**Hive Partitioning for Datasets - Lab Manual**

**Aim:**

To understand and implement Hive partitioning on datasets, improving query performance by reducing the amount of data scanned.

**Partitioning in Hive:**

Partitioning allows tables to be divided based on column values, reducing query execution time by scanning only relevant partitions.

**1. Creating a Partitioned Table**

**Definition:** A table is created with partitions based on a specific column.

**Query:**

CREATE TABLE employees\_partitioned (

employee\_id INT,

name STRING,

department STRING,

salary FLOAT

)

PARTITIONED BY (year INT)

STORED AS ORC;

**2. Loading Data into Partitions**

**Definition:** Data is inserted into the table with specific partitions.

**Query:**

LOAD DATA INPATH '/hdfs\_path/employees\_2023.csv'

INTO TABLE employees\_partitioned

PARTITION (year=2023);

**3. Querying Partitioned Data**

**Definition:** Queries can filter specific partitions to improve performance.

**Query:**

SELECT \* FROM employees\_partitioned WHERE year = 2023;

**Output:**

| **employee\_id** | **name** | **department** | **salary** | **year** |
| --- | --- | --- | --- | --- |
| 101 | Alice | HR | 60000 | 2023 |
| 102 | Bob | IT | 75000 | 2023 |

**4. Viewing Partitions**

**Definition:** Displays existing partitions in a table.

**Query:**

SHOW PARTITIONS employees\_partitioned;

**Output:**

year=2022

year=2023

year=2024

**5. Dropping a Partition**

**Definition:** Removes a specific partition and its data from the table.

**Query:**

ALTER TABLE employees\_partitioned DROP PARTITION (year=2022)

**Result:**

Hive partitioning enhances query performance by limiting data scans to relevant partitions. It is an essential optimization technique for large datasets in distributed environments.