Loading Parquet data from Cloud Storage

This page provides an overview of loading Parquet data from Cloud Storage into BigQuery.

Parquet (http://parquet.apache.org) is an open source column-oriented data format that is widely used in the Apache Hadoop ecosystem.

When you load Parquet data from Cloud Storage, you can load the data into a new table or partition, or you can append to or overwrite an existing table or partition. When your data is loaded into BigQuery, it is converted into columnar format for Capacitor (/blog/products/gcp/inside-capacitor-bigquerys-next-generation-columnar-storage-format) (BigQuery's storage format).

When you load data from Cloud Storage into a BigQuery table, the dataset that contains the table must be in the same regional or multi- regional location as the Cloud Storage bucket.

For information about loading Parquet data from a local file, see Loading data into BigQuery from a local data source (/bigquery/docs/loading-data-local).

Parquet schemas

When you load Parquet files into BigQuery, the table schema is automatically retrieved from the self-describing source data. When BigQuery retrieves the schema from the source data, the alphabetically last file is used.

For example, you have the following Parquet files in Cloud Storage:

```
mybucket/00/
arquet
arquet
mybucket/01/
arquet
```

Running this command in the `bq` command-line tool loads all of the files (as a comma-separated list), and the schema is derived from `mybucket/01/b.parquet`:

```
ad \
rce_format=PARQUET \
et.table \
/mybucket/00/*.parquet","gs://mybucket/01/*.parquet"
```

When you load multiple Parquet files that have different schemas, identical columns specified in multiple schemas must have the same mode (/bigquery/docs/schemas#modes) in each schema definition.

When BigQuery detects the schema, some Parquet data types are converted to BigQuery data types to make them compatible with BigQuery SQL syntax. For more information, see Parquet conversions (#parquet_conversions).

Parquet compression

BigQuery supports the following compression codecs for data blocks in Parquet files:

- `Snappy`
- `GZip`
- `LZO_1C and LZO_1X`

Required permissions

When you load data into BigQuery, you need permissions to run a load job and permissions that let you load data into new or existing BigQuery tables and partitions. If you are loading data from Cloud Storage, you also need permissions to access to the bucket that contains your data.

BigQuery permissions

At a minimum, the following permissions are required to load data into BigQuery. These permissions are required if you are loading data into a new table or partition, or if you are appending or overwriting a table or partition.

- `bigquery.tables.create`

- `bigquery.tables.updateData`
- `bigquery.jobs.create`

The following predefined IAM roles include both `bigquery.tables.create` and `bigquery.tables.updateData` permissions:

- `bigquery.dataEditor`
- `bigquery.dataOwner`
- `bigquery.admin`

The following predefined IAM roles include `bigquery.jobs.create` permissions:

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

In addition, if a user has `bigquery.datasets.create` permissions, when that user creates a dataset, they are granted `bigquery.dataOwner` access to it. `bigquery.dataOwner` access lets the user create and update tables in the dataset by using a load job.

For more information on IAM roles and permissions in BigQuery, see Access control (/bigquery/access-control).

Cloud Storage permissions

To load data from a Cloud Storage bucket, you must be granted `storage.objects.get` permissions. If you are using a URI wildcard (#wildcard-support), you must also have `storage.objects.list` permissions.

The predefined IAM role `storage.objectViewer` (/storage/docs/access-control/iam) can be granted to provide both `storage.objects.get` and `storage.objects.list` permissions.

Loading Parquet data into a new table

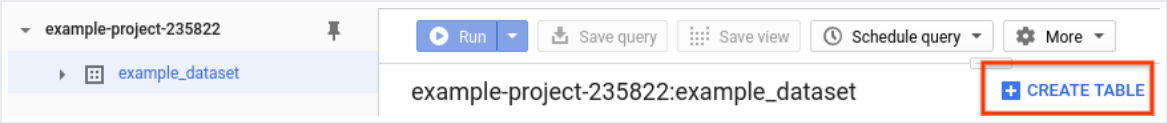
You can load Parquet data into a new table by using one of the following:

- The Cloud Console or the classic web UI
- The `bq` command-line tool's `bq load` command
- The `jobs.insert` API method and configuring a `load` job
- The client libraries

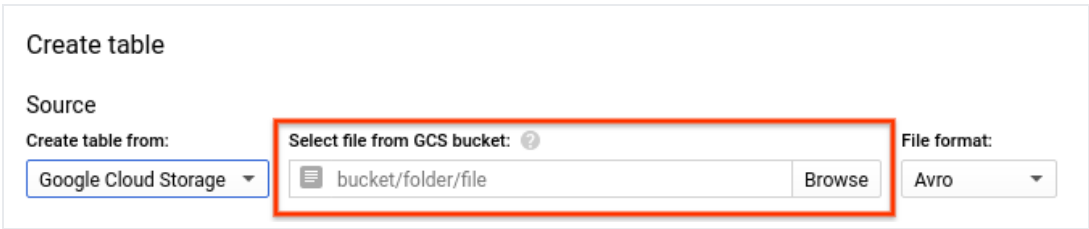
To load Parquet data from Cloud Storage into a new BigQuery table:

ConsoleClassic UI (#classic-ui)bq (#bq)API (#api)Go (#go)Java (#java)Node.js (#node.js)PHP (#php)P

1. Open the BigQuery web UI in the Cloud Console.
   Go to the Cloud Console (https://console.cloud.google.com/bigquery)
2. In the navigation panel, in the **Resources** section, expand your Google Cloud project and select a dataset.
3. On the right side of the window, in the details panel, click **Create table**. The process for loading data is the same as the process for creating an empty table.



4. On the **Create table** page, in the **Source** section:
   - For **Create table from**, select Cloud Storage.
   - In the source field, browse to or enter the Cloud Storage URI (#gcs-uri). Note that you cannot include multiple URIs in the Cloud Console, but wildcards (/bigquery/docs/loading-data-cloud-storage#load-wildcards) are supported. The Cloud Storage bucket must be in the same location as the dataset that contains the table you're creating.



   - For **File format**, select **Parquet**.
5. On the **Create table** page, in the **Destination** section:
   - For **Dataset name**, choose the appropriate dataset.

- Verify that **Table type** is set to **Native table**.
- In the **Table name** field, enter the name of the table you're creating in BigQuery.

6. In the **Schema** section, no action is necessary. The schema is self-described in Parquet files.
7. (Optional) To partition the table, choose your options in the **Partition and cluster settings**:
   - To create a partitioned table (/bigquery/docs/creating-column-partitions), click **No partitioning**, select **Partition by field** and choose a `DATE` or `TIMESTAMP` column. This option is unavailable if your schema does not include a `DATE` or `TIMESTAMP` column.
   - To create an ingestion-time partitioned table (/bigquery/docs/creating-partitioned-tables), click **No partitioning** and select **Partition by ingestion time**.
8. (Optional) For **Partitioning filter**, click the **Require partition filter** box to require users to include a `WHERE` clause that specifies the partitions to query. Requiring a partition filter can reduce cost and improve performance. For more information, see Querying partitioned tables (/bigquery/docs/querying-partitioned-tables#querying_partitioned_tables_2). This option is unavailable if **No partitioning** is selected.
9. (Optional) To cluster (/bigquery/docs/creating-clustered-tables) the table, in the **Clustering order** box, enter between one and four field names.
10. (Optional) Click **Advanced options**.
    - For **Write preference**, leave **Write if empty** selected. This option creates a new table and loads your data into it.
    - For **Number of errors allowed**, accept the default value of `0` or enter the maximum number of rows containing errors that can be ignored. If the number of rows with errors exceeds this value, the job will result in an `invalid` message and fail.
    - For **Unknown values**, leave **Ignore unknown values** unchecked. This option applies only to CSV and JSON files.
    - For **Encryption**, click **Customer-managed key** to use a Cloud Key Management Service key (/bigquery/docs/customer-managed-encryption). If you leave the **Google-managed key** setting, BigQuery encrypts the data at rest (/security/encryption-at-rest/default-encryption).
11. Click **Create table**.

Note: When you load data into an empty table by using the Cloud Console, you cannot add a label, description, table expiration, or partition expiration.

After the table is created, you can update the table's expiration, description, and labels, but you cannot add a partition expiration after a table is created using the Cloud Console. For more information, see Managing tables (/bigquery/docs/managing-tables).

Appending to or overwriting a table with Parquet data

You can load additional data into a table either from source files or by appending query results.

In the console and the classic BigQuery web UI, you use the **Write preference** option to specify what action to take when you load data from a source file or from a query result.

You have the following options when you load additional data into a table:

Console option	Classic web UI option	Command-line flag	BigQuery API property	Description
Write if empty	Write if empty	None	`WRITE_EMPTY`	Writes the data only if the table is empty.
Append to table	Append to table	`--noreplace` or `--replace=false`; if `--[no]replace` is unspecified, the default is append	`WRITE_APPEND`	(Default) Appends the data to the end of the table.
Overwrite table	Overwrite table	`--replace` or `--replace=true`	`WRITE_TRUNCATE`	Erases all existing data in a table before writing the new data.

If you load data into an existing table, the load job can append the data or overwrite the table.

You can append or overwrite a table by using one of the following:

- The Cloud Console or the classic web UI
- The `bq` command-line tool's `bq load` command
- The `jobs.insert` API method and configuring a `load` job

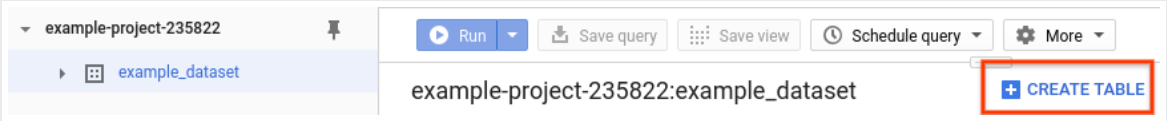
- The client libraries

This page does not cover appending or overwriting partitioned tables. For information on appending and iting partitioned tables, see: Appending to and overwriting partitioned table data uery/docs/managing-partitioned-table-data#append-overwrite).

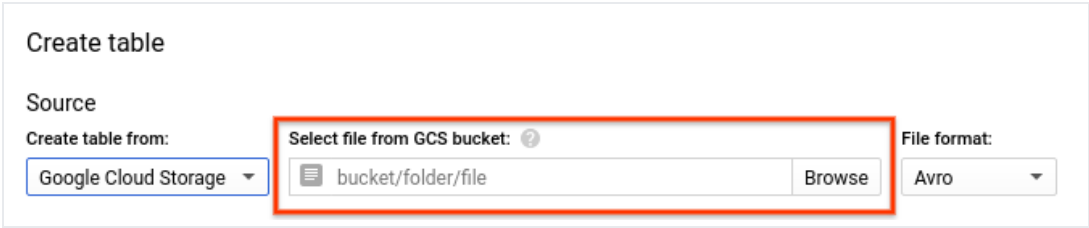
To append or overwrite a table with Parquet data:

ConsoleClassic UI (#classic-ui)bq (#bq)API (#api)Go (#go)Java (#java)Node.js (#node.js)PHP (#php)P

1. Open the BigQuery web UI in the Cloud Console.
   Go to the Cloud Console (https://console.cloud.google.com/bigquery)
2. In the navigation panel, in the **Resources** section, expand your Cloud project and select a dataset.
3. In the details panel, click **Create table**. The process for appending and overwriting data in a load job is the same as the process for creating a table in a load job.



4. On the **Create table** page, in the **Source** section:
   - For **Create table from**, select Cloud Storage.
   - In the source field, browse to or enter the Cloud Storage URI (#gcs-uri). Note that you cannot include multiple URIs in the BigQuery web UI, but wildcards (/bigquery/docs/loading-data-cloud-storage#load-wildcards) are supported. The Cloud Storage bucket must be in the same location as the dataset that contains the table you're appending or overwriting.



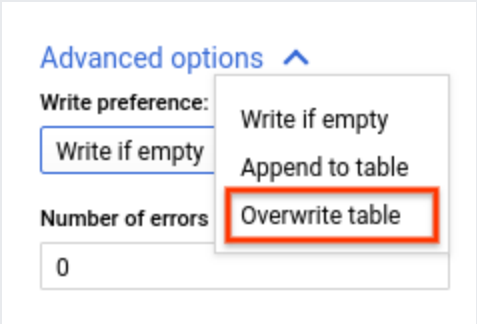
   - For **File format**, select **Parquet**.
5. On the **Create table** page, in the **Destination** section:
   - For **Dataset name**, choose the appropriate dataset.

- In the **Table name** field, enter the name of the table you're appending or overwriting in BigQuery.
- Verify that **Table type** is set to **Native table**.

6. In the **Schema** section, no action is necessary. The schema is self-described in Parquet files.

★ **Note:** It is possible to modify the table's schema when you append or overwrite it. For more information on supported schema changes during a load operation, see Modifying table schemas (/bigquery/docs/managing-table-schemas).

7. For **Partition and cluster settings**, leave the default values. You cannot convert a table to a partitioned or clustered table by appending or overwriting it, and the Cloud Console does not support appending to or overwriting partitioned or clustered tables in a load job.
8. Click **Advanced options**.
   - For **Write preference**, choose **Append to table** or **Overwrite table**.
   - For **Number of errors allowed**, accept the default value of `0` or enter the maximum number of rows containing errors that can be ignored. If the number of rows with errors exceeds this value, the job will result in an `invalid` message and fail.
   - For **Unknown values**, leave **Ignore unknown values** unchecked. This option applies only to CSV and JSON files.
   - For **Encryption**, click **Customer-managed key** to use a Cloud Key Management Service key (/bigquery/docs/customer-managed-encryption). If you leave the **Google-managed key** setting, BigQuery encrypts the data at rest (/security/encryption-at-rest/default-encryption).



9. Click **Create table**.

Loading hive-partitioned Parquet data

BigQuery supports loading hive partitioned Parquet data stored on Cloud Storage and populates the hive partitioning columns as columns in the destination BigQuery managed table. For more information, see Loading externally partitioned data (/bigquery/docs/hive-partitioned-loads-gcs).

Parquet conversions

BigQuery converts Parquet data types to the following BigQuery data types:

Type conversions

Parquet type	Parquet converted type(s)	BigQuery data type
BOOLEAN	NONE	Boolean
INT32	NONE, UINT_8, UINT_16, UINT_32, INT_8, INT_16, INT_32	Integer
INT32	DECIMAL (see DECIMAL annotation (#decimal-annotation))	Numeric
INT32	DATE	Date
INT64	NONE, UINT_64, INT_64	Integer
INT64	DECIMAL (see DECIMAL annotation (#decimal-annotation))	Numeric
INT64	TIMESTAMP_MILLIS	Timestamp
INT64	TIMESTAMP_MICROS	Timestamp
INT96	NONE	Timestamp
FLOAT	NONE	Floating point
DOUBLE	NONE	Floating point
BYTE_ARRAY	NONE	Bytes
BYTE_ARRAY	UTF8	String
FIXED_LEN_BYTE_ARRAY	DECIMAL (see DECIMAL annotation (#decimal-annotation))	Numeric

Parquet type	Parquet converted type(s)	BigQuery data type
FIXED_LEN_BYTE_ARRAY	NONE	Bytes

Other combinations of Parquet types and converted types are not supported.

Decimal annotation

Parquet types with the `DECIMAL` annotation can have at most a precision of 38 (total number of digits) and at most a scale of 9 (digits to the right of the decimal). The number of integer digits, which is the precision minus the scale, can be at most 29. For example, `DECIMAL(38, 9)` is supported because the precision is 38 and the scale is 9. In this example, the number of integer digits is 29. `DECIMAL(38, 5)` is not supported because it has a precision of 38 and a scale of 5. In this example, the number of integer digits is 33.

Column name conversions

A column name must contain only letters (a-z, A-Z), numbers (0-9), or underscores (_), and it must start with a letter or underscore. The maximum column name length is 128 characters. A column name cannot use any of the following prefixes:

- `_TABLE_`
- `_FILE_`
- `_PARTITION`

Duplicate column names are not allowed even if the case differs. For example, a column named `Column1` is considered identical to a column named `column1`.

Currently, you cannot load Parquet files containing columns that have a period (.) in the column name.

If a Parquet column name contains other characters (aside from a period), the characters are replaced with underscores. You can add trailing underscores to column names to avoid collisions. For example, if a Parquet file contains 2 columns `Column1` and `column1`, the columns are loaded as `Column1` and `column1_` respectively.

Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/), and code samples are licensed under the Apache 2.0 License (https://www.apache.org/licenses/LICENSE-2.0). For details, see the Google Developers Site Policies (https://developers.google.com/site-policies). Java is a registered trademark of Oracle and/or its affiliates.

Last updated 2020-08-10 UTC.