BDH-LAB PRACTICAL 06

|  |
| --- |
| **Name:HimanshuDhomane** |
| **Reg no: 21070823** |
| **Roll no: 43** |

**Aim:**

**Case Study: Analyzing Olympic Data Set Using Hive**

**Theory:**

Hive is a data warehousing tool built on top of Hadoop for querying and analyzing large datasets stored in HDFS. Hive uses SQL-like queries, making it accessible for data analysts. This case study focuses on using Hive to extract meaningful insights from the Olympic Dataset.

**Key Concepts of Hive**

1. **Hive Architecture**:
   * **Metastore**: Stores schema and metadata information of Hive tables.
   * **HiveQL**: A query language similar to SQL used to interact with the data.
   * **Execution Engine**: Converts HiveQL queries into MapReduce jobs.
   * **Storage**: Hive tables store data in HDFS, and queries access data directly from HDFS files.
2. **Tables in Hive**:

Hive supports **managed tables** (where Hive manages the data) and

**external tables** (data is managed externally).

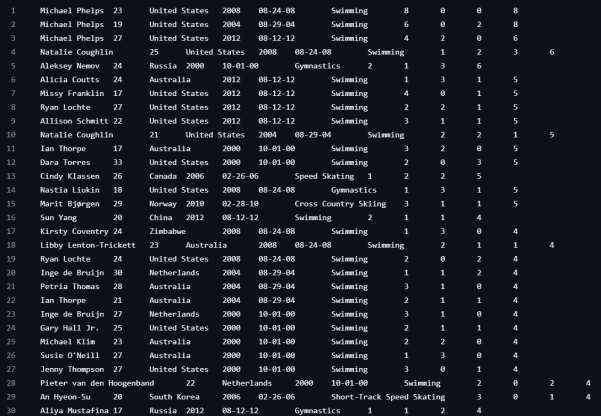
* + **Row format delimited**: Specifies how the data is structured (e.g., CSV, TSV).
  + **Storage format**: Tables can be stored as **text, ORC, Parquet**, etc.

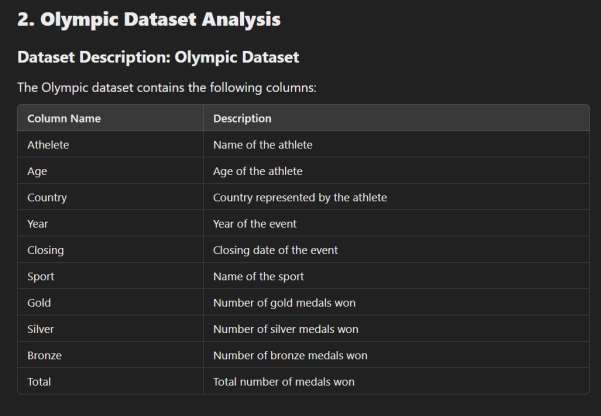
1. **Partitioning and Bucketing**:
   * **Partitioning**: Divides a table into smaller parts based on a column (e.g., year).
   * **Bucketing**: Further divides data into buckets for optimization.

**Olympic Dataset Use Case**:

* + **Sports analysts** and **committees** can use this data to evaluate performance trends by country or sport.
  + Identifying medal trends helps countries strategize future participation and investments in sports.

**Dataset:**





**Implementation:**

#Step:1

#Creation of Table in Hive and Loading of data:

create table olympic (athelete STRING,age INT,country STRING,year STRING,closing STRING,sport STRING,gold INT,silver INT,bronze INT,total INT) row format delimited fields terminated by ‘\t’ stored as textfile;

load data local inpath ‘/home/acadgild/Downloads/olympic\_data.csv’ into table olympic;

select country,SUM(total) from olympic where sport = “Swimming” GROUP BY country;

select year,SUM(total) from olympic where country = “India” GROUP BY year;

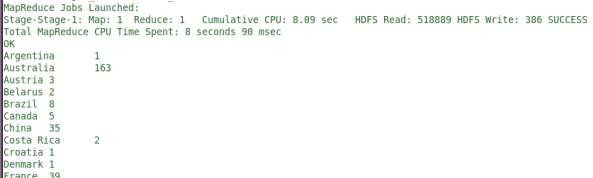
select country,SUM(total) from olympic GROUP BY country; select country,SUM(gold) from olympic GROUP BY country;

Create table Olympic(athelete string, age int, country string, year string , closing string sport string , gold int, silver int, bronze int, total int) row format delimited fields terminated by ‘\t’ stored as textfile;

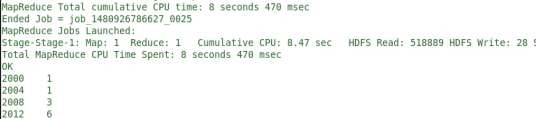
Load data local inpath ‘/home/LocalFileSystem/Hive/Olympic.csv’ into table Olympic;

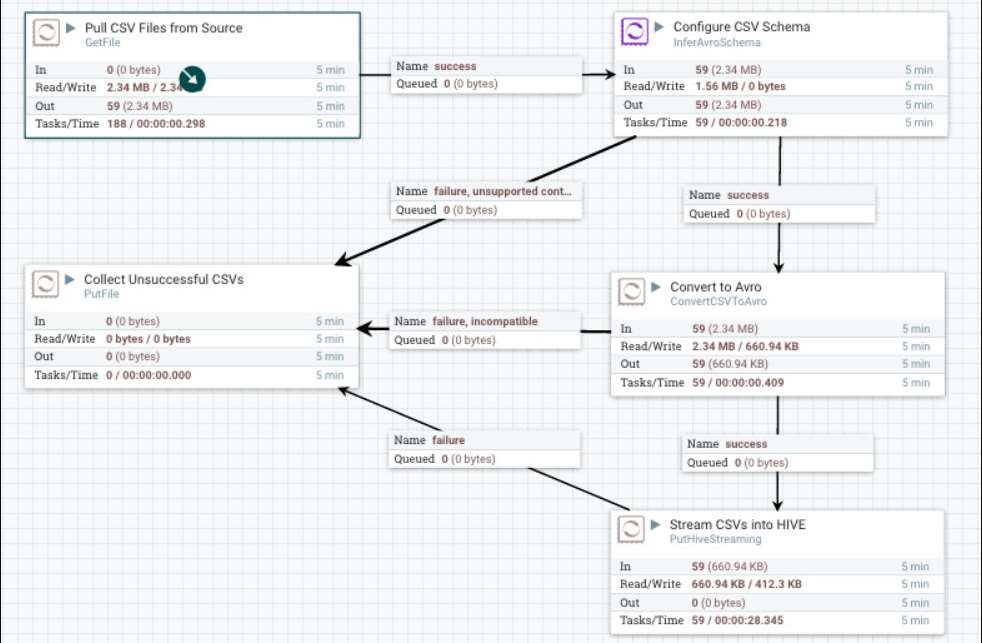
**Q. Total number of medals won by each country in swimming.**

**Select country, sum(total) rom Olympic where sport=’swimming’ group by country;**



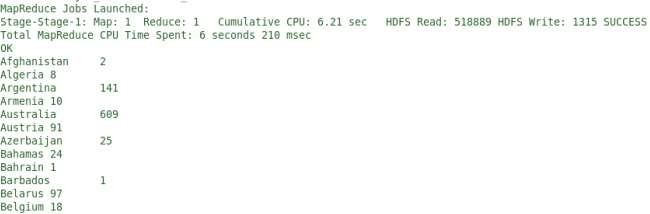
**Q. Display number of medals India won year wise**

**Select year, sum(total) from olympic where country=’India’ group by year;**



**Q. Find the total number of medals each country won display the name along with total medals.**

Select country, sum(total) from Olympic group by country;



**Conclusion:**

This case study demonstrates how to use **Hive SQL** for data analysis on two different datasets. In the **Olympic Dataset**, queries provide insights into medal counts by country, sport, and year. Hive simplifies working with large datasets by providing a SQL-like interface for quick and efficient analysis.

This practical implementation offers a glimpse into how businesses and analysts can leverage Hive for extracting actionable insights from diverse datasets.