**PIG Use Case: Pokemon Data Analysis**

**Note: Before running pig scripts, start all Hadoop Daemons with start-all.sh command, and check using jps command whether all daemons are running or not i.e. Namenode, Datanode, SecondaryNamenode, ResourceMangaer, NodeManager. There is no need to start jobhistoryserver because pig is launched in LOCAL mode.**

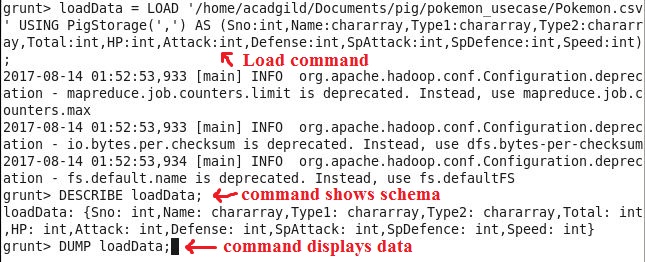
**What has to be done?**

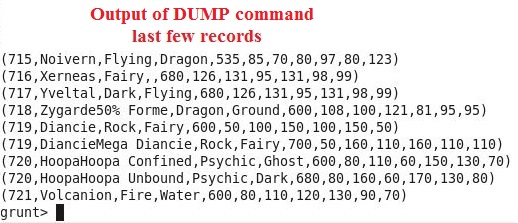
The Pokémon Fight League (PFL) management for the 2017 match has first of all decided a **minimum criterion for the entry selection process, i.e.** **defense power for any Pokémon should ideally be greater than 55.** Our job is to give 2 lists consisting names of those Pokémons who will be eligible for taking part in PFL this year from the list of all the participating 800 Pokémons.

In order to complete above specified task, below steps are followed:

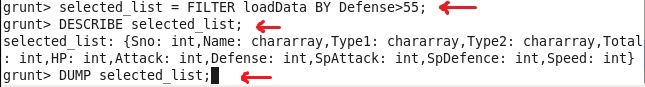
**Step 1: Launch pig in LOCAL mode using below command.**

**Step 2: Load the dataset inside PIG using LOAD command where loadData relation holds loaded data. Using DESCRIBE command we can check the schema of loadData relation and finally using DUMP command we can check whether data has been loaded correctly or not from local file system to loadData relation in pig.**





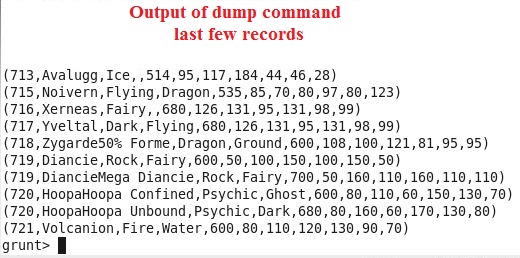
#### Step 3: To find the list of players that have been selected in the qualifying round (DEFENCE>55), below command is used:

**Explanation of above commands:**

**Line 1:** **loadData** relation is filtered on the basis of **Defense>55** condition and stored in **selected\_list** relation, and hence out of all the 800 Pokémons, only 544 are eligible to take part in the tournament.

**Line 2:** schema of **selected\_list** relation is displayed.

**Line 3:** data inside **selected\_list** relation is displayed.



In order to get the count, refer the next step.

#### G:\ACADGILD\course material\Hadoop\Sessions\Session 5\Assignments\Assignment3\Screenshots\Query2.1.JPGStep 4: To state the number of players taking part in the competition after getting selected in the qualifying round, below command is used:

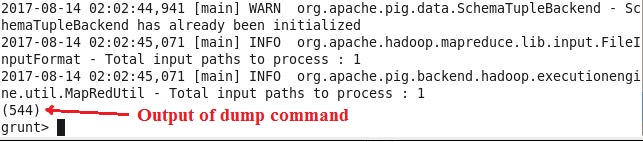
Explanation of above commands:

**Line 1:** **GROUP ALL** command groups all the tuples of **selected\_list** relation in one group, and the result is stored in **group\_selected\_list** relation.

**Line 2:** schema of **group\_selected\_list** relation is displayed, where we can see all tuples are grouped into one group.

**Line 3:** **count\_selected\_list** relation stores **COUNT** of selected players.

**Line 4:** schema of **count\_selected\_list** relation is displayed, where we can see only one field appears i.e. countOfPlayers.

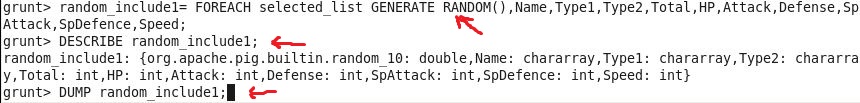
**Line 5:** data inside **count\_selected\_list** relation is displayed.

Now, two teams of 5 Pokémons need to be extracted out randomly from the selected list of players i.e. 544 players.

Seems like, this way we will have 2 lists containing 5 Pokémons each so to have them fight with each other.

To find random list of players refer next step.

#### Step 5: Using random(), generate random numbers for each Pokémon in the selected list.

**Below command is used to generate random players List 1:**

**Explanation of above commands:**

**Line 1:** generates random list of players and stores in **random\_include1** relation.

**Line 2:** schema of **random\_include1** relation is displayed, which shows one extra field is added, i.e. random and rest fields are same as that of loadData relation.

**Line 3:** shows data inside **random\_include1** relation.



#### G:\ACADGILD\course material\Hadoop\Sessions\Session 5\Assignments\Assignment3\Screenshots\Query4.1.JPGStep 6: Arrange the random\_include1 list in a descending order according to a random column i.e. first column using below command:

**Explanation of above commands**:

**Line 1:** gives us consequently a layer arrangement to pick the player from random list.

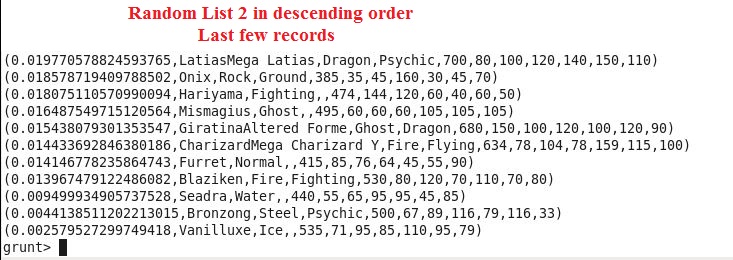
**Line 2:** schema of relation random1\_desc created at line 1 is displayed.

**Line 3:** data of relation is displayed.



So, using Step 5 and Step 6 first random list is created from which first player to fight will be selected, likewise we can create second random list from which second player for fight will be selected. For this refer Step 7.

#### G:\ACADGILD\course material\Hadoop\Sessions\Session 5\Assignments\Assignment3\Screenshots\Query5.1.JPGStep 7: Now on a new relation again associate random numbers for each Pokémon and arrange in descending order according to a random column, below commands are used to do this:

**Explanation**: same as in Steps 5 and 6.

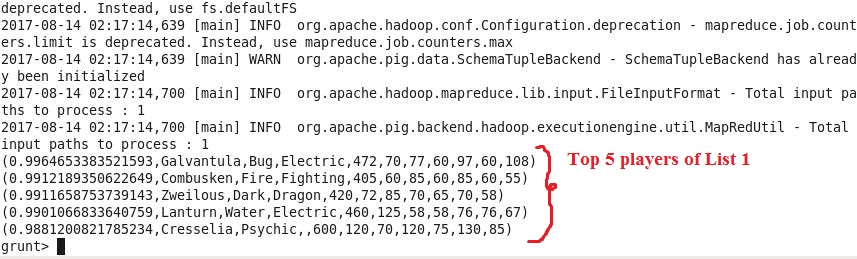
Now, we have to find top 5 players from each list, for this refer below steps.

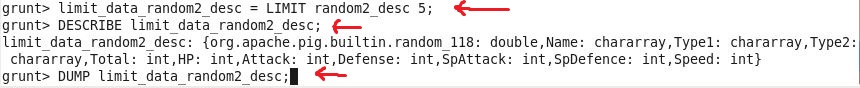
#### G:\ACADGILD\course material\Hadoop\Sessions\Session 5\Assignments\Assignment3\Screenshots\Query6.1.JPGStep 8: From the two different descending lists of random Pokémons, select the top 5 Pokémons for 2 different players, using below commands:

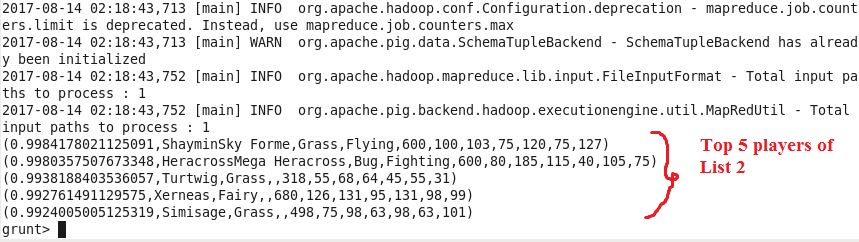
**Above lines of code find top 5 players from List 1. Explanation is as follows:**

**Line 1:** stores top 5 players of first list in **limit\_data\_random1\_desc** relation.

**Line 2:** shows schema of **limit\_data\_random1\_desc** relation.

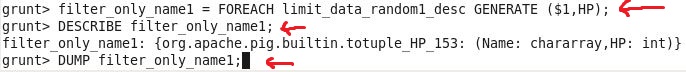
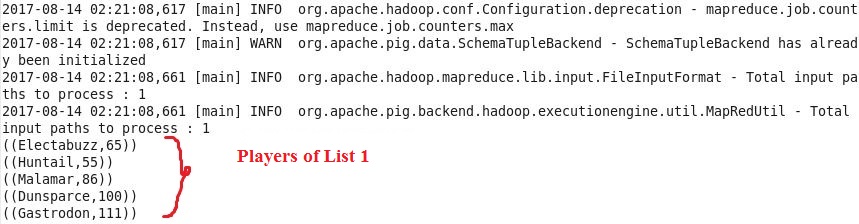
**Line 3:** displays data inside **limit\_data\_random1\_desc** relation.

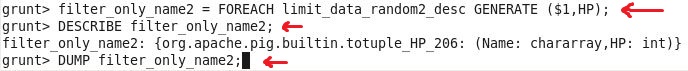
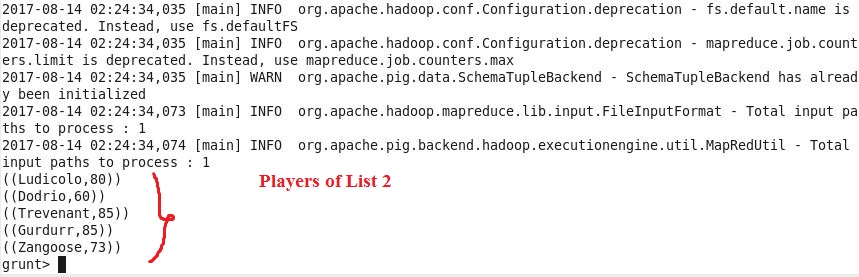
**To find top 5 players from List 2, follow below lines of code:**



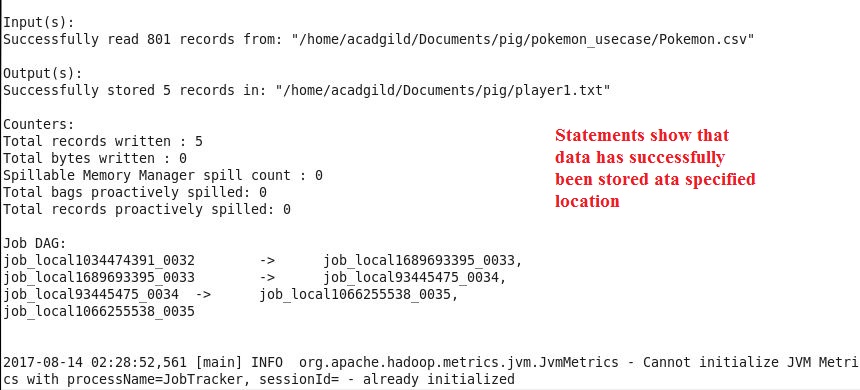
#### Step 9: Store the data on a local drive to announce for the final match, by the name player1 and player2 (only show the NAME and HP).

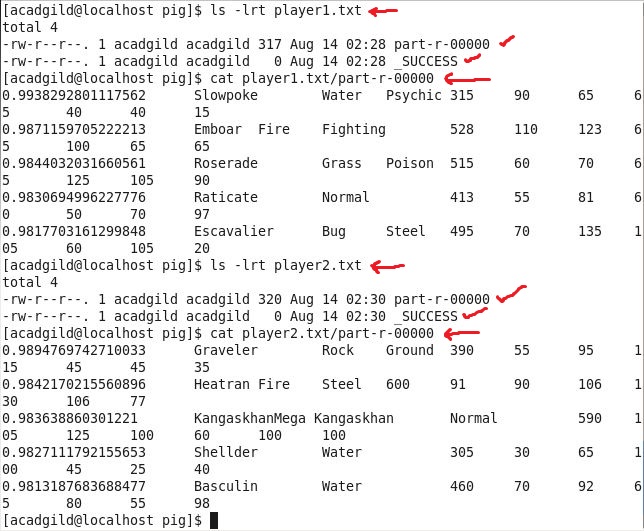
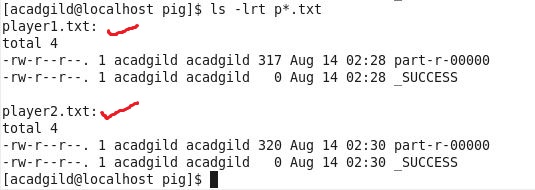
**Filtering List 1 to find top 5 players with Name and HP only, using below commands:**



**Filtering List 2 to find top 5 players with Name and HP only, using below commands:**

Two lists of Players have been generated, now storing the result in Local file system.



**Below co** **mmands verify that data has been stored successfully at location /home/acadgild/Documents/pig.**