**ASS-26.2**

**Explain in breif with an example**

**● Bucketing:**

• Bucketing concept is based on (hashing function on the bucketed column) mod (by total

number of buckets). The hash\_function depends on the type of the bucketing column.

• Records with the same bucketed column will always be stored in the same bucket.

• We use CLUSTERED BY clause to divide the table into buckets.

• Physically, each bucket is just a file in the table directory, and Bucket numbering is 1-based.

• Bucketing can be done along with Partitioning on Hive tables and even without partitioning.

• Bucketed tables will create almost equally distributed data file parts, unless there is skew in data.

• Bucketing is enabled by setting hive.enforce.bucketing = true;

Bucketed tables offer efficient sampling than by non-bucketed tables. With sampling, we can try out queries on a fraction of data for testing and debugging purpose when the original data sets are very huge.

• As the data files are equal sized parts, map-side joins will be faster on bucketed tables than non-bucketed tables.

• Bucketing concept also provides the flexibility to keep the records in each bucket to be sorted by one or more columns. This makes map-side joins even more efficient, since the join of each bucket becomes an efficient merge-sort.

**Bucketing V/S Partitioning:**

|  |  |
| --- | --- |
| **Partitioning** | **Bucketing** |
| Hive Partitioning dividing the large amount of data into number pieces of folders based on table columns value. | Hive bucketing is responsible for dividing the data into number of equal parts |
| If you want to use Partition in hive then you should use PARTITIONED BY (COL1,COL2…etc) command while hive table creation. | If you want to use bucketing in hive then you should use CLUSTERED BY (Col) command while creating a table in Hive. |
| We can perform partition on any number of columns in a table by using hive partition concept. | We can perform Hive bucketing optimization only on one column only not more than one. |
| Partitioning is works better when the cardinality of the partitioning field is not too high. | bucketing works well when the field has high cardinality and data is evenly distributed among buckets. |

**Sampling:**

1. Sampling is concerned with the selection of a subset of data from a large dataset to run queries and verify results.

2. The dataset may be too large to run queries on the whole data. Therefore in development and testing phases it is a good idea to run queries on a sample of dataset.

3. We can run Hive queries on a sample of data using the TABLESAMPLE clause. Any column can be used for sampling the data. We need to provide the required sample size in the queries.

4. We can use TABLESAMPLE clause to bucket the table on the given column and get data from only some of the buckets.

5. example: TABLESAMPLE (BUCKET x OUT OF y [ON colname])

colname indicates the column to be used to bucket the data into y buckets[1-y]. All the rows which are in the bucket x are returned.

6. If the table is not bucketed on the column(s) used in sampling, TABLESAMPLE will scan the entire table and fetch the sample.

7. If the hive table is bucketed on some column(s), then we can directly use that column(s) to get a sample. In this case Hive need not read all the data to generate sample as the data is already organized into different buckets using the column(s) used in the sampling query.

8. Hive will read data only from some buckets as per the size specified in the sampling query.

9. Block sampling allows Hive to select at least n% data from the whole dataset. Sampling granularity is at the HDFS block size level. If HDFS block size is 64MB and n% of input size is only 10MB, then 64MB of data is fetched.