**## Creating Database ##**

create database battingdb;

use battingdb;

**### 1. Load batting.csv into a mysql in a database battingdb and table batting ###**

**## creating Table ##**

create table batting(playerID varchar(30),yearID int,stint int,teamID varchar(10),lgID varchar(10),G int,G\_batting int,AB int,R int,H int,2B int,3B int,HR int,RBI int,SB int,CS int,BB int,SO int,IBB int,HBP int,SH int,SF int,GIDP int,G\_old int);

**##lLoading Data into Table ##.**

Load data infile '/home/cloudera/Lab/DataFiles/Batting1.csv' into table batting fields terminated by ',' Lines terminated by '\n';

**### 2. Sqoop the details into hdfs. ###**

sqoop import --connect jdbc:mysql://localhost/battingdb --username root --password cloudera --table batting --m 1

hadoop fs -cat /user/cloudera/batting/part-m-00000;

**### 3. Sqoop the details into hive. ###**

type command

>>hive;

create database battingdb;

use battingdb;

create table batting2(playerID STRING,yearID int,stint int,teamID STRING,lgID STRING,G int,G\_batting int,AB int,R int,H int,B2 int,B3 int,HR int,RBI int,SB int,CS int,BB int,SO int,IBB int,HBP int,SH int,SF int,GIDP int,G\_old int) row format delimited fields terminated by ',' stored as textfile;

LOAD DATA LOCAL INPATH '/home/cloudera/Lab/DataFiles/Batting1.csv' into table batting;

**### 4. Implement a PIG script to ###**

**### a) Find the total count of participation of G 112 ###**

batting\_list = LOAD '/home/cloudera/Lab/DataFiles/Batting1.csv' USING PigStorage(',') as (playerID:chararray,yearID:int,stint:int,teamID:chararray,lgID:chararray,G:int,G\_batting:int,AB:int,R:int,H:int,B2:int,B3:int,HR:int,RBI:int,SB:int,CS:int,BB:int,SO:int,IBB:int,HBP:int,SH:int,SF:int,GIDP:int,G\_old:int);

dump batting\_list;

count\_g = FILTER batting\_list BY G == 112;

group\_count\_g = GROUP count\_g All;

total\_count = foreach group\_count\_g Generate COUNT(count\_g.G);

dump total\_count;

store total\_count into 'count\_g112';

**### b) Find the player details with "david" ###**

david = Filter batting\_list by(playerID MATCHES 'david.\*');

dump david;

**### c) Find the average count of "NL" ###**

NL\_filter = Filter batting\_list by lgID =='NL';

NL\_Group = Group NL\_filter All;

NL\_avg = foreach NL\_Group Generate AVG(NL\_filter.G\_batting);

DUMP NL\_avg

**### d) Find the count of teams ###**

team\_count = GROUP batting\_list by teamID;

team\_group = GROUP team\_count All;

result\_count = Foreach team\_group Generate COUNT(team\_count);

dump result\_count

**### 5. Implement a Hive script to ##**

### a) Find the total count of player details with "david" ###

select count(\*) from batting where playerID REGEXP 'david[a-z]\*';

clustered by (playerID) INTO 3 buckets

**### b) Create a patition on the TEAMID ###**

**### c) Create 3 buckets on the partition. ###**

create external table batting\_part(playerID string,

yearID int,

stint int,

lgID string,

G int,

G\_batting int,

AB int,

R int,

H int,

B2 int,

B3 int,

HR int,

RBI int,

SB int,

CS int,

BB int,

SO int,

IBB int,

HBP int,

SH int,

SF int,

GIDP int,

G\_old int)

partitioned by (teamID string)

clustered by (lgID) INTO 3 buckets

row format delimited

fields terminated by ','

stored as textfile;

create external table batting\_hive(playerID string,

yearID int,

stint int,

teamID string,

lgID string,

G int,

G\_batting int,

AB int,

R int,

H int,

B2 int,

B3 int,

HR int,

RBI int,

SB int,

CS int,

BB int,

SO int,

IBB int,

HBP int,

SH int,

SF int,

GIDP int,

G\_old int)

row format delimited

fields terminated by ','

stored as textfile;

from batting\_hive bat INSERT OVERWRITE TABLE batting\_part PARTITION(teamID)

select bat.playerID,bat.yearID,

bat.stint ,bat.teamID,bat.lgID ,

bat.G ,bat.G\_batting ,

bat.AB ,bat.R ,bat.H ,

bat.B2 ,bat.B3 ,

bat.HR ,bat.RBI ,

bat.SB ,bat.CS ,bat.BB ,

bat.SO ,bat.IBB, bat.HBP,

bat.SH ,bat.SF,bat.GIDP,

bat.G\_old

DISTRIBUTE BY teamID;

LOAD DATA LOCAL INPATH '/home/cloudera/Desktop/Batting.csv' OVERWRITE INTO TABLE batting\_hive;

**### d) Extract the details on player "aaronha01" ###**

select \* from batting\_part where playerID='aaronha01';

**### e) Find the count of teams ###**

select count(distinct(teamID)) from batting\_hive;

**### 6. Using python ###**

**### a) Write a map-reducer program to find the total count of the players ###**

from mrjob.job import MRJob

from mrjob.step import MRStep

import sys

class MRWordFrequencyCount(MRJob):

def mapper1(self, \_, lines):

data = lines.split(',')

players = data[0].strip()

yield players,None

def combiner(self, word, counts):

yield word,None

def reducer1(self, key, counts):

yield "total players",key

def reducer2(self, key,word):

c =0

# for i in word:

# c+=1

yield key,len(list(word))

def steps(self):

return [ MRStep(mapper=self.mapper1,reducer=self.combiner),MRStep(reducer=self.reducer1),MRStep(reducer=self.reducer2)]

if \_\_name\_\_ == '\_\_main\_\_':

MRWordFrequencyCount.run()

**### b) Write a map-reducer program to find the total number of the teams. ###**

from mrjob.job import MRJob

from mrjob.step import MRStep

import sys

class MRWordFrequencyCount(MRJob):

def mapper1(self, \_, lines):

data = lines.split(',')

players = data[3].strip()

yield players,None

def combiner(self, word, counts):

yield word,None

def reducer1(self, key, counts):

yield "total teams",key

def reducer2(self, key,word):

c =0

for i in word:

c+=1

yield key,c

def steps(self):

return [ MRStep(mapper=self.mapper1,reducer=self.combiner),MRStep(reducer=self.reducer1),MRStep(reducer=self.reducer2)]

if \_\_name\_\_ == '\_\_main\_\_':

MRWordFrequencyCount.run()

**### 7. Visualize the battings.csv based on the frequency of palyer inclusion yearwise. ###**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

df1 = dataset.groupby('yearID')['playerID'].count()

df1.head(5)

plt.figure(figsize=(15,10),dpi=100)

plt.plot(df1, linestyle='dotted', marker = '\*', color = 'blue', label = 'Players')

plt.xlabel('Year')

plt.ylabel('Players Included')

plt.show()

**### 8. From halloffame.csv ###**

**### List the managers. ###**

#List the managers.

from mrjob.job import MRJob

class MRmyjob(MRJob):

def mapper(self,\_,line):

#split the line with tab separated fields

data = line.split(',')

hofid = data[0].strip()

category = data[7].strip()

if category == 'Manager':

yield hofid,None

def reducer(self, key, list\_of\_values):

yield "manager",key

if \_\_name\_\_ == '\_\_main\_\_':

MRmyjob.run();

**### Find the numbers of votes got year wise by "chancfr01h".###**

from mrjob.job import MRJob

class MRmyjob(MRJob):

def mapper(self,\_,line):

#split the line with tab separated fields

data = line.split(',')

hofid = data[0].strip()

year = data[1].strip()

votes = data[5].strip()

if hofid == 'chancfr01h':

try:

yield year,int(votes)

except:

yield year,0

def reducer(self, key, list\_of\_values):

yield key,sum(list\_of\_values)

if \_\_name\_\_ == '\_\_main\_\_':

MRmyjob.run();

**### Count the votes got by each person overall. ###**

**#Count the votes got by each person overall.**

from mrjob.job import MRJob

class MRmyjob(MRJob):

def mapper(self,\_,line):

#split the line with tab separated fields

data = line.split(',')

hofid = data[0].strip()

year = data[1].strip()

votes = data[5].strip()

try:

yield hofid,int(votes)

except:

yield hofid,0

def reducer(self,key,list\_of\_values):

yield key,sum(list\_of\_values)

if \_\_name\_\_ == '\_\_main\_\_':

MRmyjob.run();

**### 9. Using hive,partition by year. Then, find the year wise count of participants,**

**find the total votes got by the players. ###**

|  |
| --- |
|  |
| create table halloffame(hofID STRING, yearid INT, votedBy STRING, ballots INT, needed INT, votes INT,inducted STRING, category STRING, needed\_note STRING) row format delimited fields terminated by ',' stored as textfile; |
|  |  |
|  | LOAD DATA LOCAL INPATH '/home/cloudera/Lab/DataFiles/HallOfFame.csv' into table halloffame; |
|  |  |
|  | set hive.exec.dynamic.partition.mode=nonstrict; |
|  | set hive.exec.dynamic.partition=true; |
|  | set hive.enforce.bucketing=true; |
|  |  |
|  | create table halloffame\_part hof(hofID STRING, votedBy STRING, ballots INT, needed INT, votes INT,inducted STRING, category STRING, needed\_note STRING) partitioned by(yearid INT) row format delimited fields terminated by ',' lines terminated by '\n'; |
|  |  |
|  | from halloffame hof INSERT OVERWRITE TABLE halloffame\_part PARTITION(yearid) select hof.hofID, hof.votedBy, hof.ballots, hof.needed, hof.votes, hof.inducted, hof.category, hof.needed\_note, hof.yearid DISTRIBUTE BY yearid; |
|  |  |
|  | select yearid, count(hofid) from halloffame\_part group by yearid; |
|  |  |
|  | select hofid, sum(votes) from halloffame\_part group by hofid; |
|  |  |

**###10. Using python, map-reducer, find the average votes on the year 1936.###**

from mrjob.job import MRJob

class MRmyjob(MRJob):

def mapper(self,\_,line):

#split the line with tab separated fields

data = line.split(',')

date = data[1].strip()

votes = data[5].strip()

if date == '1936':

yield "votes",votes

def reducer(self, key, list\_of\_values):

count = 0

total = 0

for x in list\_of\_values:

total = total + float(x);

count = count+1

avglen = ("%.2f" % (total / count))

yield key,avglen

if \_\_name\_\_ == '\_\_main\_\_':

MRmyjob.run();