Assignment Hive-1

Objective: In this assignment, we will create ipl database from 3 given files.

Given file:

1) Batting.csv

This file contains batting statistics of few players in the specified format.

	A	B	C	D	E	F	G	H
1	ID	Name	Team	Runs	HS	Average	Strikerate	Sixs
2	1	RV Uthappa	Kolkata	572	83	47.66	137.5	14
3	2	GJ Maxwell	Punjab	533	95	44.41	193.81	35
4	3	DA Warner	Hyderaba	524	90	52.4	140.48	24
5	4	DR Smith	Chennai	501	79	38.53	136.14	33

2) Bowling.csv

	A	B	C	D	E	F	G	H	I
1	ID	Name	Team	Overs	Runs	Wkts	Ave	Econ	SR
2	1	RV Uthappa	Kolkata	36.4	299	16	18.68	8.15	13.7
3	2	GJ Maxwell	Punjab	14.2	113	3	37.66	7.88	28.6
4	4	DR Smith	Chennai	13	135	4	33.75	10.38	19.5
5	7	SK Raina	Chennai	8	43	4	10.75	5.37	12

3) Player_value.csv

	A	B	C
1	ID	Player	value
2	1	RV Uthappa	10.5
3	2	GJ Maxwell	9.5
4	3	DA Warner	5.5
5	4	DR Smith	5
6	5	AB de Villiers	6

Q1) Load batting and bowling data into hive tables partitioned by their respective team

Solution:

Partitioned data in our case will be created in two ways

a) Java MR function will be employed to partition Batting.csv and respective data sets will be put into tables.

b) Dynamic partitions will be created from Bowling.csv

JAVA MR function

Mapper class

```
public class bat_map extends Mapper<LongWritable,Text,Text,Text>{
	public void map(LongWritable key,Text value,Context context){
		String line = value.toString();
		int i = 1;
		String tab = null;
		String keyy=null;
		StringTokenizer token = new StringTokenizer(line,",");
		tab = token.nextToken();
		while(token.hasMoreElements()){
			if(i != 2){
				tab = tab + "," + token.nextToken();
				i=i+1;
			}
			else{
				keyy = token.nextToken();
				i=i+1;
			}
		}
		try {
			context.write(new Text(keyy), new Text(tab));
		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (InterruptedException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
	}
}
```

Reducer Class

```
public class bat_red extends Reducer<Text,Text,NullWritable,Text>{
	public void reduce(Text key,Iterable<Text> values,Context context){
		NullWritable none = NullWritable.get();
		MultipleOutputs<NullWritable,Text> m = new MultipleOutputs<NullWritable,Text>(context);

		for(Text t:values){
			try {
				m.write(none, t, key.toString());
			} catch (IOException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			} catch (InterruptedException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
		}
		try {
			m.close();
		} catch (IOException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		} catch (InterruptedException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
	}
}
```

Runner_class:

```
public class bat_run {
        public static void main(String[] args) throws IOException, ClassNotFoundException, InterruptedException {
                Configuration conf = new Configuration();
                conf.set("heading", "breaking in to partitions to be used in hive");

                Job job = new Job(conf);
                job.setJarByClass(bat_run.class);

                FileInputFormat.setInputPaths(job, args[0]);
                FileOutputFormat.setOutputPath(job, new Path(args[1]));

                job.setMapperClass(bat_map.class);
                job.setReducerClass(bat_red.class);

                job.setInputFormatClass(TextInputFormat.class);
                job.setOutputFormatClass(TextOutputFormat.class);

                job.setMapOutputKeyClass(Text.class);
                job.setMapOutputValueClass(Text.class);

                job.setOutputKeyClass(NullWritable.class);
                job.setOutputValueClass(Text.class);
                job.setNumReduceTasks(1);

                System.exit(job.waitForCompletion(true)?0:1);
        }

}
```

Go to parent directory

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
Bangalore-r-00000	file	0.1 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup
Chennai-r-00000	file	0.17 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup
Delhi-r-00000	file	0.17 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup
Hyderabad-r-00000	file	0.06 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup
Kolkata-r-00000	file	0.15 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup
Mumbai-r-00000	file	0.07 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup
Punjab-r-00000	file	0.22 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup
Rajasthan-r-00000	file	0.14 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup
_SUCCESS	file	0 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup
_logs	dir				2014-05-24 05:46	rwxr-xr-x	training	supergroup
part-r-00000	file	0 KB	1	64 MB	2014-05-24 05:46	rw-r--r--	training	supergroup

Hive Scripts for loading batting data set:

```
create database case_ipl;
use case_ipl;

show tables;

create table batting
(id int, name String, runs int, high_score int, average float, strike_rate float, sixes int)
partitioned by (team String)
row format delimited
fields terminated by ','
stored as textfile;

describe batting;

load data inpath '/user/training/hive_case/out_bat/Bangalore-r-00000'
overwrite into table batting
partition (team = 'Bangalore');

load data inpath '/user/training/hive_case/out_bat/Chennai-r-00000'
overwrite into table batting
partition (team = 'Chennai');

load data inpath '/user/training/hive_case/out_bat/Delhi-r-00000'
overwrite into table batting
partition (team = 'Delhi');

load data inpath '/user/training/hive_case/out_bat/Hyderabad-r-00000'
overwrite into table batting
partition (team = 'Hyderabad');

load data inpath '/user/training/hive_case/out_bat/Kolkata-r-00000'
overwrite into table batting
partition (team = 'Kolkata');

load data inpath '/user/training/hive_case/out_bat/Mumbai-r-00000'
overwrite into table batting
partition (team = 'Mumbai');

load data inpath '/user/training/hive_case/out_bat/Punjab-r-00000'
overwrite into table batting
partition (team = 'Punjab');

load data inpath '/user/training/hive_case/out_bat/Rajasthan-r-00000'
overwrite into table batting
partition (team = 'Rajasthan');

select * from batting;
```

Problem with above method is that it becomes cumbersome with too many partitions

Dynamic Partitions for bowling data set

```
create table bowling_no_partition
(id int, name String, team String, overs float, runs int, wickets int, avg float, economy float, strike_rate float)
row format delimited
fields terminated by ','
stored as textfile;

load data local inpath '/home/training/Desktop/hive/Bowling.csv'
overwrite into table bowling_no_partition;

create table bowling
(id int, name String, overs float, runs int, wickets int, avg float, economy float, strike_rate float)
partitioned by (team String)
row format delimited
fields terminated by ','
stored as textfile;

set hive.exec.dynamic.partition.mode=nonstrict;
set hive.exec.dynamic.partition=true;

insert overwrite table bowling
partition (team)
select bl.id, bl.name, bl.overs, bl.runs, bl.wickets, bl.avg, bl.economy, bl.strike_rate, bl.team
from bowling_no_partition bl;

set hive.exec.dynamic.partition.mode=nonstrict;
set hive.exec.dynamic.partition=true;

drop table bowling_no_partition;
```