Start by creating an instance with a single cluster. Use the `bigtable instances create` command to create an instance:

```
gcloud bigtable instances create INSTANCE_ID \
    --cluster=CLUSTER_ID \
    --cluster-zone=CLUSTER_ZONE \
    --display-name=DISPLAY_NAME \
    [--cluster-num-nodes=CLUSTER_NUM_NODES] \
    [--cluster-storage-type=CLUSTER_STORAGE_TYPE] \
    [--instance-type=INSTANCE_TYPE]


$ gcloud bigtable instances create abcxyz --
cluster=hell12345 --cluster-zone=us-central1-f --display-
name=yesplease
Creating bigtable instance abcxyz...done.
```

Provide the following values:

- `INSTANCE_ID` : The permanent identifier for the instance.
- `CLUSTER_ID` : The permanent identifier for the cluster.
- `CLUSTER_ZONE` : The zone where the cluster runs.
- If you plan to use replication, make sure Cloud Bigtable is available in at least one other zone in your preferred region. View the zone list.
- `DISPLAY_NAME` : A human-readable name that identifies the instance in the GCP Console.

The command accepts the following optional flags:

- `--cluster-num-nodes=CLUSTER_NUM_NODES` : The number of nodes in the cluster. Clusters in a production instance must have 3 or more nodes. The default value is `3` . If you aren't sure how many nodes you need, use the default. You can add more nodes later. Learn more.
- Do not use this flag for development instances.
- `--cluster-storage-type=CLUSTER_STORAGE_TYPE` : The type of storage to use for the cluster. Each cluster in an instance must use the same storage type. Accepts the values `SSD` and `HDD` . The default value is `SSD` .
- In most cases, the default value is best. This choice is permanent. Learn more.
- `--instance-type=INSTANCE_TYPE` : The type of instance to create. Accepts one of the following values:

- `PRODUCTION` (default): A high-availability, full-powered instance. This choice is permanent. Learn more.
- `DEVELOPMENT` : A low-cost instance for development and testing, with limited performance and no SLA. You can upgrade to a production instance later. Learn more.

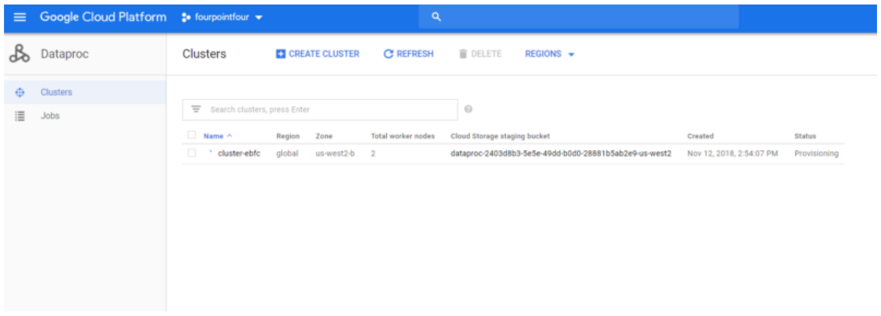
====================================
===

Cloud Dataproc

To use Dataproc you have to enable the API first.

Google Cloud Dataproc lets you provision Apache Hadoop clusters and connects to underlying analytic data stores

Create a Cluster



Name	Region	Zone	Total worker nodes	Cloud Storage staging bucket	Created	Status
cluster-ebfc	global	us-west2-b	2	dataproc-2403d8b3-5e5e-49dd-b0d0-28881b5ab2e9-us-west2	Nov 12, 2018, 2:54:07 PM	Provisioning

dataproc cluster

After creating cluster now add a job to the cluster.

====================================
==

Estimating the cost of Bigquery Query

On-demand queries are charged based on the number of bytes read. For current on-demand query pricing, see the pricing page.

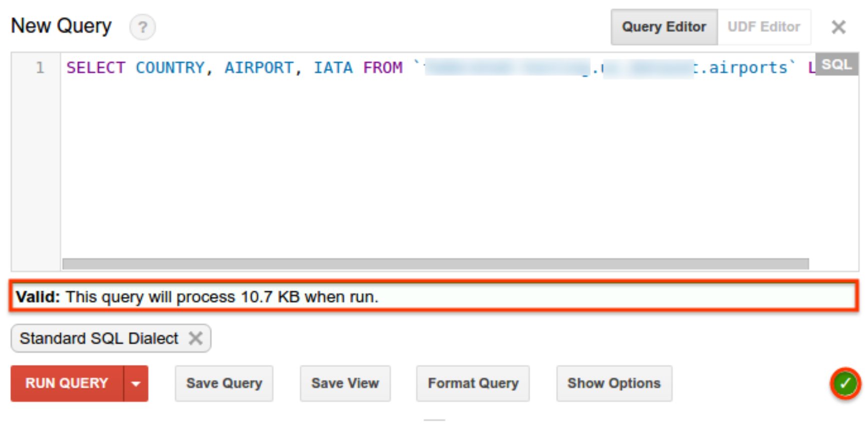
To estimate costs before running a query using the:

- Query validator in the web UI

- `--dry_run` flag in the CLI
- `dryRun` parameter when submitting a query job using the API
- Google Cloud Platform Pricing Calculator

Estimating query costs

When you enter a query in the web UI, the query validator verifies the query syntax and provides an estimate of the number of bytes read. You can use this estimate to calculate query cost in the pricing calculator.



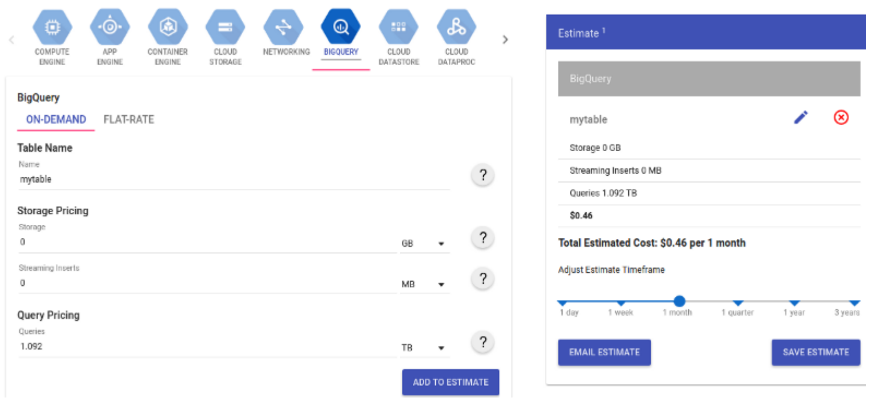
```
SELECT COUNTRY, AIRPORT, IATA FROM `[redacted].[redacted].airports` L
```

Valid: This query will process 10.7 KB when run.

If your query processes a small amount of data, you might need to convert the bytes that are processed from KB to MB. MB is the smallest measure that is used by the pricing calculator.

Estimating query costs using the Google Cloud Platform Pricing Calculator

To estimate on-demand query costs in the Google Cloud Platform Pricing Calculator, enter the number of bytes that are processed by the query as MB, GB, TB, or PB. If your query processes less than 1 TB, the estimate is $0 because BigQuery provides 1 TB of on-demand query processing free per month.

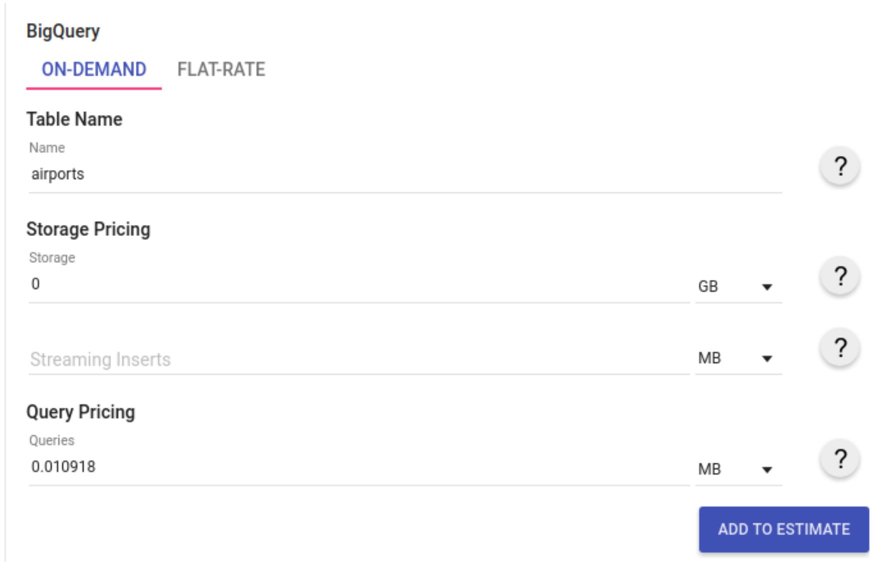


BigQuery
ON-DEMAND FLAT-RATE
Table Name
Name
mytable
Storage Pricing
Storage
0 GB
Streaming Inserts
0 MB
Query Pricing
Queries
1.092 TB
ADD TO ESTIMATE

Estimate [1]
BigQuery
mytable
Storage 0 GB
Streaming Inserts 0 MB
Queries 1.092 TB
$0.46
Total Estimated Cost: $0.46 per 1 month
Adjust Estimate Timeframe

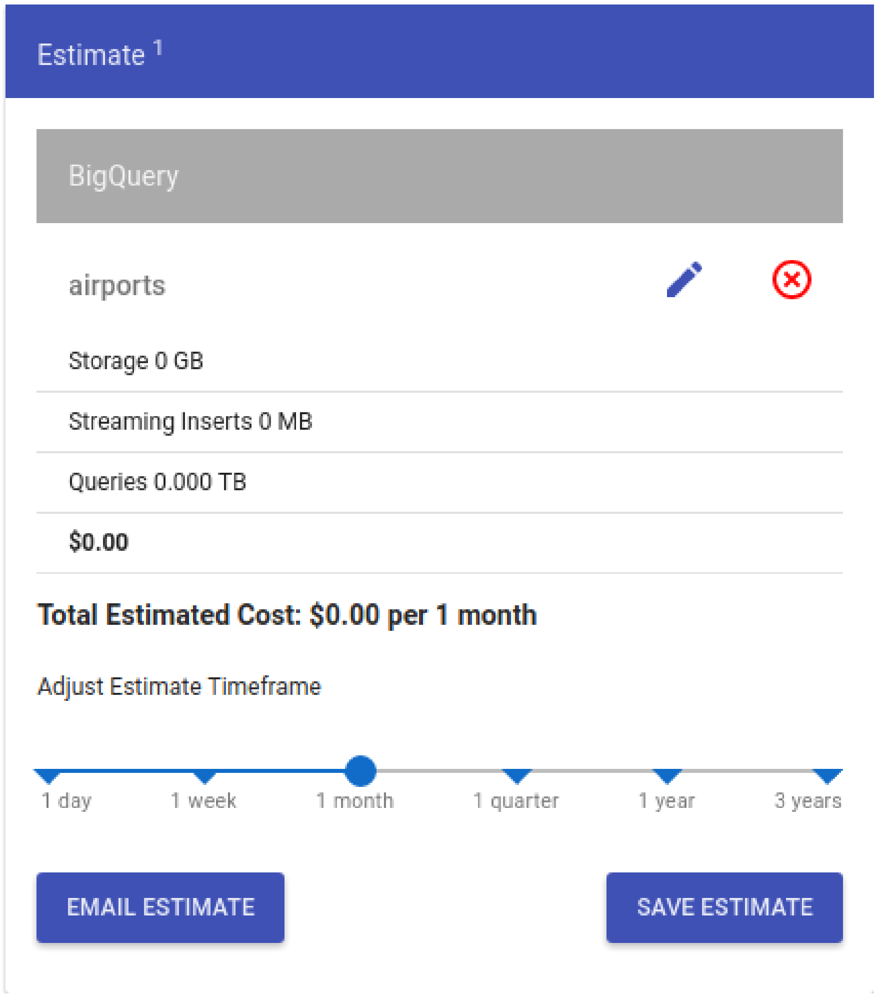
To estimate the cost of a query using the pricing calculator:

1. Open the Google Cloud Platform Pricing Calculator,
2. Click BigQuery.
3. Click the **On-Demand** tab.
4. For **Table Name**, type the name of the table. For example, `airports` .
5. For **Storage Pricing**, enter `0` in the **Storage** field.
6. For **Query Pricing**, enter the estimated bytes read from your dry run or the query validator. If the value is less than 1 MB, you must convert it to MB for the pricing calculator. Using 10918 bytes as an example, the value is approximately 0.01091 MB.



BigQuery
ON-DEMAND FLAT-RATE
Table Name
Name
airports
Storage Pricing
Storage
0 GB
Streaming Inserts MB
Query Pricing
Queries
0.010918 MB
ADD TO ESTIMATE

1. Click **Add To Estimate**.
2. The estimate appears to the right. Notice that you can save or email the estimate.

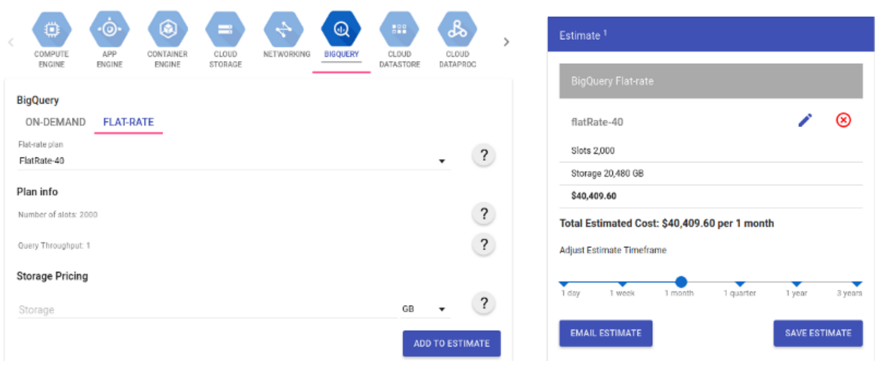


Storage 0 GB
Streaming Inserts 0 MB
Queries 0.000 TB
$0.00
Total Estimated Cost: $0.00 per 1 month
Adjust Estimate Timeframe

In this case, the number of bytes read by the query is below the 1 TB of on-demand processing provided via the free tier. As a result, the estimated cost is $0.

Including flat-rate pricing in the pricing calculator

If you have flat-rate pricing applied to your billing account, you can click the **Flat-Rate** tab, choose your flat-rate plan, and add your storage costs to the estimate.



BigQuery

Flat-rate plan: FlatRate-40

Plan info

Number of slots: 2000

Query Throughput: 1

Storage Pricing

Storage GB

Estimate 1

BigQuery Flat-rate

flatRate-40

Slots 2,000

Storage 20,480 GB

$40,409.60

Total Estimated Cost: $40,409.60 per 1 month

Adjust Estimate Timeframe

1 day · 1 week · 1 month · 1 quarter · 1 year · 3 years

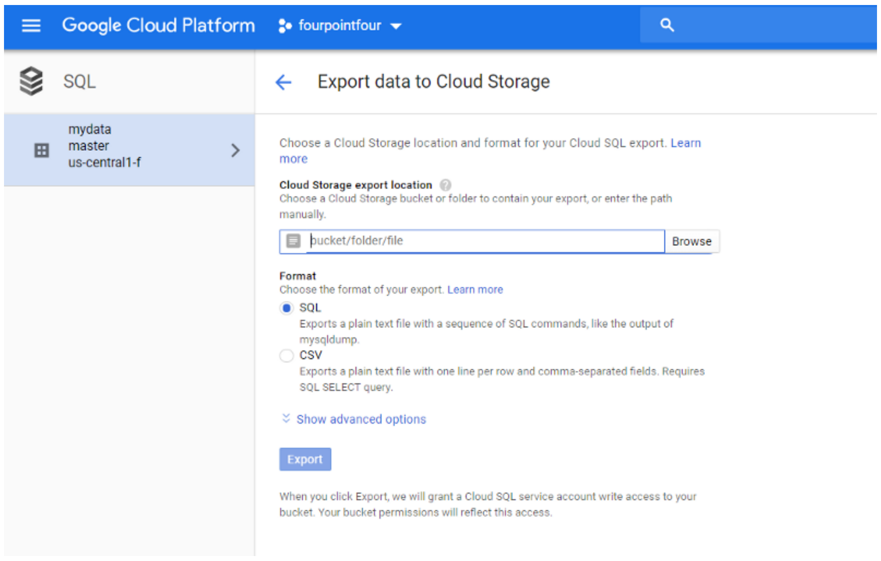
For more information, see Flat-rate pricing.

==

Backing up and restoring data instances (e.g., Cloud SQL, Cloud Datastore, Cloud Dataproc)

Cloud SQL

you can backup cloud SQL database in Bucket.



mydata
master
us-central1-f

Export data to Cloud Storage

Choose a Cloud Storage location and format for your Cloud SQL export. Learn more

Cloud Storage export location
Choose a Cloud Storage bucket or folder to contain your export, or enter the path manually.

bucket/folder/file

Format
Choose the format of your export. Learn more

(●) SQL
Exports a plain text file with a sequence of SQL commands, like the output of mysqldump.

( ) CSV
Exports a plain text file with one line per row and comma-separated fields. Requires SQL SELECT query.

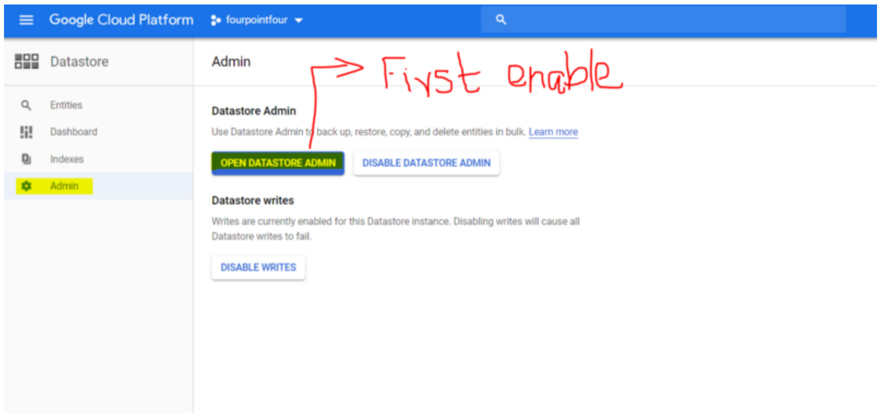
When you click Export, we will grant a Cloud SQL service account write access to your bucket. Your bucket permissions will reflect this access.

backup

and Restore using import from the same location or by manulally managing the database bac=kup.

———

Cloud Datastore



Admin

Datastore Admin

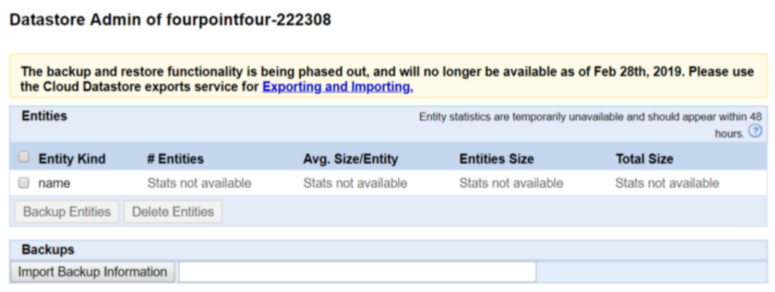
Use Datastore Admin to back up, restore, copy, and delete entities in bulk. Learn more

Datastore writes

Writes are currently enabled for this Datastore instance. Disabling writes will cause all Datastore writes to fail.

enble admin

then you can see admin panel to do backup and restore



Datastore Admin of fourpointfour-222308

The backup and restore functionality is being phased out, and will no longer be available as of Feb 28th, 2019. Please use the Cloud Datastore exports service for Exporting and Importing.

Entities

Entity statistics are temporarily unavailable and should appear within 48 hours.

Entity Kind	# Entities	Avg. Size/Entity	Entities Size	Total Size
name	Stats not available	Stats not available	Stats not available	Stats not available

=====================================
=

Reviewing job status in BigQuery

Managing BigQuery Jobs

After you submit a BigQuery job, you can view job data, list jobs, cancel a job, or rerun a job.

When a job is submitted, it can be in one of three states:

- `PENDING` —scheduled
- `RUNNING`
- `DONE` —reported as `SUCCESS` or `FAILURE` (if the job completed with errors)

Viewing job data

You can view job data and metadata by using the web UI, CLI, and API. This data includes details such as the job type, the job state, and the user who ran the job.

Required permissions

In order to get job data and metadata, you must have `bigquery.jobs.get` permissions. The following project-level, predefined IAM role includes `bigquery.jobs.get` permissions:

- `bigquery.admin`

If you grant an account the `bigquery.admin` role, the user can view all job data in the project no matter who submitted the job.

The following roles are granted `bigquery.jobs.get` permissions for self-created jobs. These users can only view job data for jobs they submit:

- `bigquery.user`
- `bigquery.jobUser`

For more information on IAM roles and permissions in BigQuery, see access control.

Viewing information about jobs

To view information about a job:

1. In the navigation pane, click **Job History**.

2. In the **Recent Jobs** section, click the job to view the details.

Listing jobs in a project

Your project maintains your job history for all jobs created in the past six months. To request automatic deletion of jobs that are more than 50 days old, contact support.

You can view your BigQuery job history via the Google Cloud Platform Console, the CLI, or the API. This history includes jobs that are in the `RUNNING` state and jobs that are `DONE` (indicated by reporting the state as `SUCCESS` or `FAILURE`).

Required permissions

In order to list jobs, you must have `bigquery.jobs.list` permissions. The following project-level, predefined IAM roles include `bigquery.jobs.list` permissions:

- `bigquery.user`
- `bigquery.admin`

When you are granted `bigquery.jobs.list` permissions, you can list all jobs in a project, but the details and metadata are redacted for jobs submitted by other users. `bigquery.jobs.list` permissions allow you to see full details for self-created jobs.

To list all jobs, including details for jobs created by other users, you must have `bigquery.jobs.listAll` permissions. Only the `bigquery.admin` role has `bigquery.jobs.listAll` permissions.

The following role is granted `bigquery.jobs.list` permissions only for self-created jobs. These users can only list jobs they submit:

- `bigquery.jobUser`

For more information on IAM roles and permissions in BigQuery, see access control.

Listing jobs

When you list jobs in a project, you do not need to provide a location. Currently, jobs are listed for all locations.

To list jobs in a project:

1. In the navigation pane, click **Job History**.
2. In the **Recent Jobs** section, your jobs are listed by creation time with the most recent jobs at the top. The list includes jobs only for the current user. To see all jobs, use the command-line tool or the API.

Canceling jobs

You can cancel a `RUNNING` or `PENDING` job in the web UI, CLI, or API. However, not all job types can be canceled. If the job cannot be canceled, an error is returned.

Even if the job can be canceled, success is not guaranteed. The job might have completed by the time the cancel request is submitted, or the job might be in a stage where it cannot be canceled.

Note: You can still incur costs after canceling a job depending on the stage at which it was canceled. For more information, see Pricing.

Required permissions

In order to cancel a job, you must have `bigquery.jobs.update` permissions. The following project-level, predefined IAM role includes `bigquery.jobs.update` permissions:

- `bigquery.admin`

If you grant an account the `bigquery.admin` role, the user can cancel any eligible job, no matter who submitted it.

The following roles can cancel self-created jobs. These users can only cancel jobs they submit:

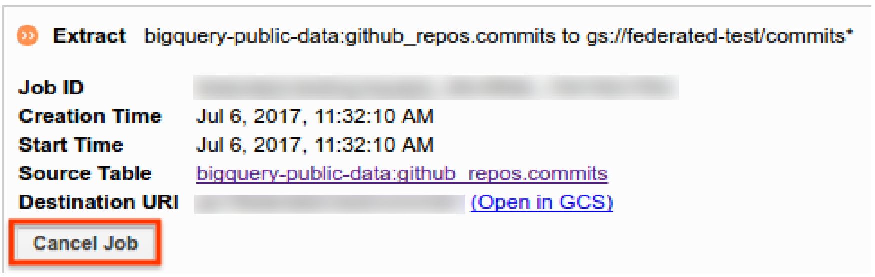
- `bigquery.user`
- `bigquery.jobUser`

For more information on IAM roles and permissions in BigQuery, see access control.

Canceling a job

WEB UI

1. In the navigation pane, click **Job History**.
2. **Note:** You cannot cancel a query job by using the web UI.
3. In the **Recent Jobs** section, click the job you're canceling. The most recent jobs appear at the top of the list.
4. In the job details, click **Cancel Job**.



Extract bigquery-public-data:github_repos.commits to gs://federated-test/commits*

Job ID	[redacted]
Creation Time	Jul 6, 2017, 11:32:10 AM
Start Time	Jul 6, 2017, 11:32:10 AM
Source Table	bigquery-public-data:github_repos.commits
Destination URI	[redacted] (Open in GCS)

Repeating a job

It is not possible to rerun a job using the same job ID. Instead, you create a new job with the same configuration. When you submit the new job in the web UI or CLI, a new job ID is assigned. When you submit the job using the API or client libraries, you must generate a new job ID.

Required permissions

In order to run a job, you must have `bigquery.jobs.create` permissions. The following project-level, predefined IAM roles include `bigquery.jobs.create` permissions:

- `bigquery.user`
- `bigquery.jobUser`

- `bigquery.admin`

For more information on IAM roles and permissions in BigQuery, see access control.

Rerunning a job

To repeat a query job:

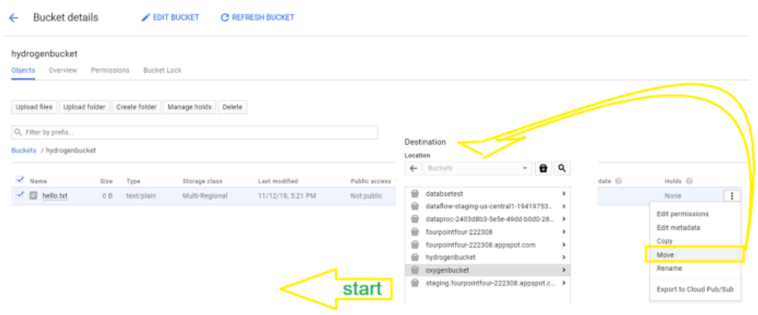
1. In the navigation pane, click **Query History**.
2. In the **Queries** section, to the right of the query, click **Open Query**.
3. Click **Run Query**.

To repeat a load job:

1. In the navigation pane, click **Job History**.
2. In the **Recent Jobs** section, click the job you want to repeat. The most recent jobs appear at the top of the list.
3. In the job details, click **Repeat Load Job**.
4. **Note:** You cannot repeat an export job or a copy job using the web UI.

======================================
===

Moving objects between Cloud Storage buckets



Name	Size	Type	Storage class	Last modified	Public access	date	Holds
hello.txt	0 B	text/plain	Multi-Regional	11/12/18, 5:21 PM	Not public		None

moving objects in buckets

> *In Cloud Shell*

#gsutil mv -p gs://hydrogenbucket/hello.txt gs://oxygenbucket

————————————————————————————————————-

Changing Object Storage Classes

This page describes how to change the storage class of objects within a bucket through overwriting the object. To learn how to change object storage classes without overwriting an object, see theObject Lifecycle Management feature. To learn more about per-object storage classes, see Per-Object Storage Classes.

Use the `-s` flag in a `rewrite` command. For example:

```
gsutil rewrite -s [STORAGE_CLASS] gs://[PATH_TO_OBJECT]
```

Where `[STORAGE_CLASS]` is the new storage class for your object and `[PATH_TO_OBJECT]` is the name of the object whose class you're changing.

Note: This guide involves overwriting data. Nearline Storage or Coldline Storage objects have early deletion charges if they are overwritten less than 30 or 90 days from their creation time, respectively. Also note that if you have enabled Object Versioning for your bucket, the original object remains in your bucket until it is explicitly deleted.

Changing the Default Storage Class of a Bucket

This page shows you how to change the *default storage class* for your buckets. When you upload an object to the bucket, if you don't specify a storage class for the object, the object is assigned the bucket's default storage class. For an overview of buckets, read the Key Terms. To learn more about storage classes, see Storage Classes.

To change the default storage class of an existing bucket:

Use the `gsutil defstorageclass set` command with the desired storage class, replacing `[VALUES_IN_BRACKETS]` with the appropriate values:

```
gsutil defstorageclass set [STORAGE_CLASS]
gs://[BUCKET_NAME]
```

The response looks like the following example:

```
Setting default storage class to "[STORAGE_CLASS]" for
bucket gs://[BUCKET_NAME]
```

==
===

Object Lifecycle Management

To support common use cases like setting a Time to Live (TTL) for objects, archiving older versions of objects, or "downgrading" storage classes of objects to help manage costs, Cloud Storage offers the Object Lifecycle Management feature. This page describes the feature as well as the options available when using it. To learn how to enable Object Lifecycle Management, and for examples of lifecycle policies, see Managing Lifecycles.

Introduction

You can assign a lifecycle management configuration to a bucket. The configuration contains a set of rules which apply to current and future objects in the bucket. When an object meets the criteria of one of the rules, Cloud Storage automatically performs a specified action on the object. Here are some example use cases:

- Downgrade the storage class of objects older than 365 days to Coldline Storage.

- Delete objects created before January 1, 2013.
- Keep only the 3 most recent versions of each object in a bucket with versioning enabled.

Lifecycle configuration

Each lifecycle management configuration contains a set of rules. When defining a rule, you can specify any set of conditions for any action. If you specify multiple conditions in a rule, an object has to match *all* of the conditions for the action to be taken. If you specify multiple rules that contain the same action, the action is taken when an object matches the condition(s) in *any* of the rules. Each rule should contain only one action.

If a single object is subject to multiple actions, Cloud Storage performs only one of the actions, and the object will be re-evaluated before any additional actions are taken. A Delete action takes precedence over a SetStorageClass action. If multiple SetStorageClass actions are specified, the action switching to the storage class with the lowest at-rest storage pricing is chosen.

So, for example, if you have one rule that deletes an object and another rule that changes the object's storage class, but both rules use the exact same condition, the delete action always occurs when the condition is met. If you have one rule that changes the object's class to Nearline Storage and another rule that changes the object's class to Coldline Storage, but both rules use the exact same condition, the object's class always changes to Coldline Storage when the condition is met.

Lifecycle actions

The following actions are supported for a lifecycle rule:

- **Delete**: Delete live and/or archived objects. (A *live* object is one that is not an archived generation. See object versioning for details.) This action can be applied to both versioned and non-versioned objects. In a bucket with versioning enabled, deleting a live object archive the object, while deleting an archived object deletes the object permanently.

- **Warning:** Once an object is deleted, it cannot be undeleted. Take care in setting up your lifecycle rules so that you do not cause more data to be deleted than you intend. It is recommended that you test your lifecycle rules on development data before applying to production, and observe the expiration time metadata to ensure your rule has the effect you intend.
- **SetStorageClass**: Change the storage class of live and/or archived objects. This action can be applied to both versioned and non-versioned objects. This action supports the following storage class transitions:
- Original storage class
- New storage class
- Multi-Regional StorageNearline Storage
- Coldline StorageRegional StorageNearline Storage
- Coldline StorageNearline StorageColdline Storage
- **Note:** The `SetStorageClass` action incurs Class A operation charges. Charges are determined by the original class of the object. For example, changing 1,000 objects from Multi-Regional Storage to Coldline Storage counts as 1,000 Class A operations and is billed at the Class A operations rate for Multi-Regional Storage.

Lifecycle conditions

The following conditions are supported for a lifecycle rule:

- **Age**: This condition is satisfied when an object reaches the specified age (in days). When you specify the `Age` condition, you are specifying a Time to Live (TTL) for objects in a bucket with lifecycle management configured. The time when the `Age` condition is considered to be satisfied is calculated by adding the specified value to the object creation time. For example, if the object creation time is 2017/01/10 10:00 UTC and the `Age` condition is 10 days, then the condition is satisfied on and after 2017/01/20 10:00 UTC. This is true even if the object becomes archived through object versioning sometime after its creation.

- **CreatedBefore**: This condition is satisfied when an object is created before midnight of the specified date in UTC.
- **IsLive**: If the value is `true` , this lifecycle condition matches only live objects; if the value is `false` , it matches only archived objects. For the purposes of this condition, objects in non-versioned buckets are considered live.
- **MatchesStorageClass**: This condition is satisfied when an object in the bucket is stored as the specified storage class. Generally, if you intend to use this condition on Multi-Regional Storage or Regional Storage objects, you should also include `STANDARD` and `DURABLE_REDUCED_AVAILABILITY` in the condition to ensure all objects of similar storage class are covered.
- **Caution:** Do not use **`MULTI_REGIONALin if`** the bucket is located in a regional location, and do not use **`REGIONAL`** in **`MatchesStorageClass`** if the bucket is located in a multi-regional location. Doing either result in an error when you attempt to set a lifecycle management configuration.
- **NumberOfNewerVersions**: Relevant only for versioned objects. If the value of this condition is set to *N*, an object satisfies the condition when there are at least *N* versions (including the live version) newer than it. For live objects, the number of newer versions is considered to be 0. For the most recent archived version, the number of newer versions is 1 (or 0 if there is no live object), and so on.

All conditions are optional, but at least one condition is required. If you attempt to set an invalid lifecycle configuration, such as by using an action or condition that does not exist, you receive a `400 Bad request` error response, and any existing lifecycle configuration remains in place.

See Managing Object Lifecycles for examples of using lifecycle configurations. For the general format of a lifecycle configuration file, see the bucket resource representation for JSON or the lifecycle configuration format for XML.

Object lifecycle behavior

- Cloud Storage regularly inspects all the objects in a bucket for which Object Lifecycle Management is configured and performs all actions applicable according to the bucket's rules.
- **Caution:** Cloud Storage performs an action asynchronously, so there can be a lag between when the conditions are satisfied and when the action is taken. Your applications should not rely on lifecycle actions occurring within a certain amount of time after a lifecycle condition is met.
- For example, if an object meets the conditions for deletion, the object might not be deleted right away. You will continue to see the object until the lifecycle action is executed on it. You are not billed for the storage of the object, but accessing the object incurs any applicable operation and bandwidth charges as described in pricing.
- Updates to your lifecycle configuration may take up to 24 hours to go into effect. This means that when you change your lifecycle configuration, Object Lifecycle Management may still perform actions based on the old configuration for up to 24 hours.
- For example, if you change an `Age` condition from 10 days to 20 days, an object that is 11 days old could be deleted by Object Lifecycle Management up to 24 hours later, due to the criteria of the old configuration.
- An object lifecycle `Delete` action will not take effect on an object while the object either has an object hold placed on it or a retention policy that it has not yet fulfilled. Any `Delete` action that would occur while an object had a hold or retention policy restriction instead occurs after the restrictions no longer apply to the object.
- An object lifecycle `SetStorageClass` action is not affected by the existence of object holds or retention policies.

Early deletion of Nearline Storage and Coldline Storage objects

Object Lifecycle Management does not rewrite an object when changing its storage class. This means that when an object is transitioned to Nearline Storage or Coldline Storage using the

`SetStorageClass` feature, any subsequent early deletion and associated charges are based on the original creation time of the object, regardless of when the storage class changed.

For example, say you upload an object as Regional Storage, and 20 days later your lifecycle configuration changes the storage class of the object to Nearline Storage. If you then delete the object immediately, there is a 10-day early deletion charge, since the object existed for 20 days. If you delete the object 10 days after changing its storage class to Nearline Storage, there is no early deletion charge, since the object existed for 30 days.

In comparison, say you upload an object as Regional Storage and 20 days later change the storage class using a rewrite (again to Nearline Storage). If you then delete the object immediately afterward, you would incur a full 30-day early deletion charge, since the rewriting time becomes the new creation time. Similarly, if you waited 10 days after the rewrite to delete the object, you would incur a 20-day early deletion charge.

Expiration time metadata

If a `Delete` action is specified for a bucket with the `Age` condition (and no `NumberOfNewerVersions` condition), then some objects may be tagged with expiration time metadata. An object's expiration time indicates the time at which the object becomes (or became) eligible for deletion by Object Lifecycle Management. The expiration time may change as the bucket's lifecycle configuration or retention policy change.

Note that the absence of expiration time metadata does not necessarily mean the object will not be deleted, but rather that not enough information is available to determine when or if it will be deleted. For example, if the object creation time is 2013/01/10 10:00 UTC and the `Age` condition is set to 10 days, then the object expiration time is 2013/01/20 10:00 UTC. However, the expiration time will not be available for the object if:

- The `NumberOfNewerVersions` condition is also specified. In this case, older versions of the object may still be deleted if new versions are added.

- The `CreatedBefore` condition is also specified and set to “2013-01-01” because the object doesn’t satisfy this condition.
- The object is under a hold because Cloud Storage cannot know when the hold will be removed.

You are not charged for storage after the object expiration time even if the object is not deleted immediately. You can continue to access the object before it is deleted and is responsible for other charges (request, network bandwidth). If the expiration time is not available for an object, the object is charged for storage until the time it is deleted.

You can access an object’s expiration time in its metadata if it is available. The REST API returns the object’s expiration time in the `x-goog-expiration` response header.

When working with expiration times, keep in mind the following:

- If the bucket has a retention policy, the expiration time is the later of the Object Lifecycle Management age condition and the time the object satisfies the retention period specified by the retention policy.
- If there are multiple conflicting expiration times applicable to an object due to different lifecycle management rules, then the earliest applicable expiration time is used.

Options for tracking Lifecycle actions

To track the lifecycle management actions that Cloud Storage takes, use one of the following options:

- Use Cloud Storage Access Logs. This feature logs both the action and who performed the action. A value of `GCS Lifecycle Management` in the `cs_user_agent` field of the log entry indicates the action was taken by Cloud Storage in accordance with a lifecycle configuration.
- Enable Cloud Pub/Sub Notifications for Cloud Storage for your bucket. This feature sends notifications to a Cloud Pub/Subtopic of your choice when specified actions occur. Note that this feature does not record who performed the actions

——————————————————————————————————————-

Set Bucket Lifecycle

By default, buckets do not have lifecycle management enabled. To set or modify lifecycle configuration for an existing bucket you make a PUT request that is scoped to the bucket and you use the `lifecycle` query string parameter. You must include an XML document in the request body that contains the lifecycle configuration. Notice that you cannot set lifecycle configuration on a new bucket that you are creating.

You must have `FULL_CONTROL` permission to set or modify lifecycle configuration for an existing bucket. Also, you must be authenticated to use the PUT Bucket method.

Query string parameters

ParameterDescriptionRequired `lifecycle` You uses this to change the lifecycle configuration on an existing bucket. You must provide the lifecycle configuration document in the request body.No

Request Headers

See common request headers.

Request body elements

The following request body elements are applicable only if you use the `lifecycle` query string parameter to specify the lifecycle configuration for an existing bucket.

ElementDescription `LifecycleConfiguration` Defines the lifecycle management policies for the bucket, which contains 0 or more (up to 100) rules. Use an empty element (for example, `<LifecycleConfiguration/>`) to disable lifecycle management for the bucket. `Rule` Defines a lifecycle management rule, which is made of an action and the condition under which the action will be taken. `Action` Defines the action to be taken, which must contain one and only one specific action element. Required. `Delete` Action element to delete objects in the bucket. `SetStorageClass` Action element to change the storage class of an object. `Condition` Defines the condition under which the action will be taken, which must contain at least one specific condition element. Required. `Age` Condition element that matches objects over the specified age (in days). `CreatedBefore` Condition element that matches objects created before midnight of the specified date in UTC. The value is an ISO date string without a time zone, e.g. "2013-01-15". `IsLive` Condition element relevant only for versioned objects. Matches live objects if the value is `true` , or archived objects if the value is `false` . `MatchesStorageClass` Condition element that matches objects of the specified storage class. This condition can be added multiple times to the same rule in order to cover more than one storage class. `NumberOfNewerVersions` Condition element relevant only for versioned objects. If the value is N, the condition is satisfied when there are at least N versions (including the live version) newer than this version of the object.

Request syntax

```
PUT /?lifecycle HTTP/1.1
Host: <bucket>.storage.googleapis.com
Date: <date and time of the request>
Content-Length: <request body length>
Content-Type: <MIME type of the body>
Authorization: <authentication string>


<xml_document_defining_lifecycle_configuration>
```

Response Headers

The request can return a variety of response headers depending on the request headers you use.

Response body elements

The response does not include an XML document in the response body.

Example

The following sample enables lifecycle management for a bucket named example.com. For more examples, see the Managing Lifecycles page.

Request

```
PUT /example.com?lifecycle HTTP/1.1
Host: storage.googleapis.com
Content-Length: 220
Authorization: Bearer
ya29.AHES6ZRVmB7fkLtd1XTmq6mo0S1wqZZi3-Lh_s-6Uw7p8vtgSwg


<?xml version="1.0" ?>
<LifecycleConfiguration>
    <Rule>
        <Action>
            <Delete/>
        </Action>
        <Condition>
            <Age>30</Age>
        </Condition>
    </Rule>
</LifecycleConfiguration>
```

Response

```
HTTP/1.1 200 OK
Date: Thu, 12 Mar 2012 03:38:42 GMT
Expires: Mon, 01 Jan 1990 00:00:00 GMT
Cache-Control: no-cache, no-store, must-revalidate
Content-Length: 0
Content-Type: text/html
```