***1.Bucketing:***

* Partition helps to increase the efficiency while performing the query process on a table.
* Assume a condition where it contains a huge [dataset](https://acadgild.com/big-data/big-data-development-training-certification).
* There were time after each partitioning on particularly on field
* When the size of the partitioned file size doesn’t match with the expected one.
* It will remain huge and there is a need to manage the result of partition into the different parts.
* In order to overcome this problem Hive was provided with the concept of ***Bucketing***.
* Which will allow the user to divide data inside the table into more manageable parts.
* Hive partition will further divide the table into number of partitions
* All those partitions will subdivided further into more number of parts known as ***Buckets or Clusters***.
* The concept of Bucketing is based on the **Hash function.**
* And it purelywhich depends upon the type of bucketing column.
* Records bucketed by the same column will be saved in the same bucket.

**---------------------------------------------------------------------------------------------------------------------**

**2.Partitioning vs Bucketing**

* **Partitioning** - for distributing load horizontally.
* And there will be performance benefit which helps in organizing data in a logical fashion.
* For eg : while dealing with a large employee table we will run using WHERE clause.
* So the result will be restricted to some particular country or department .
* In case of faster query response: Hive table should be partitioned using (Country String,Dept String).
* Hive will get modified by creating more subdirectories which will reflect the partitioning structure like
* …/employee/country=SAI/MATHURRI=STUDENT
* If the limits of the query for employee table from country=MATHURRI
* Then it will only consider the directory which contains country=MATHURRI.
* So the performance of the query will be dramatically improved.
* Under one case when the partitioning scheme reflects the common filtering only.
* Partitioning feature plays a vital role in Hive. but, when a design creates many partitions some may optimize the queries.
* There were other drawback too while using too many partitions a large number of Hadoop files and directories will be created unnecessarily .
* And will cause overhead to Name Node, so all metadata for the file system in memory.
* **Bucketing** - technique used for decomposing data sets into number of manageable parts.
* Eg –a table with date is used in the top level partition
* And employee\_id in the second level which will create a number of partitions.
* But when we bucket the employee table and as well when we use the employe-id in the bucketing column, then the value of the column will be hashed by the user-defined number into buckets.
* The records which contains the same employee\_id will be stored in same bucket.
* Assume that the the employee\_id count higher than number of buckets then each bucket will contain more number of employee\_id.
* But CLUSTEREDBY(employee\_id)INTOABCBUCKETS should be used while creating the table .
* In which ABC - number of buckets .
* There were several advantages in Bucketing.
* And the number of buckets will be fixed so there will not be any not fluctuations within the data.
* When two tables are bucketed with employee\_id  logically correct sampling will be created in hive.
* Bucketing also helps in efficient map-side joins etc.

***3.Sampling.***

***Sampling bucketized table:***

*table\_sample: TABLESAMPLE (BUCKET x OUT OF y [ON colname])*

* Tablesample will allow the users to write queries for data samples instead of using the whole table.
* The tablesample clause can be added to any table when FROM clause is used .
* The buckets number starts from 1.
* Column name will indicate a sample to each row in the table.
* Column name can be on a non-partition columns in the table or  may also be for **rand()** which indicates the sampling of entire row .
* Instead used for an individual column.
* 'bucketed' table row will be on the column name which is randomly used into the y buckets which numbered 1 throughout y.
* And all the rows which all belongs to the bucket of x will be returned.

***Block sampling:***

*block\_sample: TABLESAMPLE (n PERCENT)*

* Block sampling will allow Hive to pick a least of n% data size as inputs.
* CombineHiveInputFormat will only be supported
* Special compression formats will not be handled.
* If sample is failed then the input of MapReduce job will be whole table/partition.
* When done in HDFS block level the sampling granularity will block size.
* Eg- when block size - 256MB, and if n% of input size is -100MB, and one will 256MB of data.