As we all know,Amazon Athena is an interactive query service that makes it easy to analyze data stored in Amazon S3 using standard SQL. Athena is serverless, so there is no infrastructure to manage, and you pay only for the queries that you run. Athena is easy to use. Simply point to your data in Amazon S3, define the schema, and start querying using standard SQL.

In this blog post, we will review some of the best practices and performance tuning tips to improve query performance and reduce cost.

1.Columnar Data Store
Column data stores are great for analytical queries because of its fast query speeds.They are awesome at aggregating large volumes of data for a subset of columns.This gives advantage for analytics users to fetch a portion of all available columns.Apache Parquet and Apache ORC are popular columnar data stores. They provide features that store data efficiently by employing column-wise compression, different encoding, compression based on data type, and predicate pushdown. They are also splittable. Generally, better compression ratios or skipping blocks of data means reading fewer bytes from Amazon S3, leading to better query performance.As a result, queries run faster and the cost (5$ perTB scanned) is much lower.**Avoid running "select *" queries as they impact on performance and render huge cost,so lets query only for the columns we need**.

2.Optimize S3 file sizes
Data can be stored in CSV, JSON, ORC, Parquet and even Apache web logs format. Queries run more efficiently when reading data can be parallelized and when blocks of data can be read sequentially. Ensuring that your file formats are splittable helps with parallelism regardless of how large your files may be.
Ensure file sizes are in between 200MB-1GB as they are considered optimal, but be careful of out of memory exceptions if your file size exceeds 1GB.**Avoid creating smaller size files.**

Some benefits of having optimal size files:
-Faster listing
-Fewer Amazon S3 requests
-Less metadata to manage

3.Partitioning
Partitions divide a table into distinct independent parts and keeps the related data together based on column values such as date, country, region, etc. In Athena, each partition is defined in the table schema and mapped to a S3 folder.You can restrict the amount of data scanned by a query by specifying filters based on the partition.Partitions let you read just the data you actually need by using the

partition column in your WHERE clause.
Partitioning by time (month, day or hour) is usually the recommended way for a better query performance.

4.Compression & Splitting files
Compressing your data can speed up your queries significantly, as long as the files are either of an optimal size (see the next section), or the files are splittable. The smaller data sizes reduce network traffic from Amazon S3 to Athena.

Splittable files allow the execution engine in Athena to split the reading of a file by multiple readers to increase parallelism. If you have a single unsplittable file, then only a single reader can read the file while all other readers sit idle. Not all compression algorithms are splittable.
Generally, the higher the compression ratio of an algorithm, the more CPU is required to compress and decompress data.

Athena supports the following compression formats:

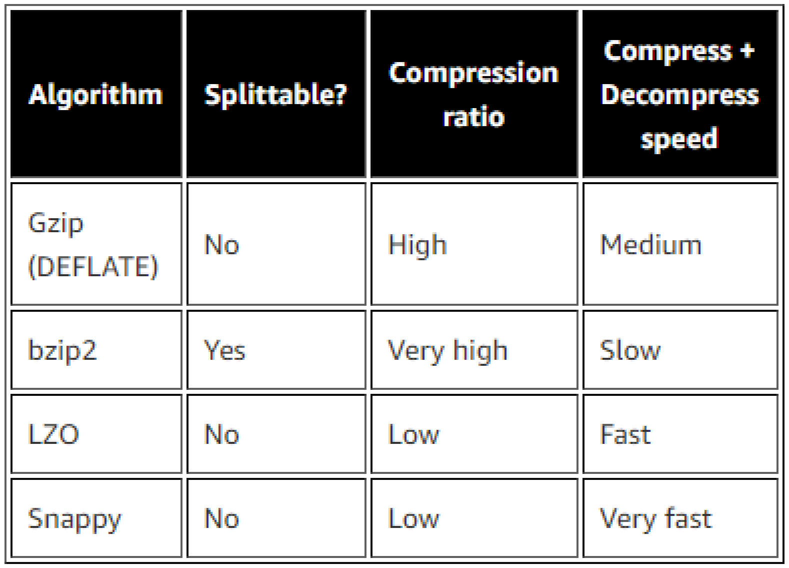
-
- SNAPPY. This is the default compression format for files in the Parquet data storage format.

-
- ZLIB. This is the default compression format for files in the ORC data storage format.

-
- LZO

- GZIP. Use the GZIP compression in Athena for querying Amazon Kinesis Data Firehose logs. Athena and Amazon Kinesis Data Firehose each support different versions of SNAPPY, so GZIP is the only compatible format.

For Athena, we recommend using either Apache Parquet or Apache ORC, which compress data by default and are splittable.



Algorithm	Splittable?	Compression ratio	Compress + Decompress speed
Gzip (DEFLATE)	No	High	Medium
bzip2	Yes	Very high	Slow
LZO	No	Low	Fast
Snappy	No	Low	Very fast

5.Optimize Joins
Athena doesn’t handle joins like a regular database.It distributes the table on the left to workers and them streams over the table on the right.It’s critical that you specify the bigger table on the left side of the join.
If you need to join 2 big tables, you will probably need to pre-join the data using a data processing system. Athena queries timeout after 30 minutes. If it happens to you, consider pre-joining your tab.

Recommended Tip

You can save 30%-90% on your query costs and get better performance:(Below Pricing example would give clear picture)
-Compression(Snappy,ZLIB,GZIP)
-Partitoning(Time based)
-Converting your data into columnar formats(Parquet,ORC)

For example:

Logs of total size of 3TB on Amazon S3 in text format,table with 3 equally sized columns.
No compression

-Athena will scan the entire file
-Query Cost: $15(3TB Scanned,3*$5=15)
Compression using GZIP
-3:1 Compression gains
-Query cost:$5(1TB scanned,1*$5/TB=$5)
Convert to Parquet,columnar format
-Athena can read 1/3 columns that is relevant for the query
-Query Cost:$1.67(0.33TB scanned,0.33*$5/TB=$1.67)

When to use Athena:
-If you want to run Adhoc queries
-If you want to query unstructured,semi structured,and structured data stored in Amazon S3.

These are some of the recommended practices for better performance using AWS Athena.