**Final Project Report**

**Name: Xuyang Li NUID:001409590**

**1. Project Overview**

This project is focus on analysis soccer data by using Hadoop MR and Pig. The dataset is the 10-year match data of English Premier League, Spanish La Liga League, German Bundesliga League, Italian Serie A League and French Ligue One League. To know more about the dataset, please refer to **Section 2 Dataset Specification**. To know about the analysis performed, please refer to **Section 3 Analysis and Result**.

Developing Environment:

Host Operating System: Win 10 Home

Virtualization software: Virtual Box 6.0.14

Virtual Machine: 18.04

Hadoop Version: 2.7.3

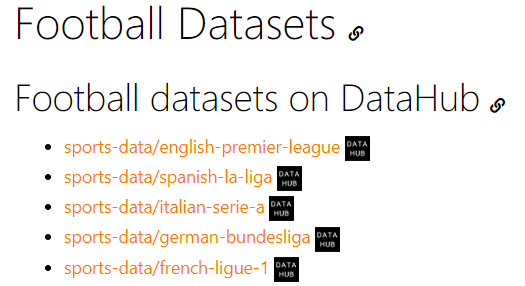
Pig Version: 0.17.0

Hbase Version: 2.1.8

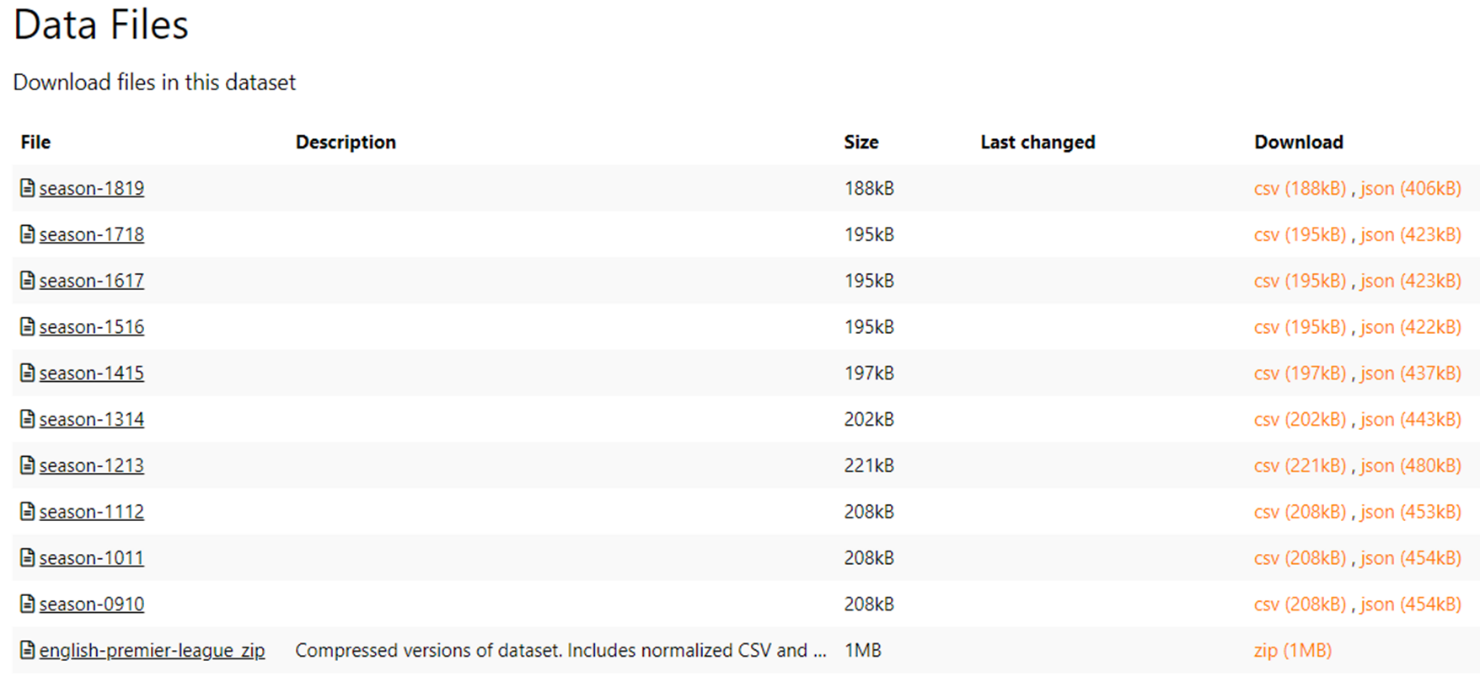
Hive Version: 2.3.6

**2. Dataset Specification**

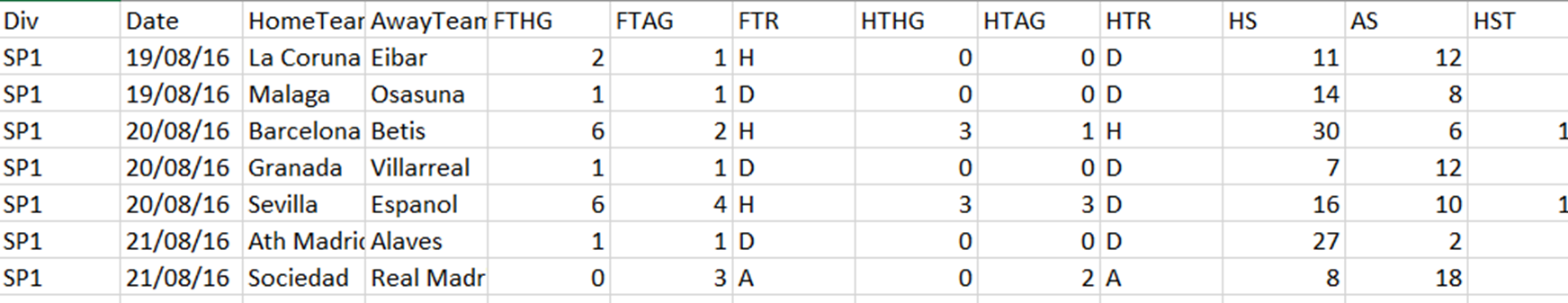
The Dataset is coming from Datahub.io. The link is <https://datahub.io/collections/football>. Under the “Football datasets on DataHub” heading, there are 5 hyperlinks linked to the of English Premier League, Spanish La Liga League, German Bundesliga League, Italian Serie A League and French Ligue One League. See figure below.



When you clicked each link, you can find recent ten seasons data of each league, from season 09-10 to season 18-19. See figure below. They provide both CSV and JSON format for download.



Each record (each line) of the CSV file represents a match between two teams. See figure below.

****

And part of the columns is described below:

Div = League Division

Date = Match Date (dd/mm/yy)

HomeTeam = Home Team

AwayTeam = Away Team

FTHG = Full Time Home Team Goals

FTAG = Full Time Away Team Goals

FTR = Full Time Result (H=Home Win, D=Draw, A=Away Win)

HTHG = Half Time Home Team Goals

The dataset is number intensive dataset, which is very suitable to be analyzed by Hadoop MR and other big data tools.

**3. Analysis and Result**

**3.1 Analysis One**

**3.1.1 Problem statement:**

Get the teams which downgraded in La Liga in 17-18 season and teams which upgraded in 17-18 season.

**3.1.2 Approach:**

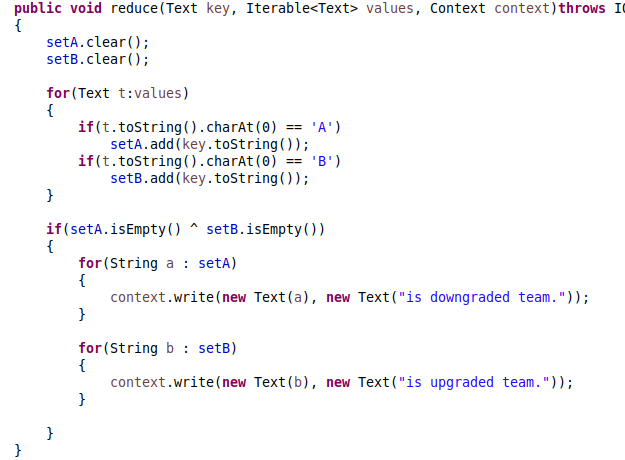
Step 1: Get distinct team list from season 17-18 and season 18-19.

Step 2: Perform anti-join between two team lists.

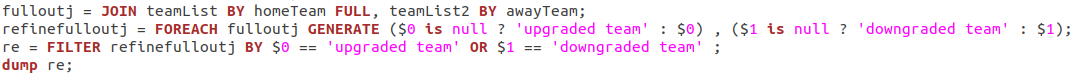
The reason of performing anti-join:

Teams that exist in season 17-18 but not in season 18-19 is a downgraded team and teams that exist in season 18-19 but not in season 17-18 is an upgraded team.

The core code of the anti-join part in Hadoop MR:

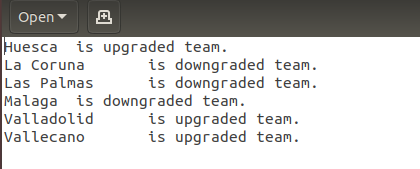


While in Pig, it does not provide anti-join grammar. The solution is first performing full outer join first then filtering the non-empty tuples. The snippet of the Pig script:

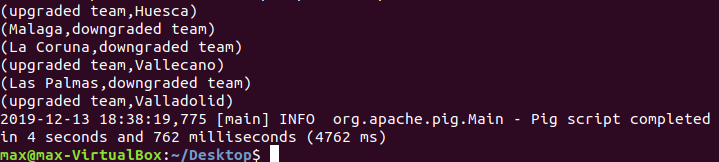


**3.1.3 Result and Validation**

Hadoop MR result:

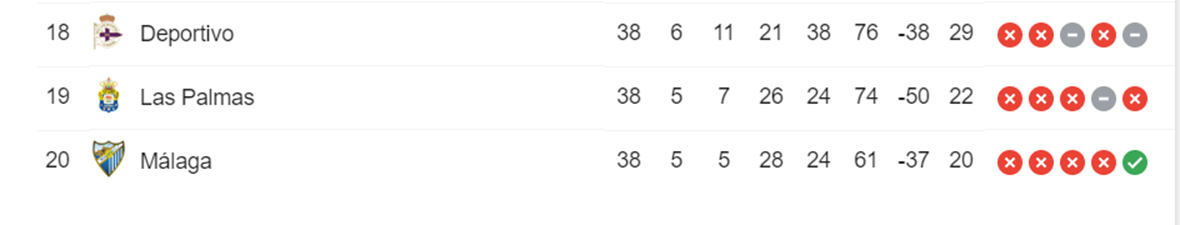


Pig result:



Result validation:

Teams that downgraded from 17-18 season. Snapshot is from Google. Notice that full name of Deportivo is Deportivo de La Coruña, which is La Coruna in the running result.



Teams that upgraded from 17-18 season. Snapshot is from Wikipedia. Notice that Valladolid upgraded finally in qualification matches.



**3.2 Analysis Two**

**3.2.1 Problem Statement:**

Get the most tough 3 teams which Barcelona played with in the past 10 seasons.

**3.2.2 Approach:**

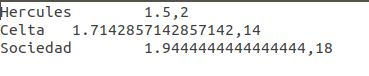
Step 1: Get the average points that Barcelona played with each team. (Win is 3 points, draw is 1 point, lose is 0 point.)

Step 2: Sort the team by average points in ascending order to get top 3.

**3.2.3 Result and Validation**

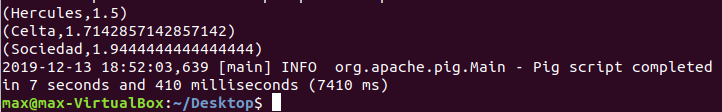
Hadoop MR result:

The first word is the team name, the second word is the average points Barcelona got from the team, the third word is how many matches Barcelona played with the team.



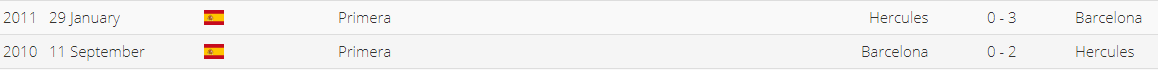
Pig result:

The first field of the tuple is the team name, the second field is the average points Barcelona got from the team.



Result validation:

Currently there is no data to validate the all the result. But as a super fan of Barcelona, I can say the result reflecting the truth. Barcelona always plays hard with the teams in the list. As we can see from the Hadoop MR result, Barcelona only play two matches with Hercules, this part of result can be validated in the next figure.



Barcelona only got 3 points in two matches, which means that 1.5 points in average.

**3.3 Analysis Three**

**3.3.1 Problem Statement:**

Construct the standing of the Premier league in season 18-19.

**3.3.2 Approach:**

Step 1: Count each team’s number of win matches, draw matches and lose matches, score, lose goal, and goal difference, calculate the points.

Step 2: Sort the team order by points in descending order first and by goal difference in ascending order second.

**3.3.3 Result and Validation**

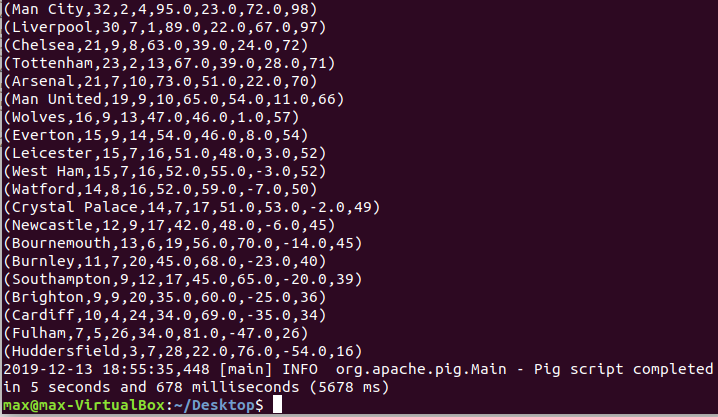
Hadoop MR result:

The first word is the team name, then follows are number of win matches, draw matches, lose matches, goal in total, lose goal in total, goal difference in total and points.



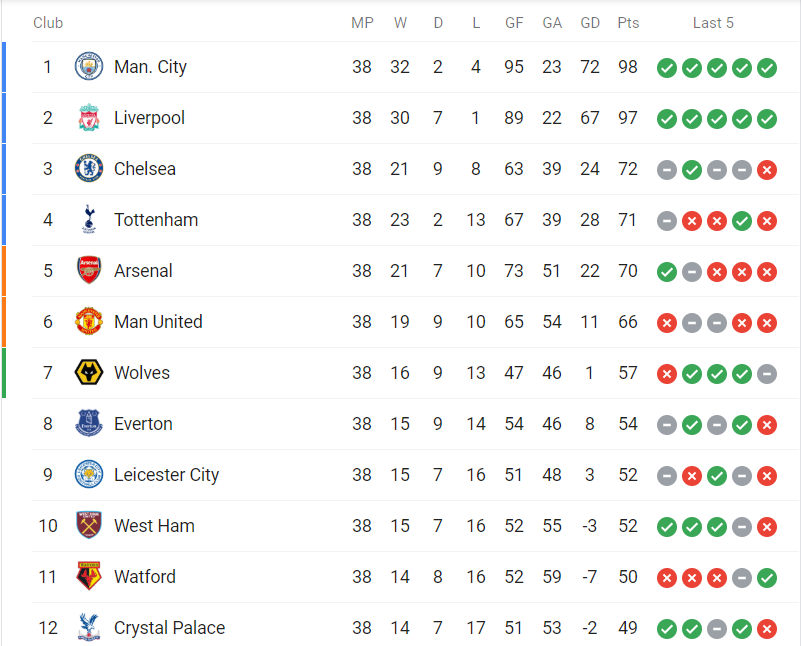
Pig result:

The result format is exactly same with the Hadoop MR.



Result Validation:

Standing of Premier league 18-19 season. Snapshot is from Google.

****

**3.4 Analysis Four**

**3.4.1 Problem Statement:**

Get the champions of La Liga in the past 10 seasons.

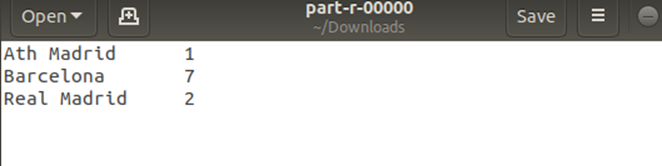
**3.4.2 Approach:**

Step 1: For-loop demo3 10 times, get the standing result of every season (10 seasons in total).

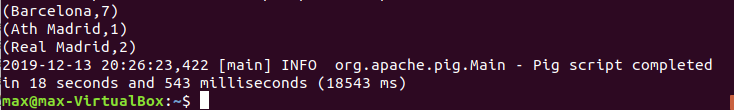
Step 2: Get the first record from each standing (the champion of each season), count how my time they win the champions

**3.4.3 Result and Validation**

Hadoop MR result:



Pig result:



Result Validation:



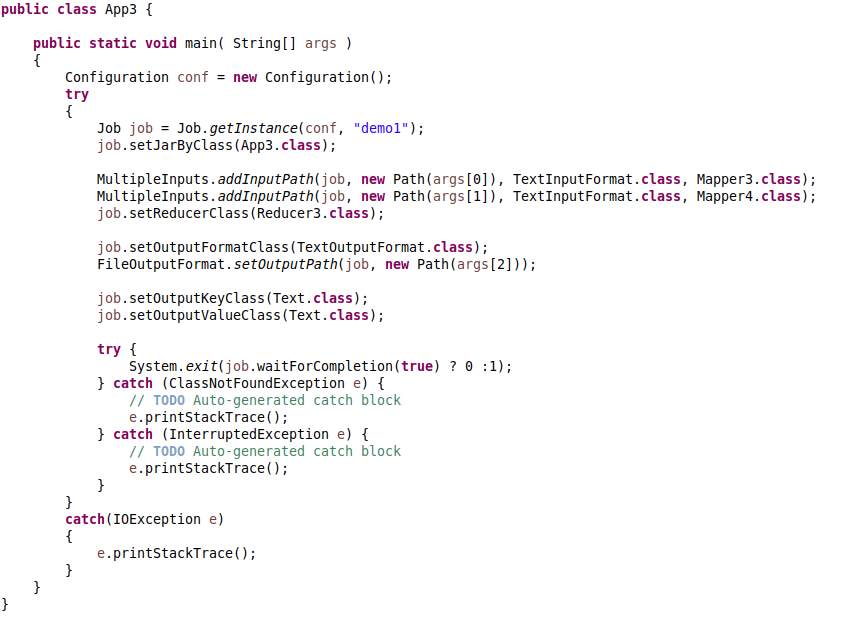
List of La Liga champions in the past 10 seasons. Snapshot is from Wikipedia. As we can see from the result, Barcelona won 7 times champions, Real Madrid 2 times, Atletico Madrid one time.

**4. Appendix Section**

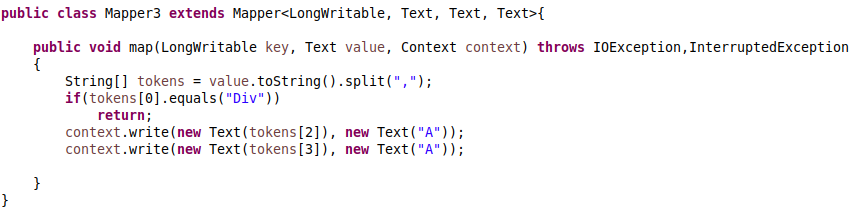
**4.1 Analysis One Source Code**

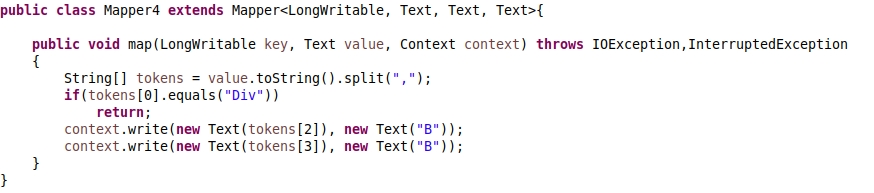
**4.1.1 Hadoop Source Code**

**Driver Code**

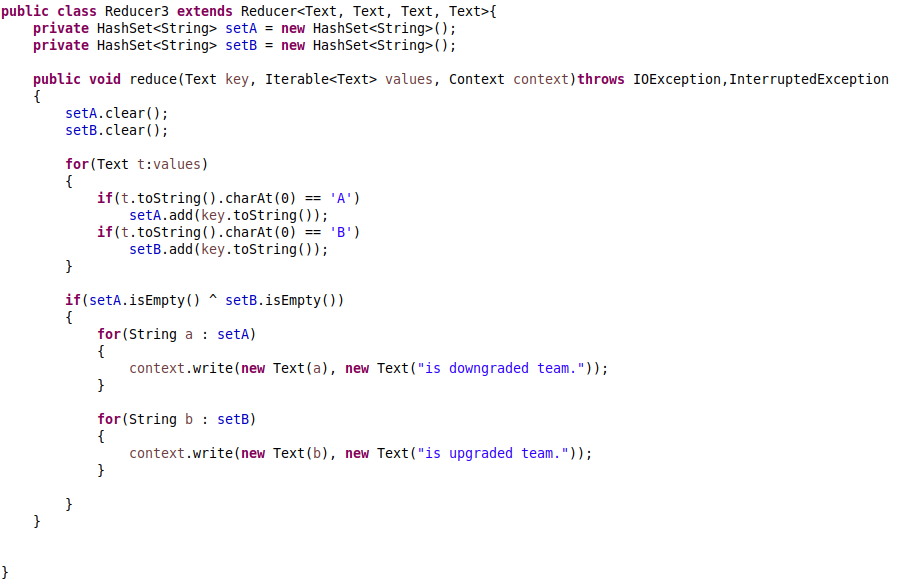
****

**Mapper Code**

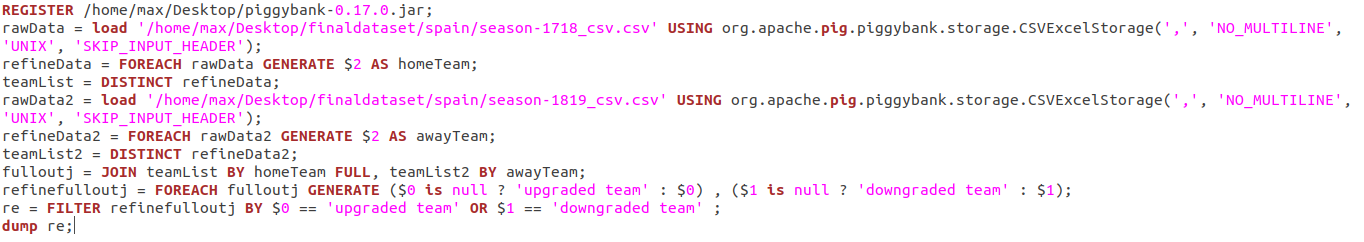
****

****

**Reducer Code**

****

**4.1.2 Pig Source Code**

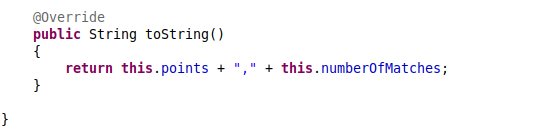
****

**4.2 Analysis Two Source Code**

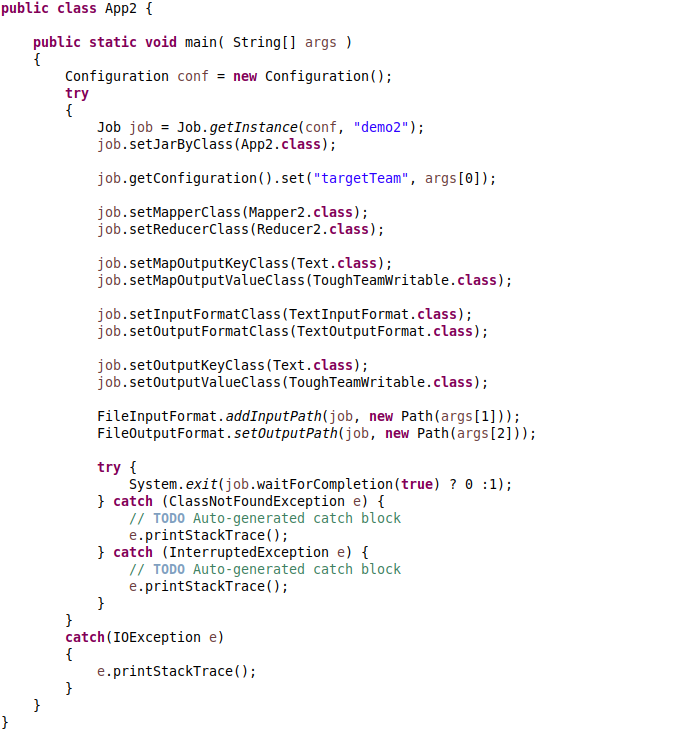
**4.2.1 Hadoop Source Code**

**Custom Writable Class Code**

****

****

**Driver Code**

****

**Mapper Code**

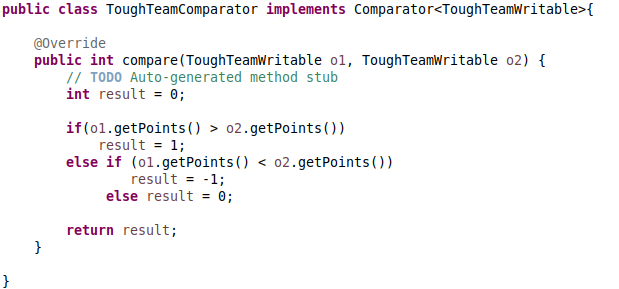
****

****

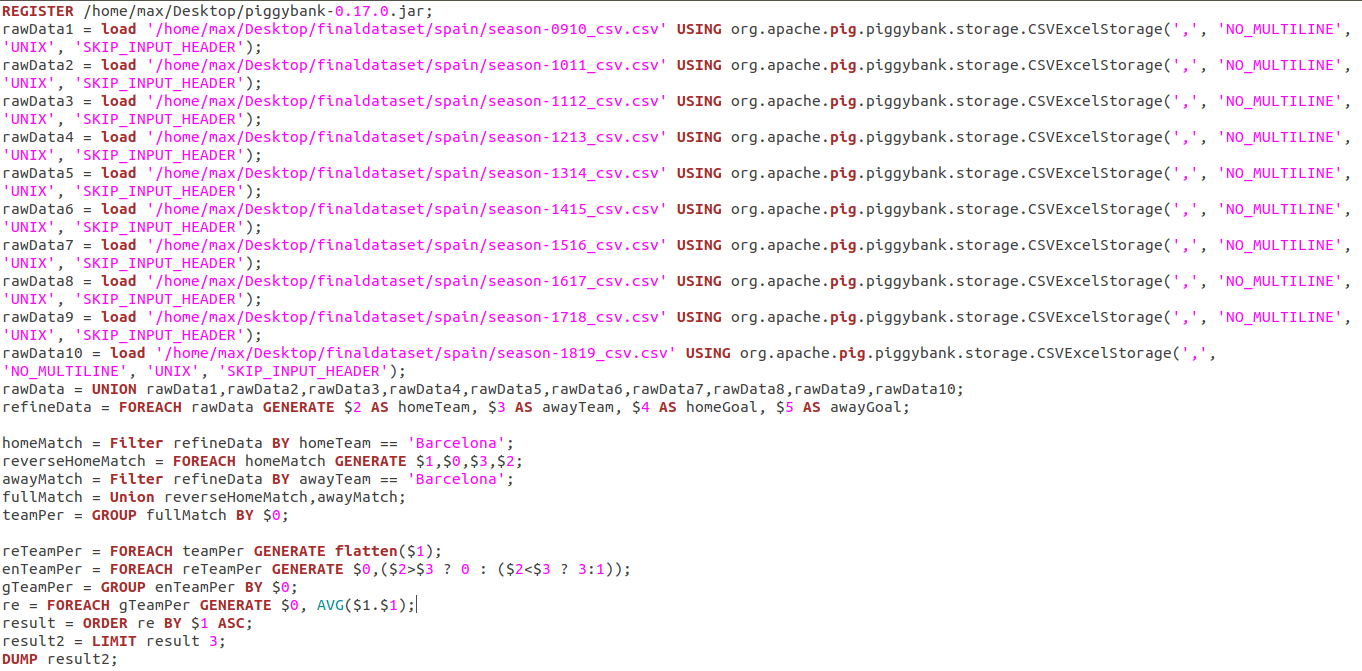
**Reducer Code**

****

**Comparator Code**

****

**4.2.2 Pig Source Code**

****

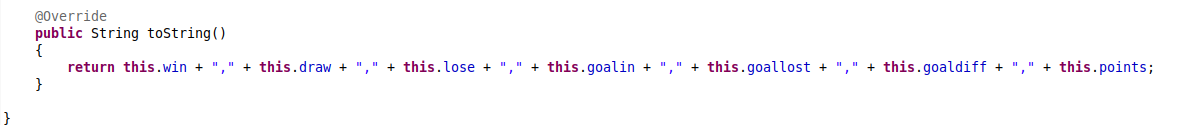
**4.3 Analysis Three Source Code**

**4.3.1 Hadoop Source Code**

**Custom Writable Class Code**

****

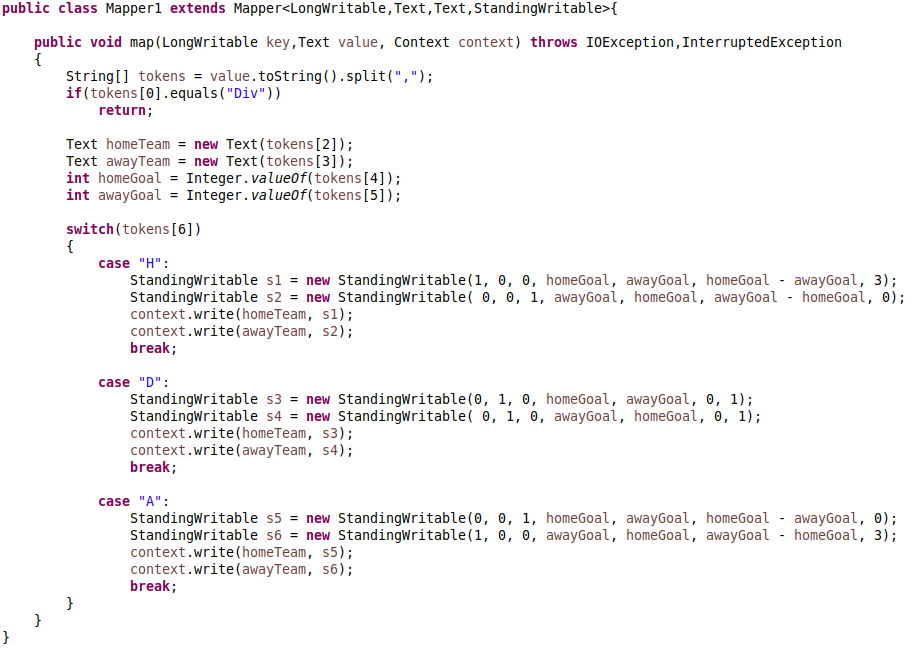
****

****

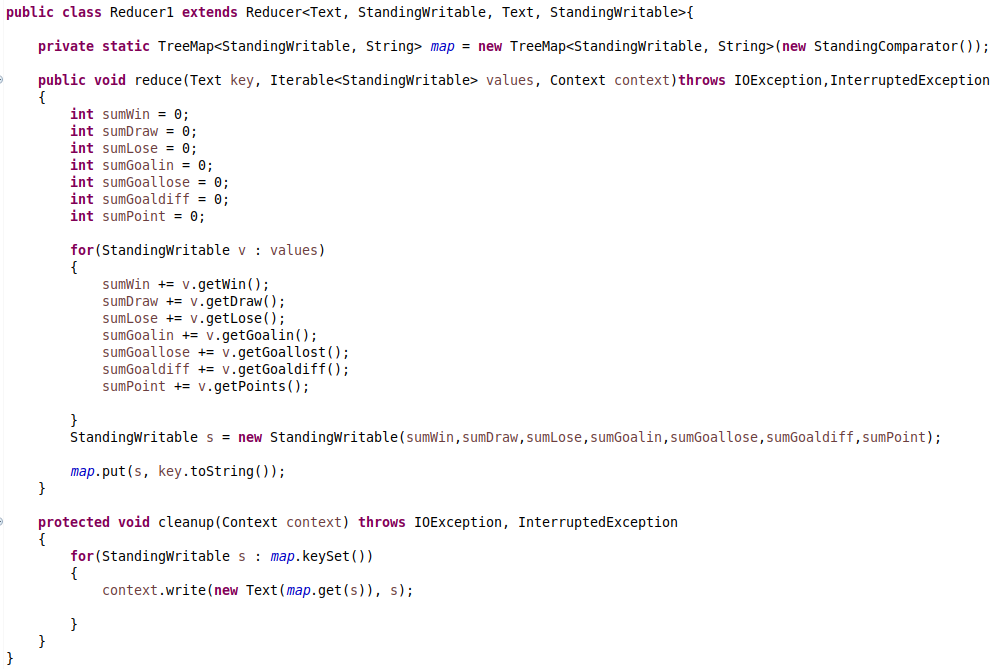
**Driver Code**



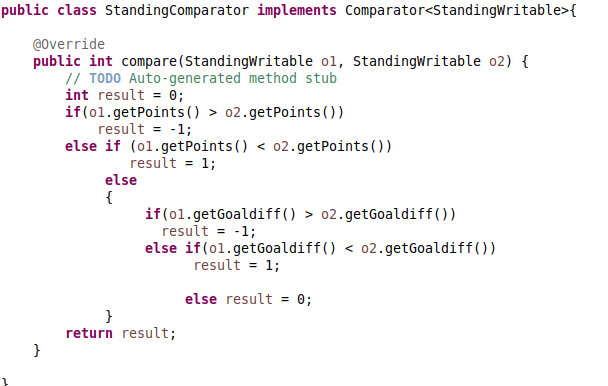
**Mapper Code**



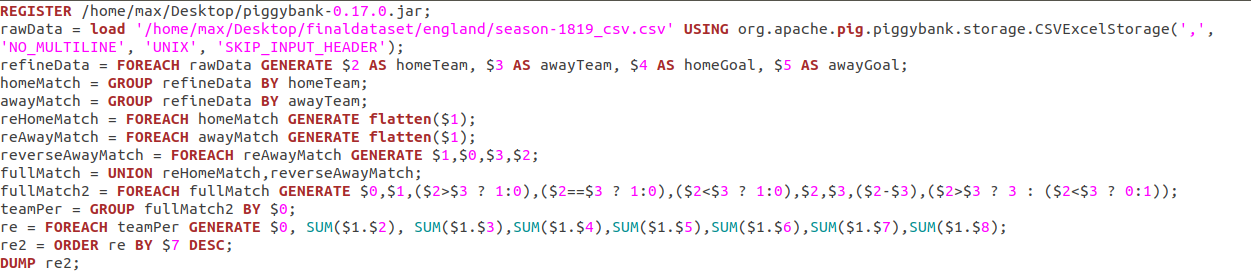
**Reducer Code**



**Comparator Code**

****

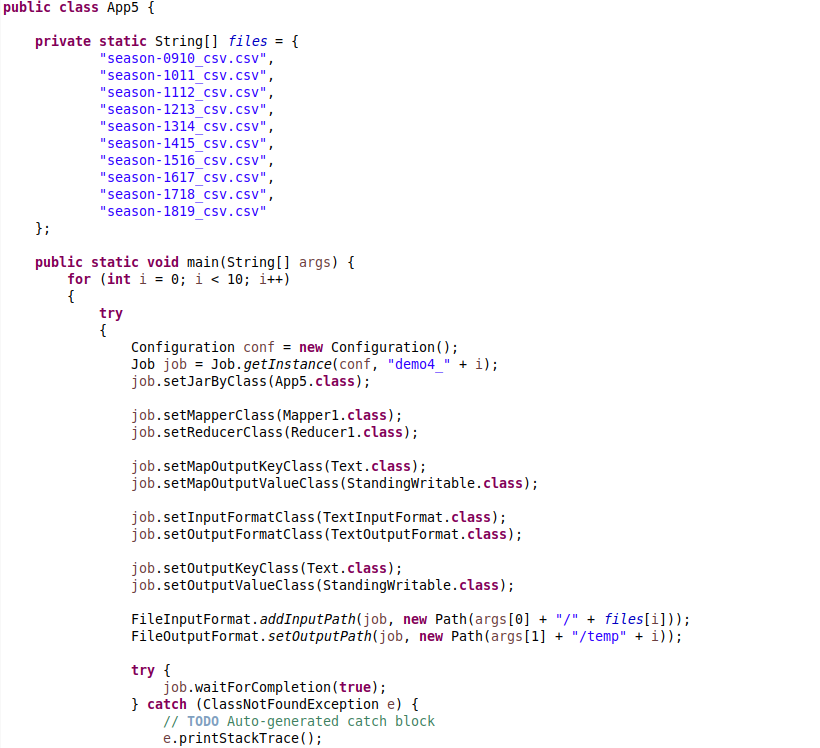
**4.3.2 Pig Source Code**

****

**4.4 Analysis Four Source Code**

**4.4.1 Hadoop Source Code**

**Driver Code**

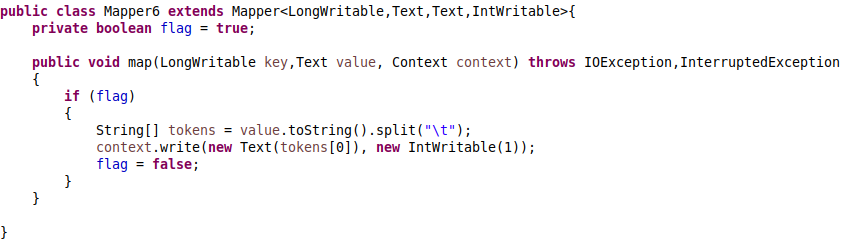
****

****

****

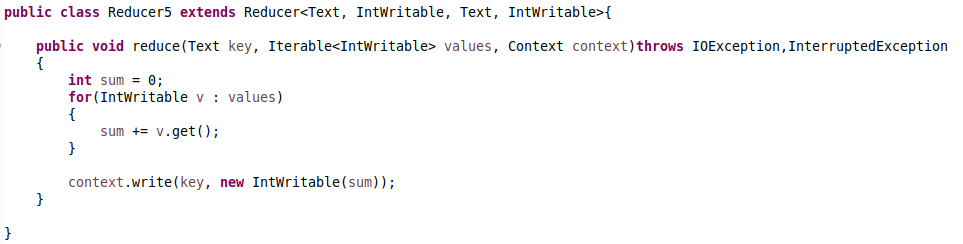
**Mapper Code**

Mapper1 class is reused here, so don’t put the snapshot. Only put the Mapper6 code.

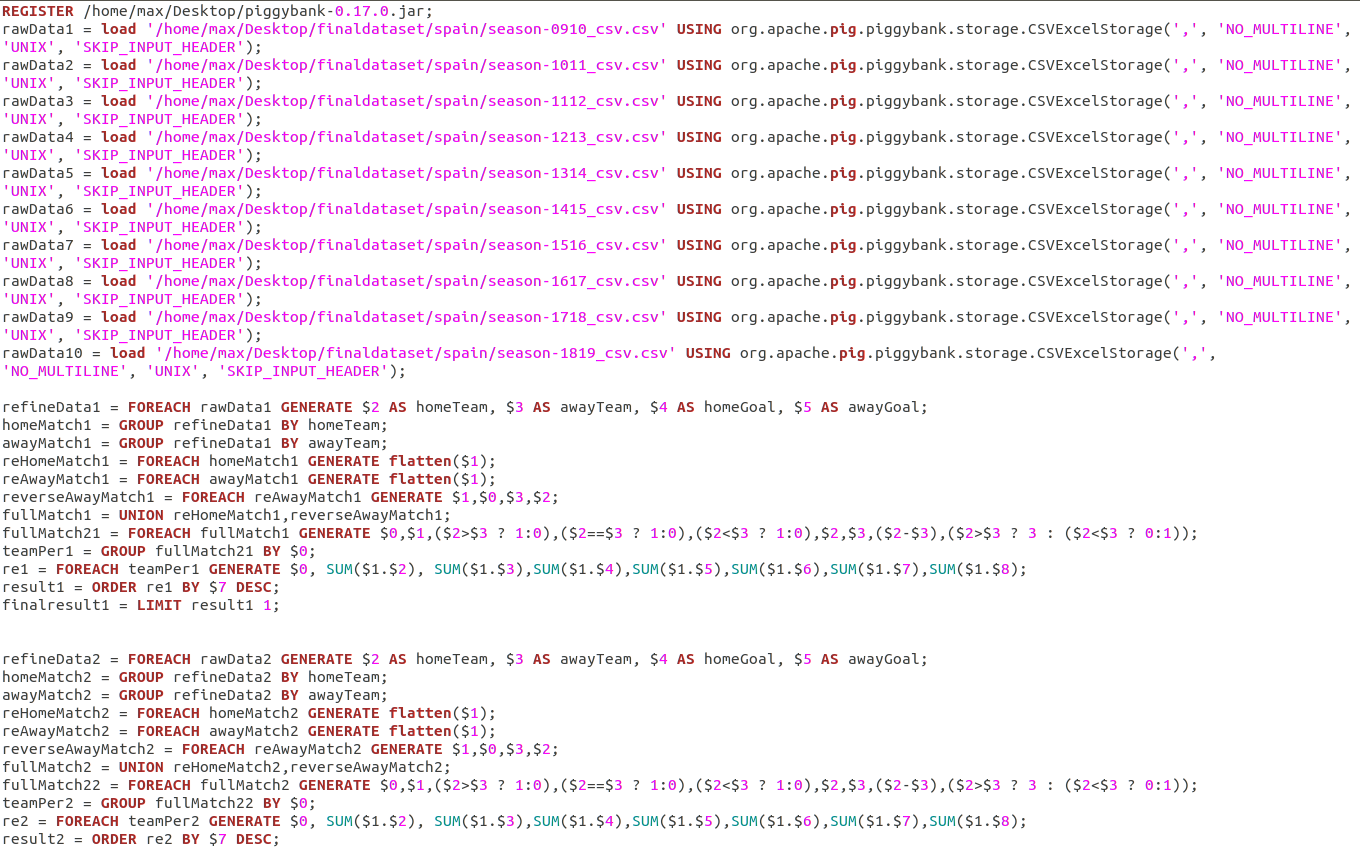
****

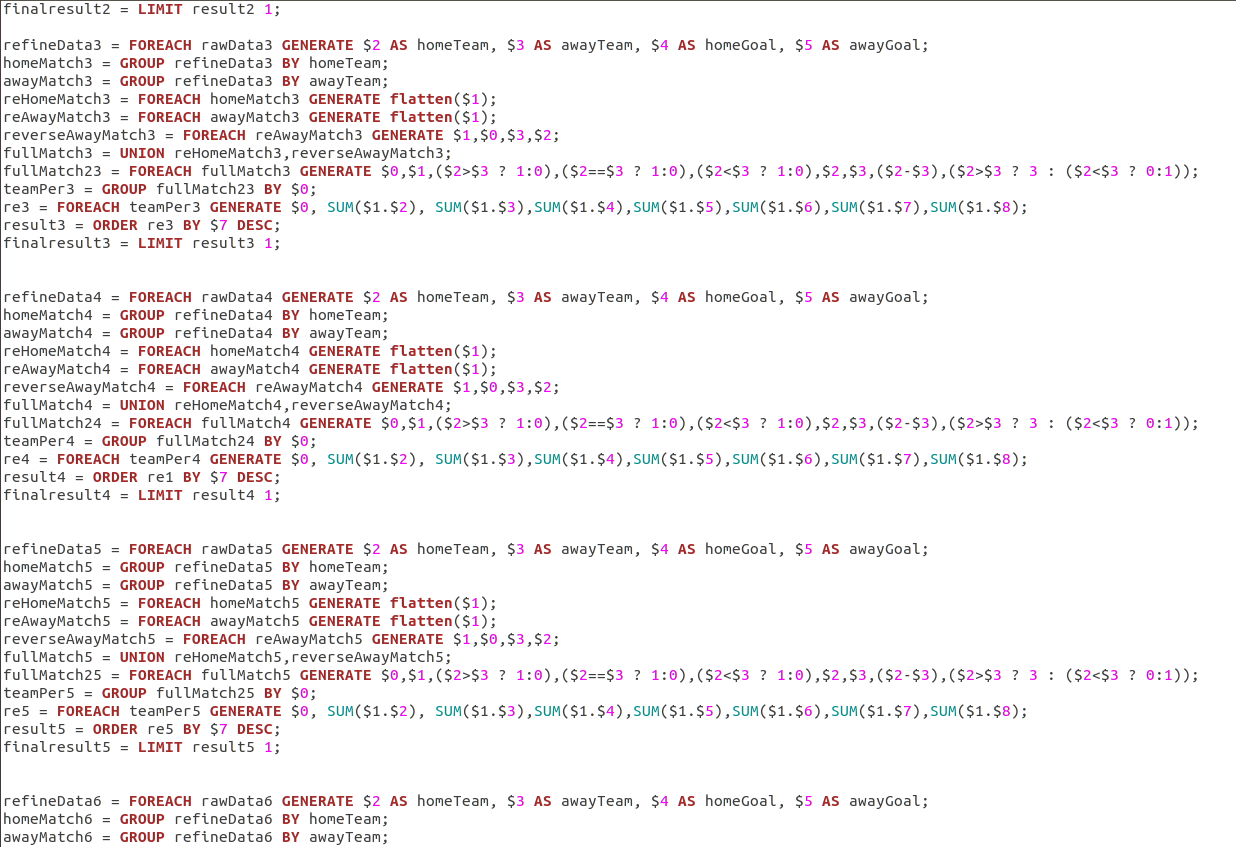
**Reducer Code**

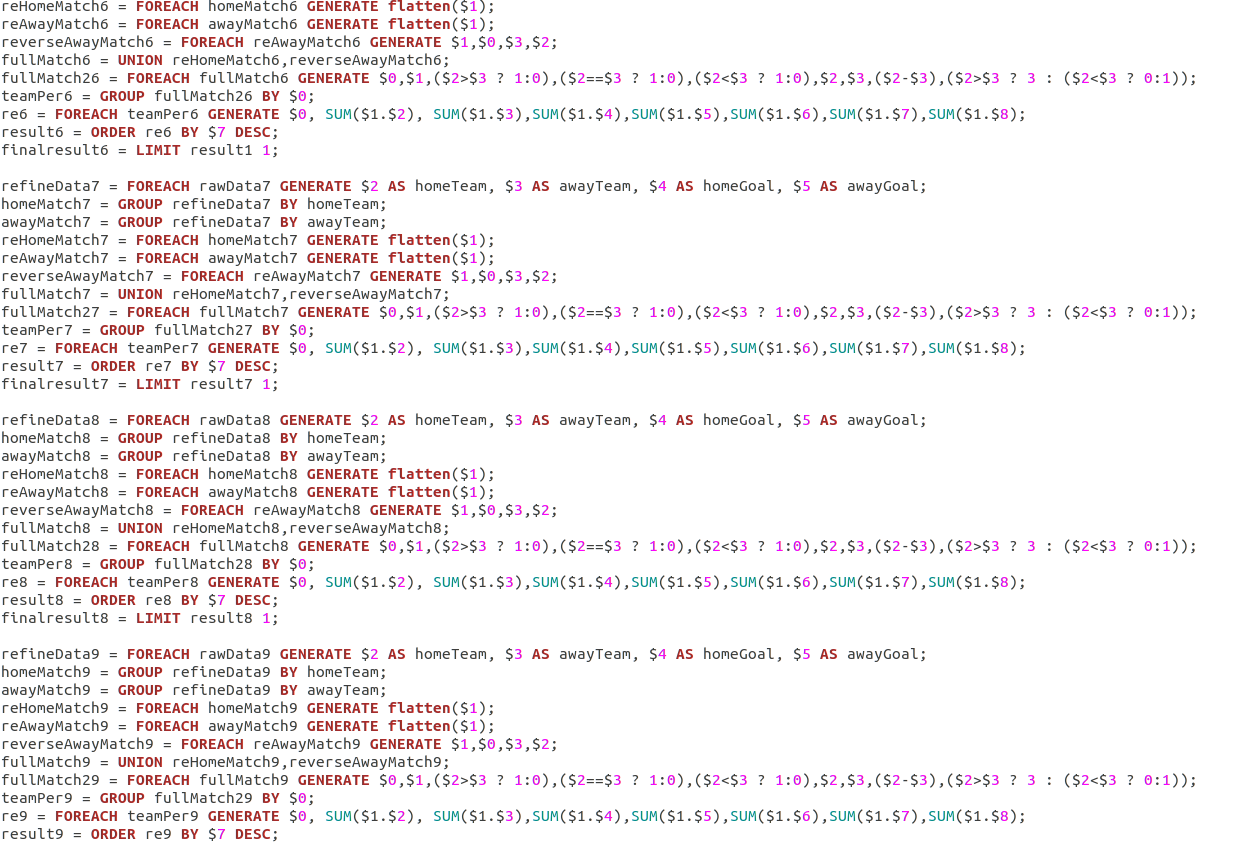
Reducer1 class is reused here, so don’t put the snapshot. Only put the Reducer5 code.

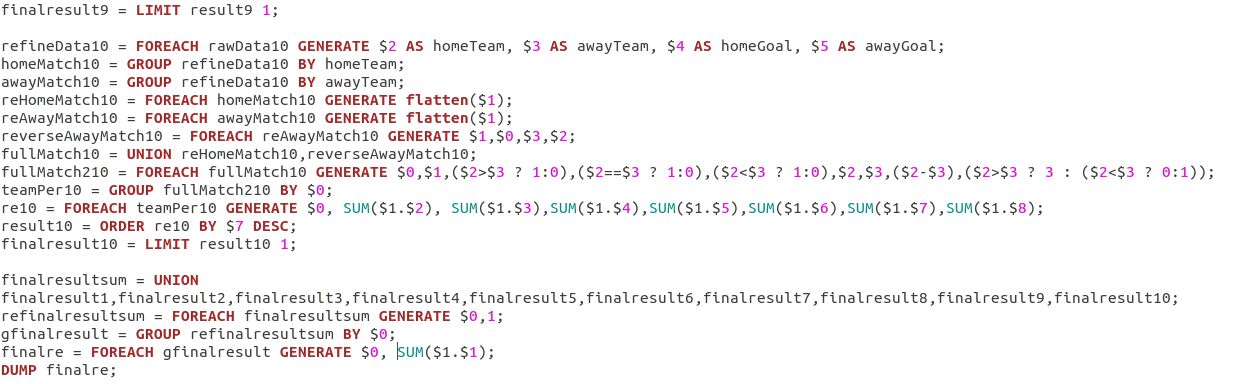
****

**4.4.2 Pig Source Code**

****

****

****

****