**Introduction to clustered tables**

This document provides an overview of table clustering capabilities in BigQuery.

**Overview**

When you create a clustered table in BigQuery, the table data is automatically organized based on the contents of one or more columns in the table’s schema. The columns you specify are used to collocate related data. When you cluster a table using multiple columns, the order of columns you specify is important. The order of the specified columns determines the sort order of the data.

Clustering can improve the performance of certain types of queries such as queries that use filter clauses and queries that aggregate data. When data is written to a clustered table by a query job or a load job, BigQuery sorts the data using the values in the clustering columns. These values are used to organize the data into multiple blocks in BigQuery storage. When you submit a query that contains a clause that filters data based on the clustering columns, BigQuery uses the sorted blocks to eliminate scans of unnecessary data. You might not see a significant difference in query performance between a clustered and unclustered table if the table or partition is under 1 GB.

Similarly, when you submit a query that aggregates data based on the values in the clustering columns, performance is improved because the sorted blocks collocate rows with similar values.

**When to use clustering**

Both partitioning and clustering can improve performance and reduce query cost.

Use clustering under the following circumstances:

* You don't need strict cost guarantees before running the query.
* You need more granularity than partitioning alone allows. To get clustering benefits in addition to partitioning benefits, you can use the same column for both partitioning and clustering.
* Your queries commonly use filters or aggregation against multiple particular columns.
* The cardinality of the number of values in a column or group of columns is large.

Use partitioning under the following circumstances:

* You want to know query costs before a query runs. Partition pruning is done before the query runs, so you can get the query cost after partitioning pruning through a [dry run](https://cloud.google.com/bigquery/docs/dry-run-queries). Cluster pruning is done when the query runs, so the cost is known only after the query finishes.
* You need partition-level management. For example, you want to set a partition expiration time, load data to a specific partition, or delete partitions.
* You want to specify how the data will be partitioned and what data is in each partition. For example, you want to define time granularity or define the ranges used to partition the table for integer range partitioning.

Prefer clustering over partitioning under the following circumstances:

* Partitioning results in a small amount of data per partition (approximately less than 1 GB).
* Partitioning results in a large number of partitions beyond the [limits on partitioned tables](https://cloud.google.com/bigquery/quotas#partitioned_tables).
* Partitioning results in your mutation operations modifying the majority of partitions in the table frequently (for example, every few minutes).

You can also combine partitioning with clustering. Data is first partitioned and then data in each partition is clustered by the clustering columns.

When the table is queried, partitioning sets an upper bound of the query cost based on partition pruning. There might be other query cost savings when the query actually runs, because of cluster pruning.

**Clustering partitioned tables**

In a partitioned table, data is stored in physical blocks, each of which holds one partition of data. A partitioned table maintains these properties across all operations that modify it: query jobs, data manipulation language (DML) statements, data definition language (DDL) statements, load jobs, and copy jobs. This requires BigQuery to maintain more metadata than a unpartitioned table. As the number of partitions increases, the amount of metadata overhead increases.

Although more metadata must be maintained, by ensuring that data is partitioned globally, BigQuery can more accurately estimate the bytes processed by a query before you run it. This cost calculation provides an upper bound on the final cost of the query.

In a clustered table, BigQuery automatically sorts the data based on the values in the clustering columns and organizes them in optimally sized storage blocks. You can achieve more finely grained sorting by creating a table that is clustered and partitioned. A clustered table maintains the sort properties in the context of each operation that modifies it. As a result, BigQuery might not be able to accurately estimate the bytes processed by the query or the query costs. When blocks of data are eliminated during a query, BigQuery provides a best effort reduction of the query costs.

**Automatic re-clustering**

As data is added to a clustered table, the newly inserted data can be written to blocks that contain key ranges that overlap with the key ranges in previously written blocks. These overlapping keys weaken the sort property of the table.

To maintain the performance characteristics of a clustered table, BigQuery performs automatic re-clustering in the background to restore the sort property of the table. For partitioned tables, clustering is maintained for data within the scope of each partition.