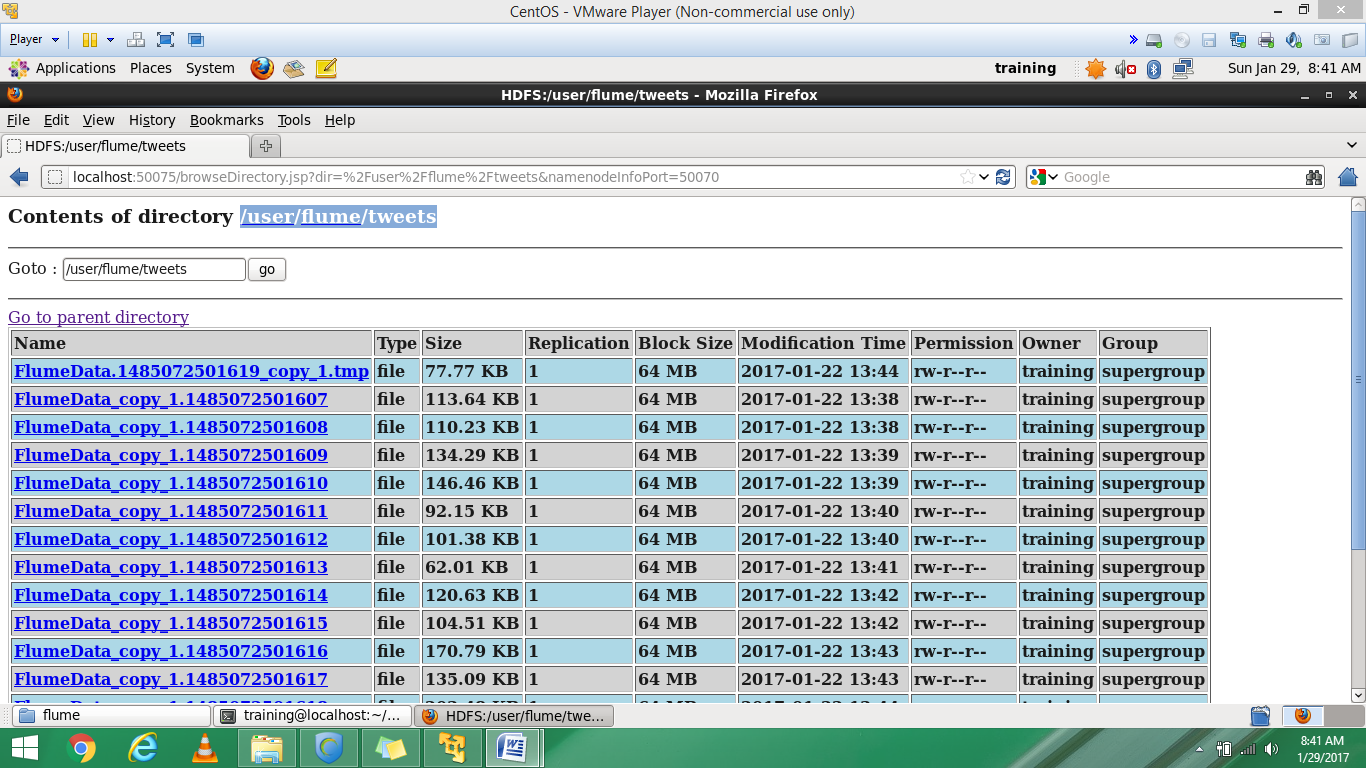
Determining twitter Popular #tags

[training@localhost Desktop]$ pig -x maprdeuce

* browse tweets files



The data from Twitter is in ‘Json’ format, so a Pig JsonLoader is required to load the data into Pig. You need to download the required jars for the JsonLoader from the below link:

[Pig JsonLoader Jar Files](https://drive.google.com/drive/u/0/folders/0ByJLBTmJojjzUXdEOUpVQ1haNG8)

Next, register the downloaded jars in Pig by using the following commands:

|  |  |
| --- | --- |
|  | REGISTER '/home/kiran/Desktop/elephant-bird-hadoop-compat-4.1.jar';    REGISTER '/home/kiran/Desktop/elephant-bird-pig-4.1.jar';    REGISTER '/home/kiran/Desktop/json-simple-1.1.1.jar'; |

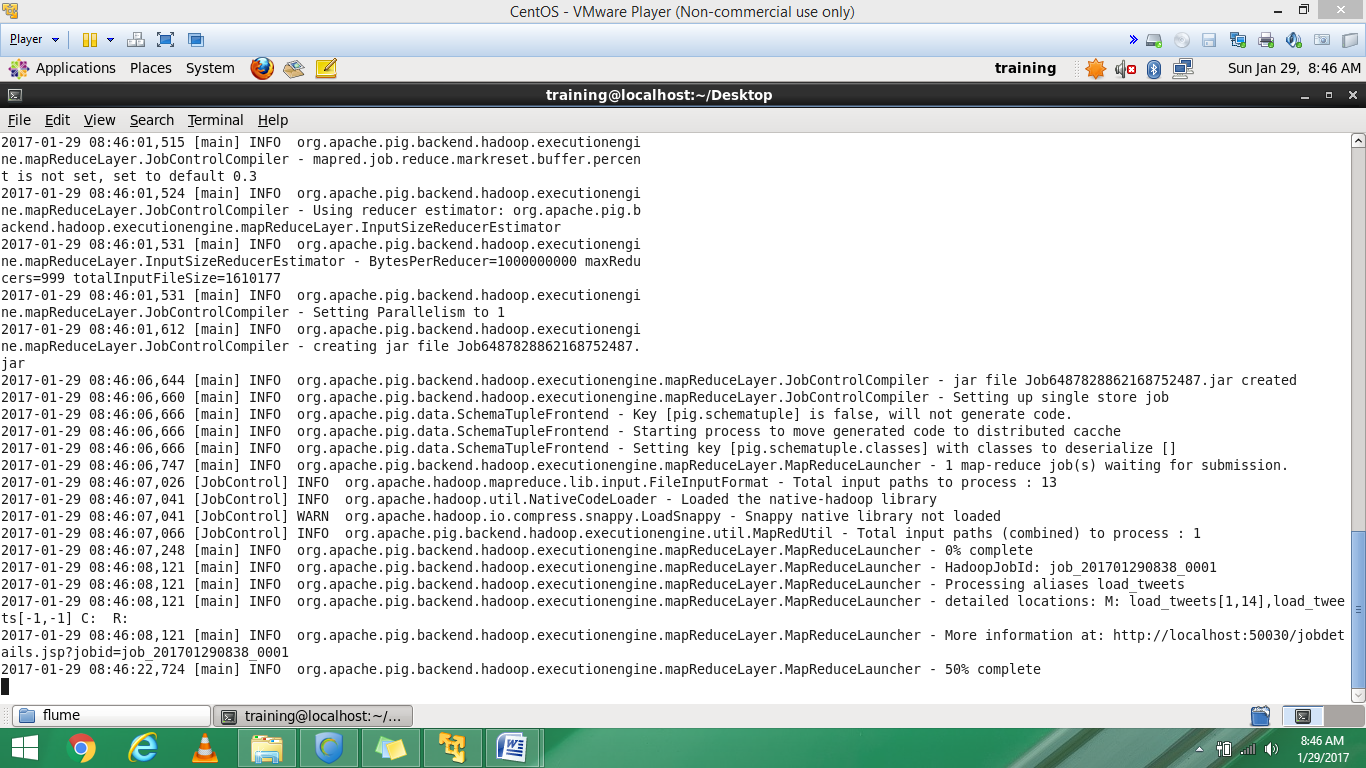
grunt> register '/home/training/Desktop/flume/elephant-bird-hadoop-compat-4.1.jar'

grunt> register '/home/training/Desktop/flume/elephant-bird-pig-4.4.jar'

grunt> register '/home/training/Desktop/flume/json-simple-1.1.1.jar'

* load the data

load\_tweets = LOAD '/user/flume/tweets/' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') AS myMap;



* extract entities and id

grunt> extract = foreach load\_tweets generate flatten(myMap#'entities') as (m:map[]),flatten(myMap#'id') as id;

In the tweet, the **hashtag**is present in the map object **entities**, which can be seen in thebelow image.

* extract hashtag and id

grunt> extract = foreach extract generate flatten(m#'hashtags') as (tags:map[]),id as id;

* extract text

|  |
| --- |
| MakkalMovement,823081363272253440)  (SaharaBirlaDiaries,823081378732457984)  (mirchiSeth,823081378732457984)  (ThankYouModiji,823081383262355456)  (tamilnadu,823081394071019520)  (jalikattu,823081394071019520)  (tamil,823081394071019520)  (modi,823081394071019520)  (LetterToPM,823081396931592193)  (CreditChor,823081407907958790)  (LetterToPM,823081030668013569)  (APDemandsSpecialStatus,823081036015923201)  (AmitShah,823081028440887296)  (Modi,823081028440887296)  (MostPowerful,823081028440887296)  (NaMo,823081028440887296)  (LEWIPS,823081028440887296)  (LetterToPM,823081045104984064)  (LetterToPM,823081045918683136)  (कभी\_सोचा\_है,823081051039866881)  (TheNewAfghanPolicy,823081093188427776)  (AmitShah,823081105582592000)  (Modi,823081105582592000)  (MostPowerful,823081105582592000)  (NaMo,823081105582592000)  (LEWIPS,823081105582592000)  (Breaking,823081146745556994)  (NarendraModi,823080257943023616)  (AkhileshYadav,823080257943023616)  (बीजेपी,823080297197699072)  (बीजेपी,823080297197699072)  (राहुल\_गाँधी,823080312406052864)  (Davos,823080334744915969)  (APDemandsSpecialStatus,823081186125824000)  (RailAccident,823081195365863424)  (HirakhandExpress,823081195365863424)  (APDemandsSpecialstatus,823081197362356224)  (SaharaBirlaDiaries,823081199434203136)  (mirchiSeth,823081199434203136)  (LetterToPM,823081219114012672)  (UPElection2017,823081245026516992)  (LetterToPM,823081248570667008)  (LetterToPM,823081272801132544)  (APDemandsSpecialStatus,823080105765322753)  (Modi,823080109053739008)  (Andhra,823080109053739008)  (Modi,823080109158629376)  (Andhra,823080109158629376)  (विभीषण,823080117203181569)  (पृथ्वीराज,823080117203181569)  (LetterToPM,823080152804499457)  (LetterToPM,823080156029927424)  (jallikattu,823080166285017089)  (AntiDalit,823080211327647744)  (LetterToPM,823080225315516417)  (LetterToPM,823080797985001472)  (बीजेपी,823080864300961793)  (बीजेपी,823080864300961793)  (uttarpradeshelections2017,823080892474281984)  (AntiDalit,823080902540595200)  (CreditChor,823079866698518528)  (APDemandsSpecialStatus,823079879469977600)  (LetterToPM,823079899808210944)  (AkhileshYadav,823079918992920577)  (ModiBhakts,823080005542547456)  (TransformingIndia,823080005542547456)  (BJPMission2019,823080005542547456)  (ModiBhakts,823080006947667969)  (TransformingIndia,823080006947667969)  (BJPMission2019,823080006947667969)  (Davos,823080067831975936)  (RTI,823080082071752704)  (RSS,823080082071752704)  (RSSBJP,823080082071752704)  (LetterToPM,823080903266025473)  (SaharaBirlaDiaries,823080910379687937)  (mirchiSeth,823080910379687937)  (LetterToPM,823080927677054976)  (APDemandsSpecialstatus,823081019360366593)  (IslamophobiaIs,823081023445565440)  (shia,823081023445565440)  (LetterToPM,823080506594099200)  (justiceforjallikatu,823080567163887616)  (jallikattu,823080567163887616)  (Modi,823080609866219520)  (TheLegendOfNarendraModi,823080609866219520)  (Modi,823080612676386816)  (LetterToPM,823080379649261569)  (CreditChor,823080401505812480)  (Modi,823080408891977728)  (OPS,823080408891977728)  (Chinnamma,823080408891977728)  (SaharaBirlaDiaries,823080414948560896)  (mirchiSeth,823080414948560896)  (SaharaBirlaDiaries,823080422116446209)  (mirchiSeth,823080422116446209)  (Davos,823081428183355392)  (झाड़ू\_वाला\_जितुगा,823081509393485824)  (SanghiTruth,823081509393485824)  (NarendraModi,823080654820831232)  (PMModi,823080654820831232)  (LetterToPM,823080690359169024)  (LetterToPM,823080745673588737)  (LetterToPM,823080746072109056) |

grunt> txt = foreach extract generate flatten(tags#'text') as txt ,id as id;

1. count the no of #tags

grunt> grp = group txt by txt;

grunt> cnt = foreach grp generate group as hashtag\_text,COUNT(txt.txt) as hashtag\_count;

|  |
| --- |
| (OPS,1)  (RSS,1)  (RTI,1)  (Modi,11)  (NaMo,2)  (live,1)  (modi,1)  (shia,1)  (Davos,3)  (enjoy,1)  (india,1)  (tamil,1)  (Andhra,2)  (LEWIPS,2)  (PMModi,1)  (RSSBJP,1)  (AmitShah,2)  (Breaking,1)  (AntiDalit,2)  (Chinnamma,1)  (adhocband,1)  (jalikattu,1)  (tamilnadu,1)  (CreditChor,3)  (LetterToPM,21)  (ModiBhakts,2)  (jallikattu,3)  (mirchiSeth,5)  (SanghiTruth,1)  (MostPowerful,2)  (NarendraModi,2)  (RailAccident,1)  (AkhileshYadav,2)  (BJPMission2019,2)  (IslamophobiaIs,1)  (MakkalMovement,1)  (ThankYouModiji,1)  (UPElection2017,1)  (HirakhandExpress,1)  (TransformingIndia,2)  (SaharaBirlaDiaries,5)  (TheNewAfghanPolicy,1)  (बीजेपी,4)  (विभीषण,1)  (justiceforjallikatu,1)  (APDemandsSpecialStatus,4)  (APDemandsSpecialstatus,2)  (TheLegendOfNarendraModi,1)  (uttarpradeshelections2017,1) |