Northeastern University

Final Project Report

Topic: Yelp Dataset Analysis

Engineering of Big Data
Academic Year: Spring 2021

Professor: Mr. Yusuf Ozbek
TA: Deepak Gopalan

Submitted by: Poojitha Muppalla

Summary:

Yelp Dataset Analysis is the Dataset that I wanted to use for project. Dataset is downloaded from the Kaggle Website which can be found at https://www.kaggle.com/ksjpswaroop/yelp-data-analysis/data

The dataset has millions of rows. The dataset has 5 csv files namely yelp_review.json, yelp_business.json, yelp_user.json, yelp_tip.json and yelp_checking.json. In which I will be working on 3 namely yelp_review.json, yelp_business.json and yelp_user.json.

About the Dataset:

- **Yelp_review**

  Attribute used:
  1. Review Id
  2. User ID
  3. Business ID
  4. Review
  5. Review Date
  6. Review Comment

- **Yelp_business**

  Attributes Used:
  1. Business_id
  2. Business Name
  3. City
  4. State
  5. Postal
  6. Rating

- **Yelp_user**

  Attribute Used:
  1. User Id
  2. Name
  3. Year Joined

I have worked upon the below analysis for the project:

S.No	Analysis Description	Implementation Details
1	Number of business in a city	**MongoDB**
2	To find which user has posted reviews for businesses	Apache Pig – **Replicated Join**
3	To find details of Business that have user reviews less than 100.	Apache **Pig**
4	To find top 3 business based on rating for each city.	Apache **Hive**
5	To find list of businesses that have above average rating in each state	Apache Hive
6	Partitioning on the basis of year user joined the yelp	MapReduce – **Data Organization** Partitioning
7	Calculate of no. of users signed up on yelp each year	MapReduce – **Chaining Map Reduce Summarization**
8	Users who have yelp account but never provided any reviews yet	**MapReduce – Joins**
9	To find the top 25 most reviewed business by users	MapReduce – **Filtering Techniques** Top n Filtering Pattern
10	To find out how well a business is been rated by users per year	MapReduce – **Secondary Sort**
11	Mahout Recommendation	Data Cleaning + **Mahout Recommendation**
12	Output Visualization	**Tableau**

Getting Started

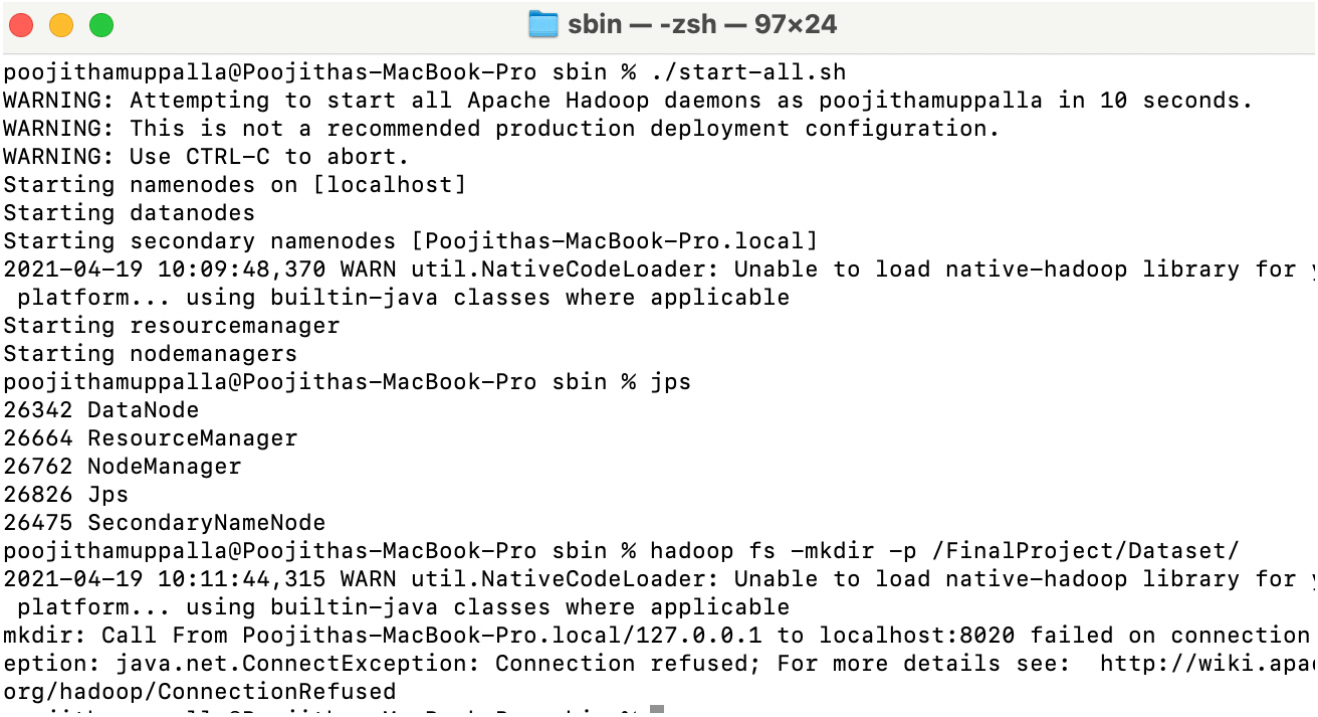
STARTING HADOOP

1. Navigate to Hadoop Directory:
   cd /usr/local/cellar/hadoop/3.3.0/sbin

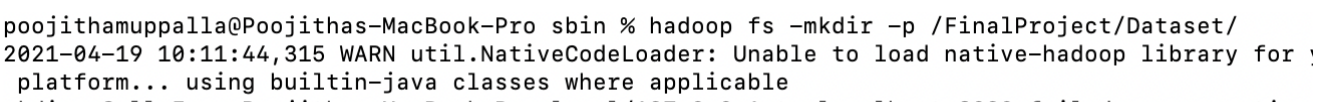
2. Start Hadoop daemons
   ./start-all.sh

3. Check Hadoop Daemons
   jps

4. Navigate to Hadoop UI localhost:9870/



COPYING FILE FROM LOCAL FILE SYSTEM TO HDFS



MONGODB ANALYSIS:

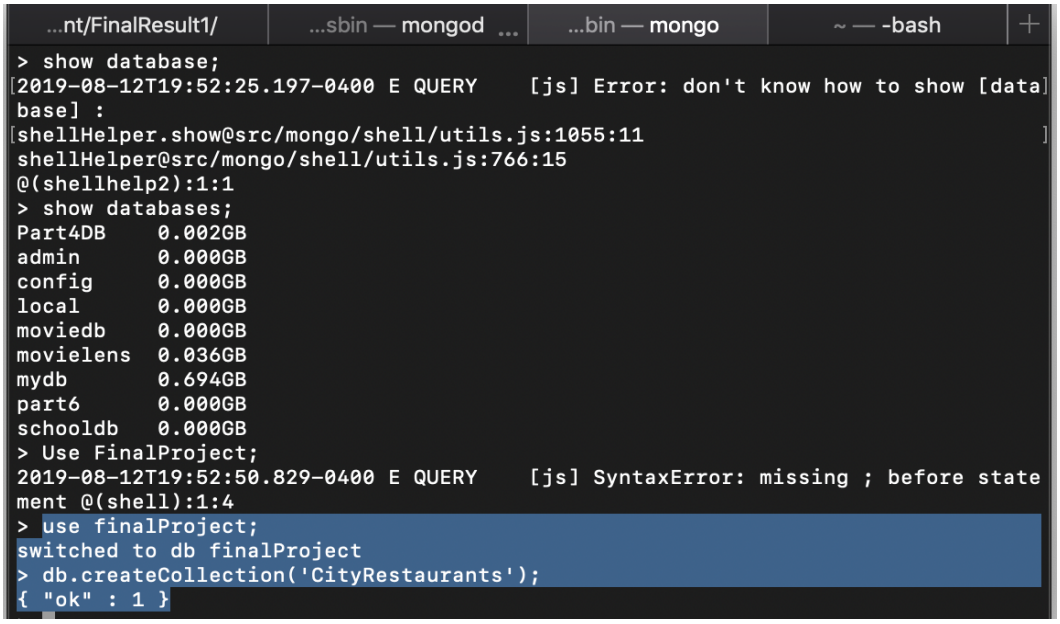
Analysis # 01: To find the count of number of business per city:

1. Create Mongo Database:

> use finalProject;
switched to db finalProject

2. Create Collection

> db.createCollection('CityRestaurants');
{ "ok" : 1 }



3. Import csv file to MongoDB:

mongoimport --db finalProject --collection CityRestaurants --type csv --file /Users/poojithamuppalla/Downloads/Project/Dataset/yelp_business.csv -headerline;

4. Write Map & Reduce Functions to perform Analysis:

```
var map= function (){ var key =
this.city; var value = {count:1};
emit(key, value);
} var reduce= function (key, values){ var cnt =
{count:0}; values.forEach(function(val){
cnt.count += parseInt(val.count);
});
return cnt;
```

```
{ "_id" : "Allegheny", "value" : { "count" : 1 } }
{ "_id" : "Allison Park", "value" : { "count" : 81 } }
{ "_id" : "Ambridge", "value" : { "count" : 36 } }
{ "_id" : "Amherst", "value" : { "count" : 91 } }
{ "_id" : "Anjou", "value" : { "count" : 14 } }
{ "_id" : "Ansnorveldt", "value" : { "count" : 1 } }
{ "_id" : "Anthem", "value" : { "count" : 87 } }
Type "it" for more
> edit map
> map
function (){
        var key = this.city;
        var value = {count:1};
        emit(key, value);
}
> reduce
function (key, values){
        var cnt = {count:0};
        values.forEach(function(val){
                cnt.count += parseInt(val.count);
        });
        return cnt;
}
>
```

5. Execute MapReduce Job:

db.CityRestaurants.mapReduce(map, reduce, {out:"restCity2"});

6. Get results:

db.restCity2.find().pretty()

```
...nt/FinalResult1/ | ...sbin — mongod ... | ...bin — mongo | ~ — -bash
}
> db.restCity2.find().pretty()
{ "_id" : "", "value" : { "count" : 1 } }
{ "_id" : "110 Las Vegas", "value" : { "count" : 1 } }
{ "_id" : "AGINCOURT", "value" : { "count" : 1 } }
{ "_id" : "Aberdour", "value" : { "count" : 1 } }
{ "_id" : "Aberlady", "value" : { "count" : 2 } }
{ "_id" : "Ahwahtukee", "value" : { "count" : 1 } }
{ "_id" : "Ahwatukee", "value" : { "count" : 16 } }
{ "_id" : "Ahwatukee Foothills Village", "value" : { "count" : 1 } }
{ "_id" : "Aichwald", "value" : { "count" : 2 } }
{ "_id" : "Ajax", "value" : { "count" : 252 } }
{ "_id" : "Alburg", "value" : { "count" : 1 } }
{ "_id" : "Alburgh", "value" : { "count" : 1 } }
{ "_id" : "Alex", "value" : { "count" : 1 } }
{ "_id" : "Allegheny", "value" : { "count" : 1 } }
{ "_id" : "Allison Park", "value" : { "count" : 81 } }
{ "_id" : "Ambridge", "value" : { "count" : 36 } }
{ "_id" : "Amherst", "value" : { "count" : 91 } }
{ "_id" : "Anjou", "value" : { "count" : 14 } }
{ "_id" : "Ansnorveldt", "value" : { "count" : 1 } }
{ "_id" : "Anthem", "value" : { "count" : 87 } }
Type "it" for more
>
```

APACHE PIG ANALYSIS

Analysis # 02: To implement Replicated join to show which user has posted what reviews against the business

Performing replicated join since the reviews dataset is very large, between reviews and users.

1. **LOAD the review dataset from HDFS Using CSVExcelStorage and skip the header**

```
reviews= LOAD
'hdfs://localhost:9000/FinalProject/Dataset/yelp_review.csv'                    USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',', 'NO_MULTILINE',
'UNIX', 'SKIP_INPUT_HEADER');
```

2. **LOAD the User dataset from HDFS Using CSVExcelStorage and skip the header**

```
LOAD='hdfs://localhost:9000/FinalProject/Dataset/yel
p_user.csv'                                    USING
org.apache.pig.piggybank.storage.CSVExcelStorage(',',
'NO_MULTILINE','UNIX',
'SKIP_INPUT_HEADER');
describe users;
```

3. **Filtering the reviews & users file**

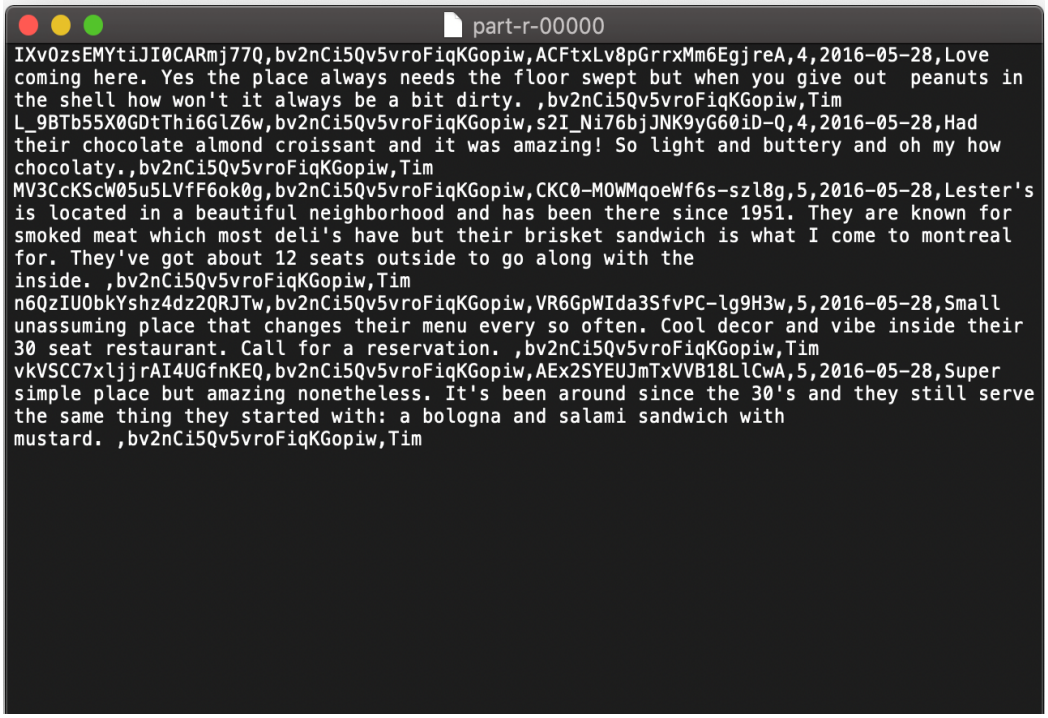
```
fltrvw = FOREACH reviews GENERATE (chararray) $0 as reviewId, (chararray) $1 as userId,
(chararray) $2 as businessID, (chararray) $3 as rating, (chararray) $4 as date, (chararray) $5 as
userReview;

describe fltrvw;

fltusr = FOREACH users GENERATE (chararray) $0 as userId,(chararray) $1 as userName;

describe fltusr;
```

4. **Performing Join operations:**

```
joined = Join fltrvw by userId, fltusr by userId USING 'replicated';

lmtjnd = LIMIT joined 5;
```

5.Storing the output file to HDFS

```
Store lmtjnd into 'hdfs://localhost:9000/FinalProject/pig_join' Using  PIGSTORAGE (',');
```



Analysis # 03: To find the business details that have been reviewed less than 100 times

1. LOAD the review dataset from HDFS

reviews = LOAD 'hdfs://localhost:9000/FinalProject/Dataset/yelp_review.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(‘,’, 'NO_MULTILINE', 'UNIX', 'SKIP_INPUT_HEADER');

2. Filtering the review dataset

fltrvw = FOREACH reviews GENERATE (chararray) $0 as reviewId, (chararray) $1 as userId, (chararray) $2 as businessID, (chararray) $3 as rating, (chararray) $4 as date, (chararray) $5 as userReview;

2. LOAD the buiness dataset from HDFS

business = LOAD 'hdfs://localhost:9000/FinalProject/Dataset/yelp_business.csv' USING org.apache.pig.piggybank.storage.CSVExcelStorage(‘,’, 'NO_MULTILINE', 'UNIX', 'SKIP_INPUT_HEADER');

4. Filtering the business dataset from HDFS

fltbus = FOREACH reviews GENERATE (chararray) $0 as businessId, (chararray) $1 as businessName, (chararray) $4 as city;

5. Grouping the filtered review by businessID

groupedByBussId = group fltrvw by businessID;

counted = FOREACH groupedByBussId GENERATE group as businessID,

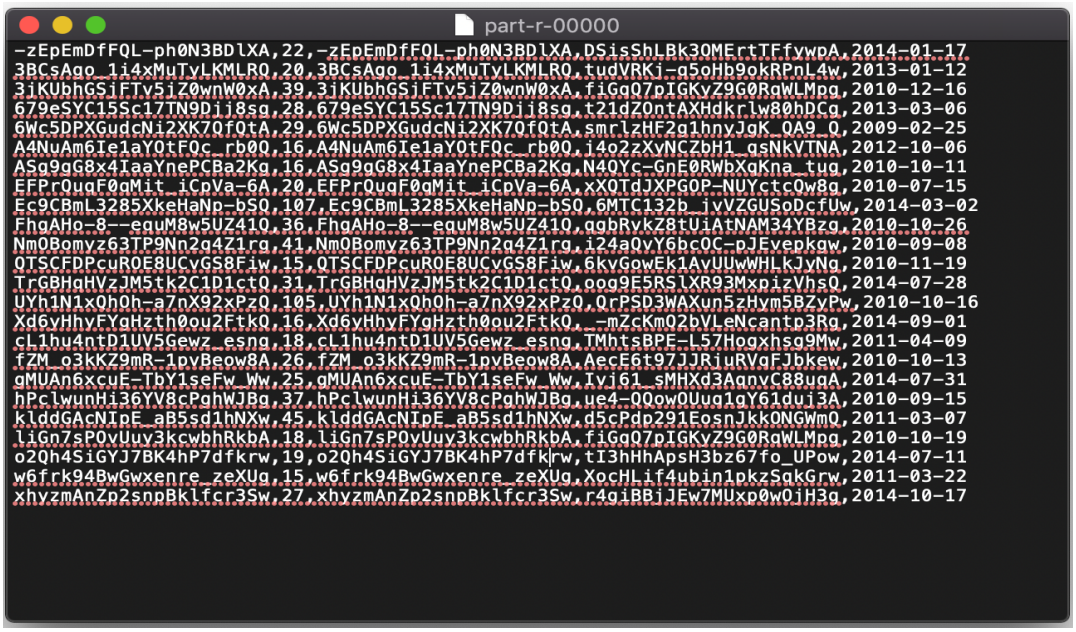
COUNT(fltrvw) as cnt;

res = FOREACH counted GENERATE businessID, cnt;

6. **Filtering the filtered review data**
result = FILTER res BY cnt <100;

7. Joining the filtered review and filtered business
joined = JOIN result by $0, fltbus by businessId;

8. Storing the output file to HDFS
STORE joined INTO '/Users/poojithamuppalla/Desktop/output.txt' Using PigStorage(',');



APACHE HIVE ANALYSIS

Analysis # 04 : To find Top 3 business based on rating for each city.

1. Create Table 'business_all'

Create table business_all(business_id String,name String,neighborhood String,address String,city String,state String,postal_code String ,latitude String,longitude String,stars Float ,review_count int,is_open BOOLEAN,categories String);

2. Load csv Data

Load Data inpath
'hdfs://localhost:9820/FinalProject/Dataset/yelp_business.csv' into table business_all;

3. Create business Table

Create Table business as Select business_id, name, city, state, Cast(stars as Double) As Ratings from business_all;

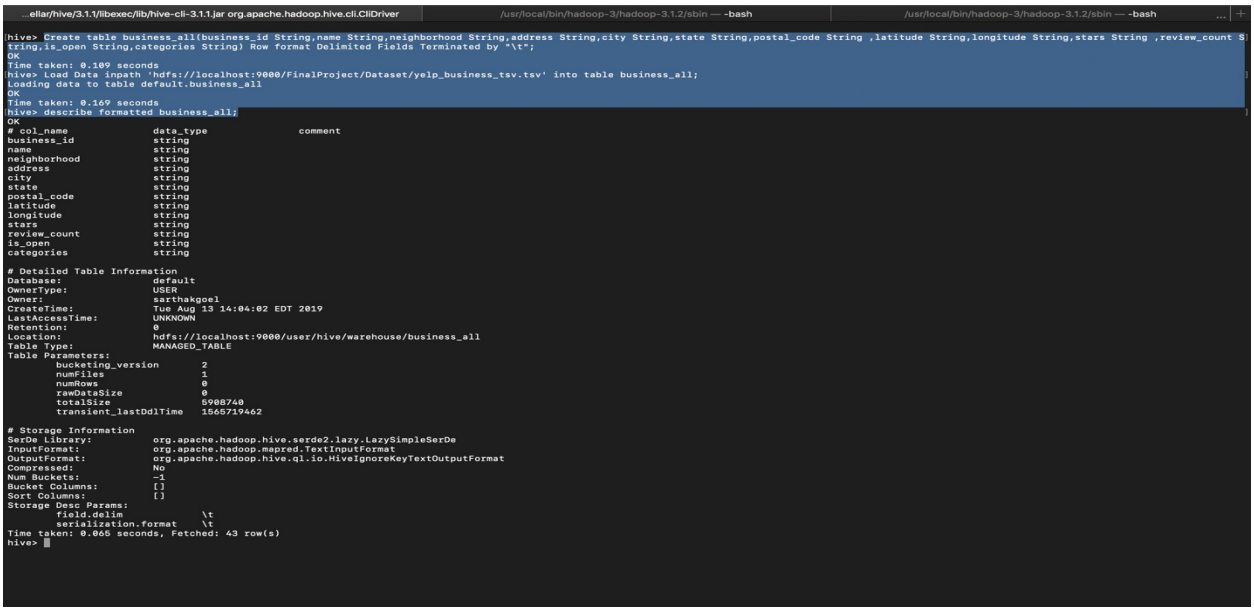
4. Top 3 business based on rating for each city:

Select * from(Select business_id, name, city, state, ratings, rank() over (partition by city order by ratings desc) as rank from business) b where rank <3 limit 10;

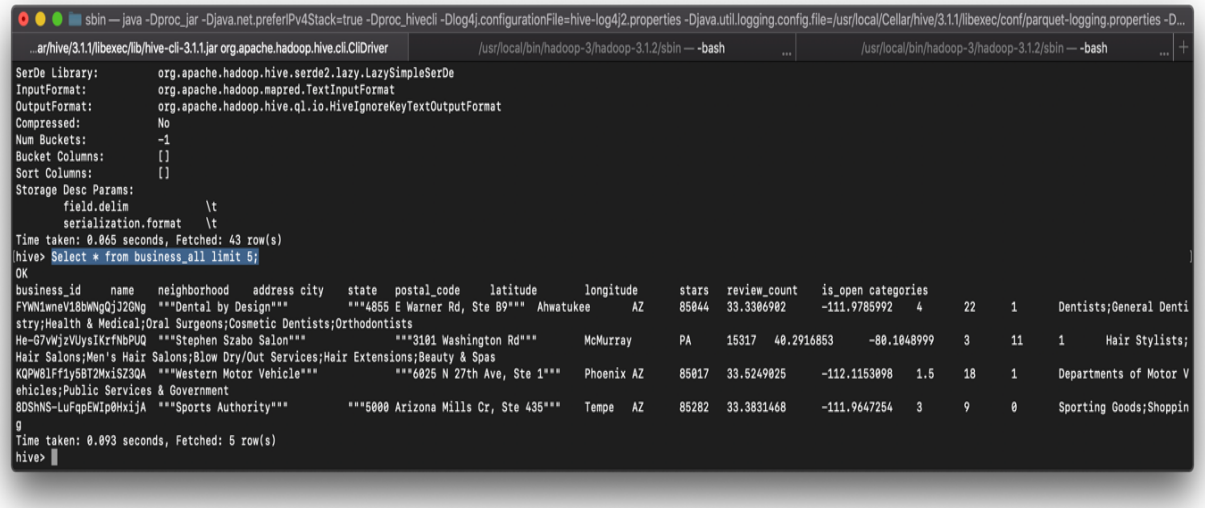
5. Inserting the Hive Output to HDFS:

Insert Overwrite Directory 'hdfs://localhost:9000/FinalProject/Hive/Analysis3/' Select * from(Select business_id, name, city, state, ratings, rank() over (partition by city order by ratings desc) as rank from business) b where rank <3;

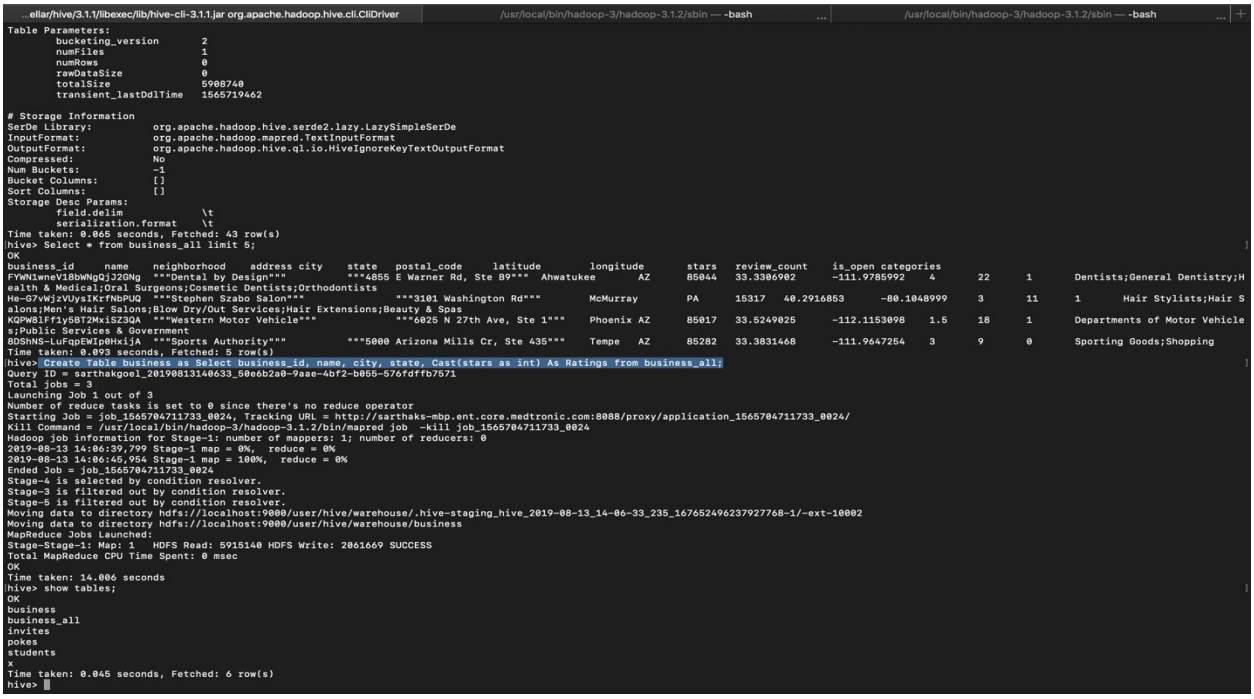
1. **Create Table 'business_all'**



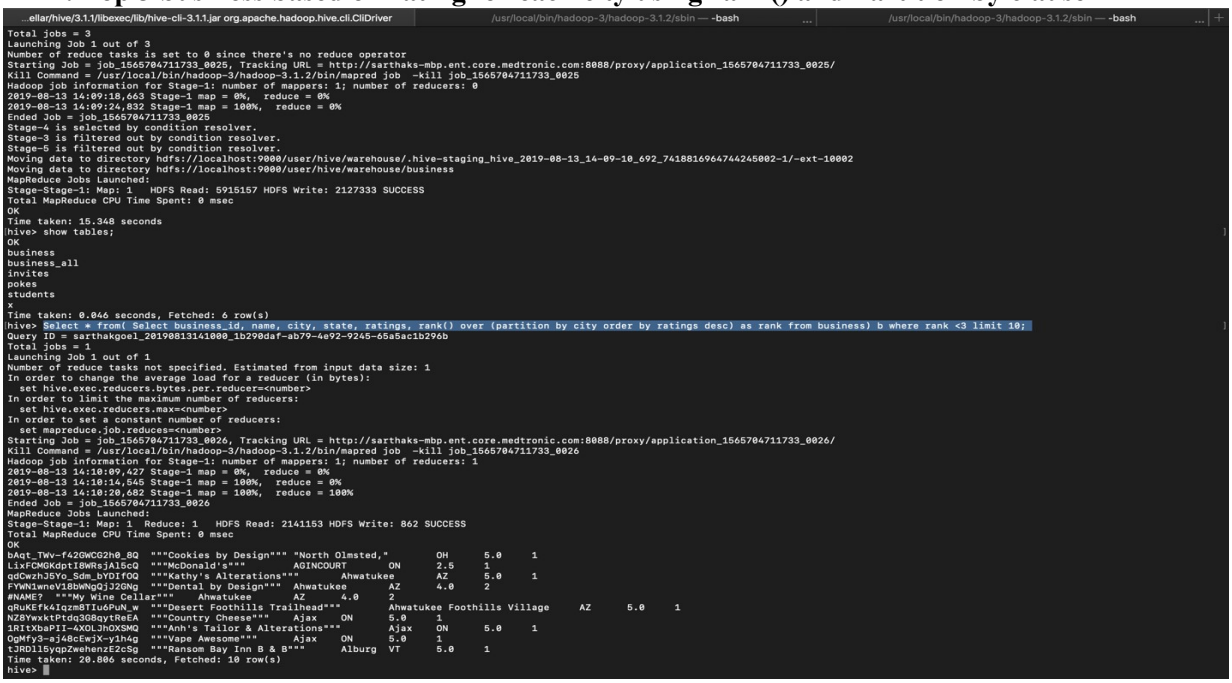
2. Load tsv Data into business_all table that you just created



3. Create business Table



4. Top 3 business based on rating for each city using rank() and Partition by clause



5. Inserting the Hive Output to HDFS using Insert Overwrite Directory command:

```
...pache.hadoop.hive.cli.CliDriver   ...3/hadoop-3.1.2/sbin — -bash ...   ...adoop-3.1.2/sbin — -bash ...
hive> Insert Overwrite Directory 'hdfs://localhost:9000/FinalProject/Hive/Analysis3/' Select *
 from( Select business_id, name, city, state, ratings, rank() over (partition by city order by
 ratings desc) as rank from business) b where rank <3;
Query ID = sarthakgoel_20190813161717_3c1fb46f-697a-4cdc-96db-8a91e3d11117
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
```

```
roxy/application_1565726254196_0003/
Kill Command = /usr/local/bin/hadoop-3/hadoop-3.1.2/bin/mapred job  -kill job_1565726254196_00
03
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-08-13 16:17:26,505 Stage-1 map = 0%,  reduce = 0%
2019-08-13 16:17:32,714 Stage-1 map = 100%,  reduce = 0%
2019-08-13 16:17:39,903 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_1565726254196_0003
Moving data to directory hdfs://localhost:9000/FinalProject/Hive/Analysis3
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   HDFS Read: 2140582 HDFS Write: 403328 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 23.543 seconds
hive>
```

000000_0

```
bAqt_TWv-f42GWCG2h0_8Q"""Cookies by Design"""North Olmsted,"OH5.01
LixFCMGKdptI8WRsjAl5cQ"""McDonald's"""AGINCOURTON2.51
qdCwzhJ5Yo_Sdm_bYDIfOQ"""Kathy's Alterations"""AhwatukeeAZ5.01
FYWN1wneaV18bWNqQjJ2GNg"""Dental by Design"""AhwatukeeAZ4.02
#NAME?"""My Wine Cellar"""AhwatukeeAZ4.02
qRuKEfk4Iqzm8TIu6PuN_w"""Desert Foothills Trailhead"""Ahwatukee Foothills VillageAZ5.01
NZ8YwxktPtdg3G8qytReEA"""Country Cheese"""AjaxON5.01
1RItXbaPII-4XOLJhOXSMQ"""Anh's Tailor & Alterations"""AjaxON5.01
0qMfy3-ai48cEwiX-y1h4g"""Vape Awesome"""AjaxON5.01
tJRDll5yqpZwehenzE2cSg"""Ransom Bay Inn B & B"""AlburgVT5.01
_WbkQha0xqf2DnYumHAaqg"""Island Tree Service"""AlburghVT3.51
UIRSlulsMyfZpULunCEgqw"""West End Overlook"""AlleghenyPA5.01
rmc_uZnP3iZtqijqkWLGpg"""Yeager Ed Auto Body"""Allison ParkPA5.01
CrDHpkNGvHUqLAMynKfBZA"""Go Ape"""Allison ParkPA5.01
ftx72aqADrNYm0lMytdtCg"""Maid On The Spot"""AmbridgePA5.01
EOmRHPvzR88a5R2i2uD0cQ"""Caffe 501"""AmbridgePA4.52
729grSa1Wsn-hfv7D5u0xg"""Pizza House"""AmbridgePA4.52
GI1WxFbY9tJ9-ChRrwnrzg"""The Stuffed Pepper"""AmbridgePA4.52
ASnhVvBM-f9Ay4R6fy65kA"""AutoNation Ford Amherst"""AmherstOH5.01
p5zGRe5qEQ3Dh5XkyzTXPg"""Sundays Plus Delicatessen"""AmherstOH5.01
ymT1fe7ylbmDVjZimyFcTQ"""Zibo Anjou"""AnjouQC3.51
1RaqclhYT9KkJ4eznsI86Q"""Bâton Rouge"""AnjouQC3.51
mMtJXj-dEEzF4aBs590psA"""Costco Anjou"""AnjouQC3.51
fNmehzZMf-yu70fi3OAt7A"""Holland Marsh Soupfest"""AnsnorveldtON4.51
TbqVqdx5tS54PxLq74Y8iA"""Andrew Z"""AnthemAZ5.01
Ke_ydwvg9LTM8ljHZjuMFg"""Douglas and Tina Kellock - Certified iCracked Technician"""AnthemAZ5.01
J9kysUeo3IVGrC70vZekqA"""Deer Valley Credit Union"""AnthemAZ5.01
fULZVqEt3qKFZjMw-cc0kQ"""Rockman Pool Service"""AnthemAZ5.01
mxpiZIFnnm_TQV3D0qclKw"""Auntie Anne's"""AnthemAZ5.01
Caa5LlzxvOXqIu8MEhKXJA"""Bodytech Health & Fitness"""AspinwallPA4.01
u5WuzpBuC79IuCqlLHXpaq"""Casa Del Sole Pizza"""AspinwallPA3.52
6FCohDH3QSX_M8mQVJvelA"""House O' Hockey"""AspinwallPA3.52
1s3U0ZZ4sNchr_izr9CUPQ"""Homegoods"""AuroraOH5.01
2KbMYZcbBb2O6NvQFAXVmg"""5th Avenue Beauty Studio"""AuroraON5.01
DOArSE1O4c84m0lnBnirtA"""That Stereo Shop"""AvalonPA5.01
7YU-IZBvVn9MZWGn9lAqiA"""Golf Zone"""AvonOH5.01
cwKX37ajw4B7K7UJZ5FijA"""Tan With Kare"""AvonOH5.01
vl5V1IRQDSw5CaqFWM61_A"""Sylvester Truck & Tire Service"""AvonOH5.01
vxqpKCqr7nUFKAk0z2_d0A"""Avon Animal & Bird Hospital"""AvonOH5.01
g2livr1lZzy8vPhBQM-1lg"""Discount Tire Store - Avon, OH"""AvonOH5.01
Y2avI2fw7Ms0vtYI7Iz_bg"""CLE Dog Training"""AvonOH5.01
ce5iUqF_dWkF3pCYN4OtOg"""Prayers from Maria - Field of Hope"""AvonOH5.01
qs8mqdVwuGjK3QTm4uIiIA"""PetPeople"""AvonOH5.01
wcfMZkhEADT32i5TTPqWEg"""Humble Pie"""Avon LakeOH5.01
uEby0G8tbJFrkdqWCqDSkQ"""Avon Lake Eye Care"""Avon LakeOH5.01
i4JISkkhysveW2DoUmQVCQ"""Floor Coverings International of Cleveland West"""Avon LakeOH5.01
j1dUPIAN4VOBmY_lAa5ZYQ"""You Buy and We Fly"""AvondaleAZ5.01
dlsTOxPYqYzSviOWSTpNAA"""9th World Vapor"""AvondaleAZ5.01
sWlkx7bBnsJsE5VFf3KV-w"""G & H Air Conditioning and Heating"""AvondaleAZ5.01
H8tTQZiFs4597r9P0Kyr_g"""Cien Motor Werks"""AvondaleAZ5.01
jlA0sxd3mLV-cPbnlMp2tQ"""Avondale Recycling Center"""AvondaleAZ5.01
NCNGUdWWqTRp4iV2CbjNow"""Modern Revival Events"""AvondaleAZ5.01
```

Analysis # 05: To find list of businesses that have above average rating in each state.

1. Implementing inner join in HQL :

Select a.business_id, a.name, a.city, a.state, a.ratings from business a inner join (Select avg(ratings) as Average, state from business group by state) b where a.state = b.state AND a.ratings > b.Average;

2. Inserting the Hive Output to HDFS :

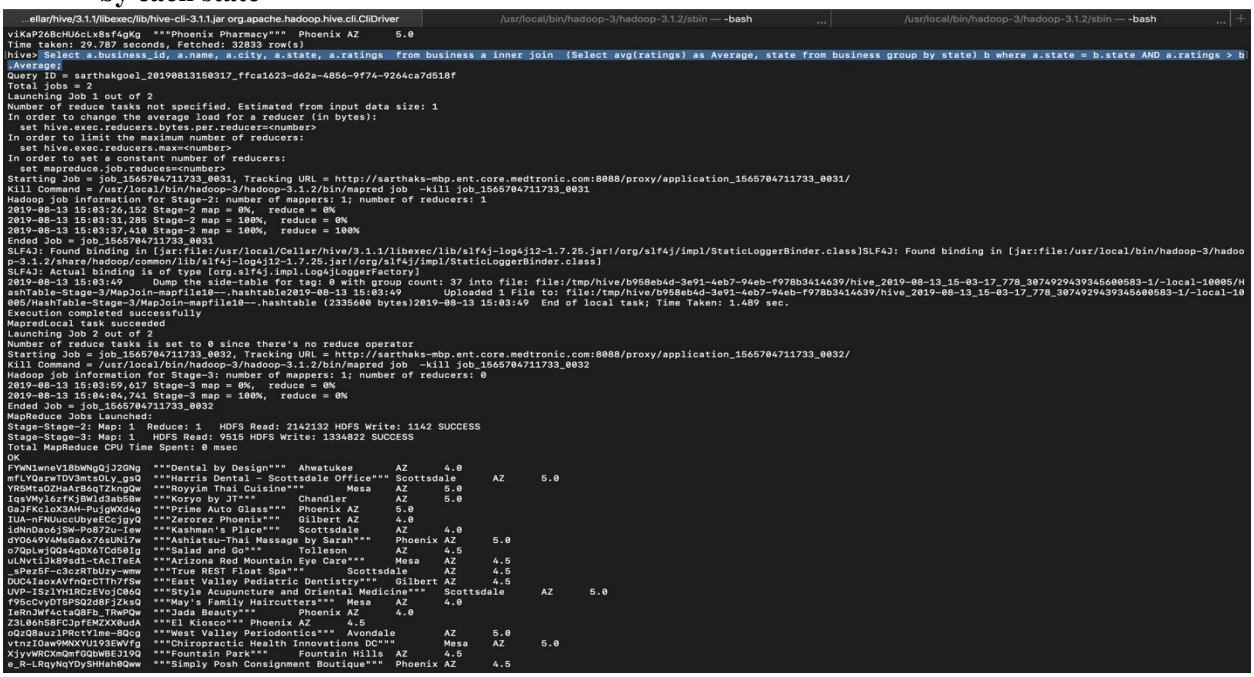
Insert Overwrite Directory 'hdfs://localhost:9000/FinalProject/Hive/Analysis4/' Select a.business_id, a.name, a.city, a.state, a.ratings from business a inner join (Select avg(ratings) as Average, state from business group by state) b where a.state = b.state AND a.ratings > b.Average;

3. Check the file in HDFS Directory :

hadoop fs -lsr /FinalProject/Hive/Analysis4/

poojithas-mbp:sbin poojithamuppalla$ hadoop fs -cat /FinalProject/Hive/Analysis4/000000_0

1. **Implementing inner joins in HQL on business and average rating values of business grouped by each state**



2. **Inserting the Hive Output to HDFS using the Insert Overwrite:**

```
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Kill Command = /usr/local/bin/hadoop-3/hadoop-3.1.2/bin/mapred job  -kill job_1565726254
4196_0001
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2019-08-13 15:58:22,301 Stage-2 map = 0%,  reduce = 0%
2019-08-13 15:58:28,437 Stage-2 map = 100%,  reduce = 0%
2019-08-13 15:58:33,539 Stage-2 map = 100%,  reduce = 100%
Ended Job = job_1565726254196_0001
SLF4J: Found binding in [jar:file:/usr/local/Cellar/hive/3.1.1/libexec/lib/slf4j-log4j1
```

```
...ellar/hive/3.1.1/libexec/lib/hive-cli-3.1.1.jar org.apache.hadoop.hive.cli.CliDriver ...  /usr/local/bin/hadoop-3/hadoop-3.1.2/sbin — -bash  /usr/local/bin/hadoop-3/hadoop-3.1.2/sbin — -bash ...
33135 RunJar
                                    fs -lsr /FinalProject/Hive/Analysis4/
2019-08-13 15:59:57,101 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
lsr: DEPRECATED: Please use 'ls -R' instead.
-rw-r--r--   1 sarthakgoel supergroup    1125693 2019-08-13 15:59 /FinalProject/Hive/Analysis4/000000_0
                                    $ hadoop fs -cat /FinalProject/Hive/Analysis4/000000_0
2019-08-13 16:00:22,434 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
FYWNiwneV18bWNgQj22GNg"""Dental by Design"""AhwatukeeAZ4.0
mfLYQarwTDV3mtsOLy_gsQ"""Harris Dental - Scottsdale Office"""ScottsdaleAZ5.0
YRSMtaOZHaAr8GqTZkngQw"""Royyim Thai Cuisine"""MesaAZ5.0
IqsVMyl6zfKJBWld3ab5Bw"""Koryo by JT"""ChandlerAZ5.0
GaJFKc1oX3AH-PujgWXd4g"""Prime Auto Glass"""PhoenixAZ5.0
IUA-nFNUuccUbyeECcjgyQ"""Zerorez Phoenix"""GilbertAZ4.0
idNnDao6jSW-Po872u-Iew"""Kashman's Place"""ScottsdaleAZ4.0
dYO649V4MsGa6x76sUNi7w"""Ashiatsu-Thai Massage by Sarah"""PhoenixAZ5.0
o7QpiwjGQs4qDX67CdG01g"""Salad and Go"""TollesonAZ4.5
uLNvtIJk89sd1-tAcITeEA"""Arizona Red Mountain Eye Care"""MesaAZ4.5
_sPez5F-c3czRTbUzy-wmw"""True REST Float Spa"""ScottsdaleAZ4.5
DUCAIaoxAVfnQrCTTh7f5w"""East Valley Pediatric Dentistry"""GilbertAZ4.5
UVP-ISz1YH1RCzEVojC06Q"""Style Acupuncture and Oriental Medicine"""ScottsdaleAZ5.0
f9SsCvyDT5PSQ2d8FjZksQ"""May's Family Haircutters"""MesaAZ4.0
1eRnJWf4ctaQ8Fb_TRwPQw"""Jada Beauty"""PhoenixAZ5.0
Z3L06hSBFCJpfEMZXX8udA"""El Kiosco"""PhoenixAZ4.5
oQzQ8auzlPRctYlme-BQcg"""West Valley Periodontics"""AvondaleAZ5.0
vtnzIOsw9WNKYUl93EWVfg"""Chiropractic Health Innovations DC"""MesaAZ5.0
XjyvWRCXmQmfGQbWBEJ19Q"""Fountain Park"""Fountain HillsAZ4.5
e_R-LRqyNqYDySNHeh0Qww"""Simply Posh Consignment Boutique"""PhoenixAZ4.5
8keN8Z2syiAEXgMC3MQpDg"""My House"""ChandlerAZ4.0
MMAupCpJO1TAoK3fuhH98A"""Steven Jones - JR Jones Realty"""ScottsdaleAZ5.0
aXsmmtepl0JZNcl3pKQpew"""Sacks Sandwiches - Hardy"""TempeAZ4.0
a1KKovXPdeDurOrER4ST6A"""Akaihana Sushi & Grill"""PhoenixAZ4.0
8fFF8nIwS01bHjkIhvI1tw"""U Fix It Automotive"""MesaAZ4.0
yqq7lrR1XwQqj-k1RXMqQ"""Revive Stainless Steel"""ScottsdaleAZ5.0
Ie1rZ0A0nX-buH730LtiyQ"""Arcadia Endodontics"""PhoenixAZ5.0
hV58TCLS0DqbrNXS3-jkBw"""R&S Mattress"""PhoenixAZ5.0
WMNaBWBBFVI26xQ8agRSoA"""Teddy Shonka - North & Co"""PhoenixAZ5.0
lc_7se15hHqecsZvUUfh1Q"""European Master"""PhoenixAZ5.0
y-5xgUA-5Tb1xfTyC6VjVw"""Sabuddy"""TempeAZ4.5
Xqu75Zj2v7Z0dyTNA7NG3Q"""Eco King"""PhoenixAZ5.0
p0bg7la5-gaUJyBDjuDCxA"""Furry Beginnings"""GilbertAZ4.0
O9Il5xsoiAzaIIiYMFwnQ"""Bindl Chiropractic"""PhoenixAZ5.0
PrBOEDmZepfXns_ws4iKpQ"""Phoenix Christian"""PhoenixAZ4.0
W2Yln--3Yrrq4neTYfn7VA"""Glamour Pets Grooming Salon"""Sun CityAZ5.0
6Anj9mAWX4gCig-Y4WCdHA"""Forum Cafe"""ScottsdaleAZ5.0
xncM4Pcodfy6xcI6iQeL2Q"""American Family Insurance - Dave Markus Agency"""TempeAZ5.0
j4vt-L8gYcj6i0MupdYgaA"""Rustic Sage Healing Haven"""ScottsdaleAZ4.0
erBte7xHcA5mGd-jNNiR-Q"""True Home Maintenance Air Conditioning & Heating"""MesaAZ5.0
hRzWsK7_6IsXC-hDeMuUYw"""Barefoot Surfaces"""GilbertAZ5.0
WPr6M46RrENG-vLS9tp0vA"""Kirkland's"""GilbertAZ4.5
#NAME?"""Cee Cee's On The Corner"""GlendaleAZ4.0
tkS0WaWBSLp_n_KUPDxKFg"""Nature's Purpose"""TempeAZ5.0
ITOa1olD2HTgEVOnHBzVog"""Espos Mexican Food"""ChandlerAZ4.5
xhfp7Zq_xx9ySVY92J7h-g"""Solar Film Design Window Tint"""ChandlerAZ5.0
yw4WP1QyOzopQyQpKT4o4g"""Golden Spoke Cyclery"""GlendaleAZ5.0
mA2E4F24DMzaJ17dS_qrGg"""Sivlik Grill"""ChandlerAZ4.0
TOQNy5T4V4UkRNFDA4FTSg"""Speedpro Imaging"""TempeAZ5.0
YdSOna9LrLyFgiierHrMIw"""Metier Pharmacy"""PhoenixAZ5.0
Mw9MteSVrYCo4F1_0c1WAw"""Jay Galati, DDS, MSD, PC"""ScottsdaleAZ5.0
Az0v-7xgMBXcT0BOLyN3Lg"""Rascal's Ranch"""ScottsdaleAZ4.0
tkwdiY1jPRcAJetoPIkf_Q"""Pima North Animal Hospital"""ScottsdaleAZ4.5
ZTYiWbl-UYX3pOPpj5TYCA"""Bike Emporium"""ScottsdaleAZ4.5
G8zFSnrALsy2xXE1gEDVUw"""NightRider Jewelry"""ScottsdaleAZ5.0
```

MAPREDUCE ANALYSIS:

Analysis # 06: To implement the partitioning on the basis of year the user joined the yelp.

Partitioning Pattern of Data Organization Technique Used
Partitioning is done on the basis of year. Year is extracted from the Date column in user dataset.

In partitioning technique, the only thing is we need to know how many partitions will be there.

Mapper Class:

public class Mapper1 extends Mapper<Object, Text, IntWritable, Text>{

private final static SimpleDateFormat frmt = new SimpleDateFormat("yyyy-MM-DD");

```
   private IntWritable outkey = new IntWritable();

        @Override
                        protected void map(Object key, Text value, Context context) throws
IOException, InterruptedException {

                String input[] = value.toString().split(",",-1);                if(input[3].length() == 10) {
                    String strDate = input[3];
      Calendar cal = Calendar.getInstance();

      try {
                                cal.setTime(frmt.parse(strDate));
                    } catch (ParseException e) {

              }

      outkey.set(cal.get(Calendar.YEAR));        context.write(outkey, value);
              }
        }
}
```

Custom Practitioner Class:

```
public class Partitioner1 extends Partitioner<IntWritable, Text> implements Configurable {

 private static final String MIN_LAST_ACCESS_DATE_YEAR =
"min.last.access.date.year";

private Configuration conf = null;
private int minLastAccessDateYear = 0;

 public int getPartition(IntWritable key, Text value, int numPartitions) {
                        return key.get() - minLastAccessDateYear;
        }

          public Configuration getConf() {                    return
conf;
        }

                public void setConf(Configuration conf) {
                  this.conf = conf;
                    minLastAccessDateYear = conf.getInt(MIN_LAST_ACCESS_DATE_YEAR,
0);
        }

          public static void setMinLastAccessDate(Job job, int minLastAccessDateYear) {
                    job.getConfiguration().setInt(MIN_LAST_ACCESS_DATE_YEAR,
minLastAccessDateYear);
```

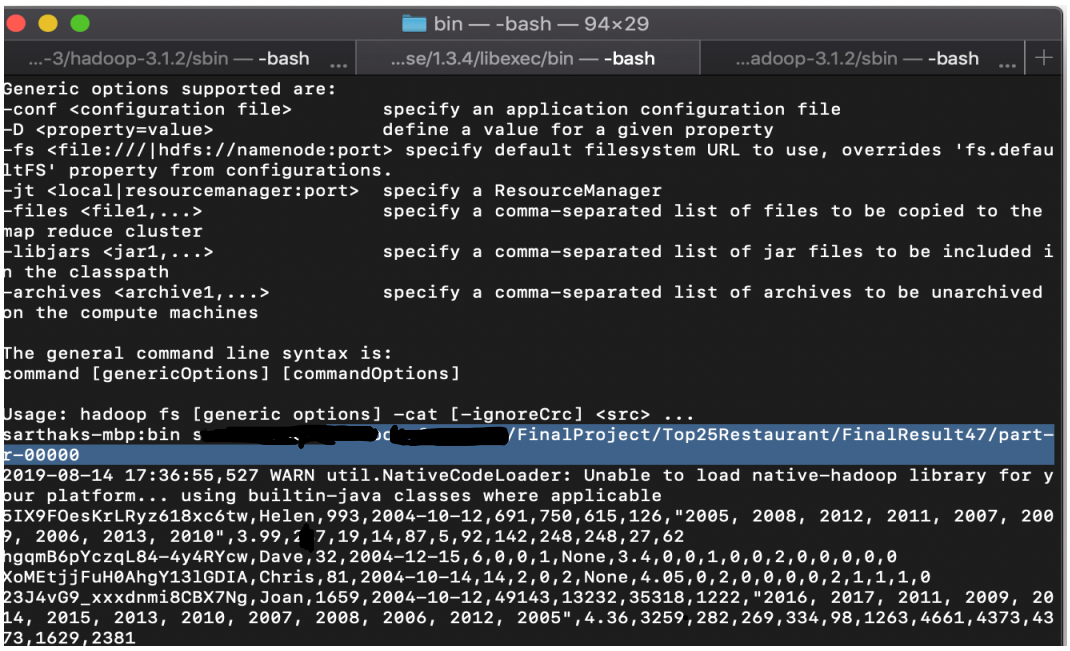
```
        }
}
```

Reducer Class

```
public class Reducer1 extends Reducer<IntWritable, Text, Text,
NullWritable> {

        protected void reduce(IntWritable key, Iterable<Text> values, Context context) throws
IOException, InterruptedException {
                for (Text t : values) {
                        context.write(t, NullWritable.get());
            }
        }
}
```

Driver Class

```
public static void main(String[] args) throws IOException,
ClassNotFoundException, InterruptedException {

                Configuration conf = new Configuration();          Job   job   =
new Job(conf, "DataOrg");
                    job.setJarByClass(DriverClass.class);

                job.setPartitionerClass(Partitioner1.class);
Partitioner1.setMinLastAccessDate(job, 2004);

                 job.setNumReduceTasks(14);

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

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

    job.setMapperClass(Mapper1.class);       job.setReducerClass(Reducer1.class);

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


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

}
```



Analysis # 07: MapReduce Chaining to calculate the no. of users signed up on yelp each year

Chaining Operation Performed

MapReduce Job 01: Partition the files on the basis of years – Partitioning Technique

MapReduce Job 02: Count the no of files in a given partition

Driver Class

```
public class DriverClass {
                    public static void main(String[] args) throws IOException, ClassNotFoundException, InterruptedException {

                Configuration conf = new Configuration();          Job  job  = new Job(conf, "MRchain");
                    job.setJarByClass(DriverClass.class);

                job.setPartitionerClass(Partitioner1.class);
Partitioner1.setMinLastAccessDate(job, 2004);

                 job.setNumReduceTasks(14);

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

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

job.setMapperClass(Mapper1.class);       job.setReducerClass(Reducer1.class);

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


        boolean result = job.waitForCompletion(true);              if(result) {
                    Job job1 = Job.getInstance(conf, "chain");
                  job1.setJarByClass(DriverClass.class);

            job1.setMapperClass(Mapper2.class);              job1.setCombinerClass(Reducer2.class);
job1.setReducerClass(Reducer2.class);

            job1.setOutputKeyClass(Text.class);
job1.setOutputValueClass(IntWritable.class);

                        FileInputFormat.addInputPath(job1, new Path(args[1]));
                        FileOutputFormat.setOutputPath(job1, new Path(args[2]));

                    result = job1.waitForCompletion(true);
        }
    }
}
```

Mapper01

```
public class Mapper1 extends Mapper<Object, Text, IntWritable, Text>{

                private final static SimpleDateFormat frmt = new
SimpleDateFormat("yyyy-MM-DD");

  private IntWritable outkey = new IntWritable();

      @Override
                protected void map(Object key, Text value, Context context) throws
IOException, InterruptedException {

                String input[] = value.toString().split(",",-1);
if(input[3].length() == 10) {                        String strDate = input[3];
                        Calendar cal = Calendar.getInstance();
        try {
                        cal.setTime(frmt.parse(strDate));
                } catch (ParseException e) {

            }

    outkey.set(cal.get(Calendar.YEAR));
    context.write(outkey, value);
```

```
                }
        }
}
```

Custom practitioner class

```
public class Partitioner1 extends Partitioner<IntWritable, Text> implements Configurable {

                private static final String MIN_LAST_ACCESS_DATE_YEAR =
"min.last.access.date.year";

        private Configuration conf = null;          private                int
minLastAccessDateYear = 0;

        public int getPartition(IntWritable key, Text value, int numPartitions) {
                        return key.get() - minLastAccessDateYear;
        }

        public Configuration getConf() {                        return
conf;
        }
 public void setConf(Configuration conf) {          this.conf = conf;
                        minLastAccessDateYear = conf.getInt(MIN_LAST_ACCESS_DATE_YEAR,
0);
        }

        public static void setMinLastAccessDate(Job job, int minLastAccessDateYear) {
                job.getConfiguration().setInt(MIN_LAST_ACCESS_DATE_YEAR,
minLastAccessDateYear);
        }
}
```

Reducer01

```
public class Reducer1 extends Reducer<IntWritable, Text, Text,
NullWritable> {

        protected void reduce(IntWritable key, Iterable<Text> values, Context context) throws
IOException, InterruptedException {
                        for (Text t : values) {
                                context.write(t, NullWritable.get());
                }
        }
}
```

Mapper2

```
public class Mapper2 extends Mapper<Object, Text, Text, IntWritable>{

        private final static SimpleDateFormat frmt = new SimpleDateFormat("yyyy-MM-DD");
```

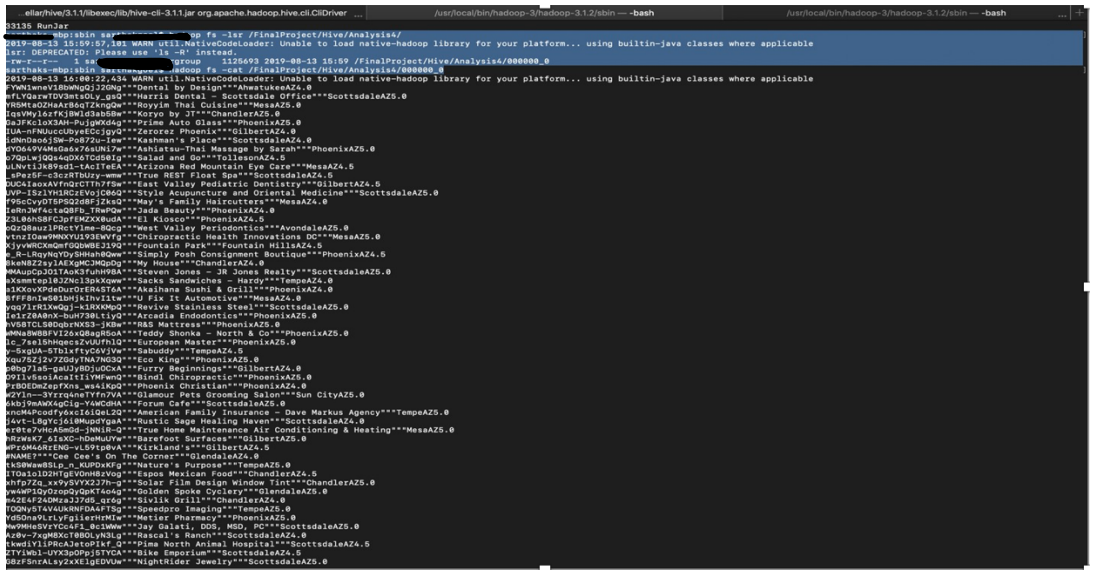
```
                private IntWritable outVal = new IntWritable();
  private Text outKey = new Text();

        @Override
                        protected void map(Object key, Text value, Context context) throws
IOException, InterruptedException {

                String input[] = value.toString().split(",",-1);
if(input[3].length() == 10) {                        String strDate = input[3];
                            Calendar cal = Calendar.getInstance();

                    try {
                                cal.setTime(frmt.parse(strDate));
                        } catch (ParseException e) {

                    }
                            outKey.set(String.valueOf(cal.get(Calendar.YEAR)));
                        outVal.set(1);

                            context.write(outKey, outVal);
            }
      }
```

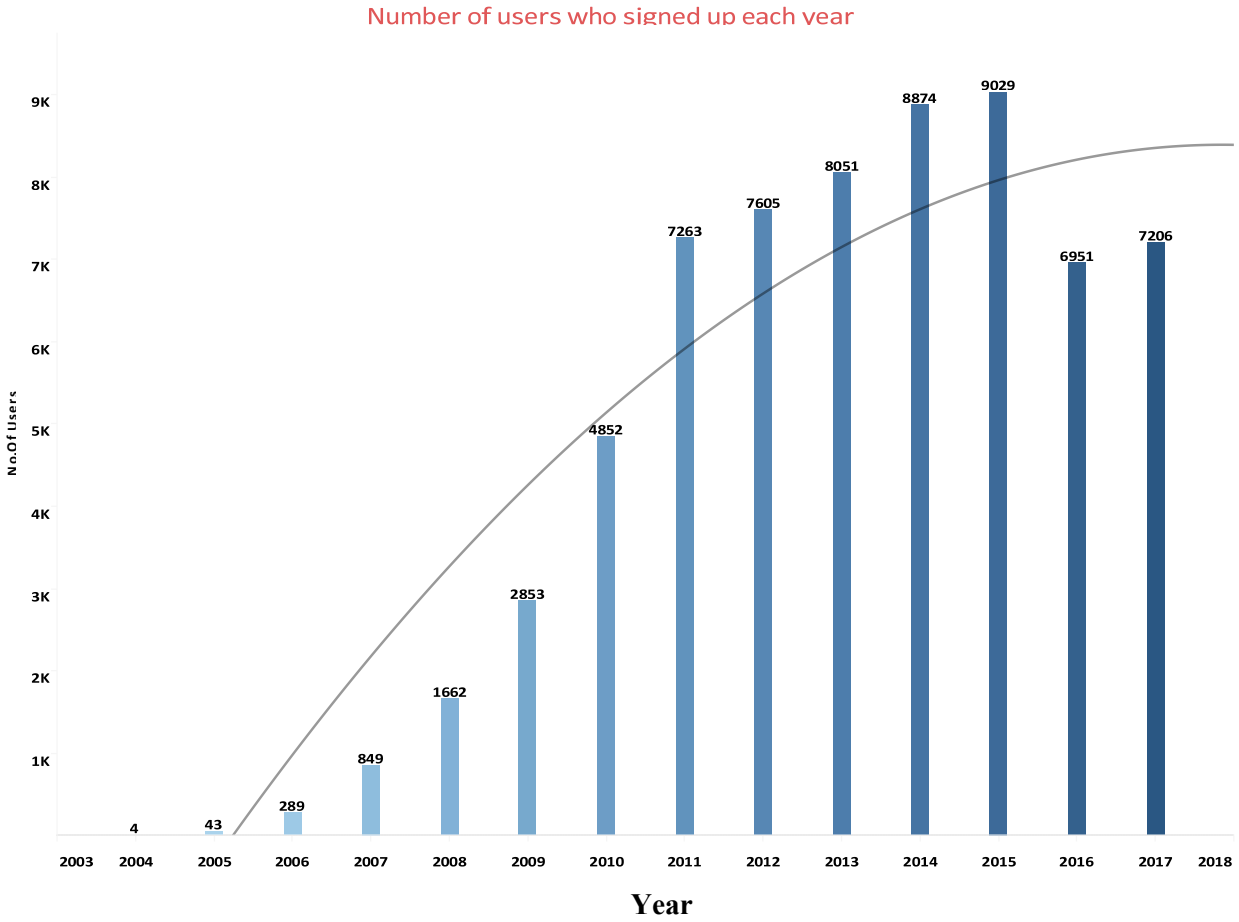
Reducer2

```
private IntWritable result = new IntWritable();

          public void reduce(Text key, Iterable<IntWritable> values,Context context) throws IOException,
InterruptedException
          {
                int sum = 0;
                     for (IntWritable val : values)
              {
                     sum += val.get();
             }       result.set(sum);
context.write(key, result);
          }
```

Command : hadoop jar /Users/poojithamuppalla/eclipse-workspace/MRchain/target/MRchain-0.0.1-SNAPSHOT.jar com.neu.edu.DriverClass/FinalProject/Dataset/yelp_user.csv /FinalProject/Top25Restaurant/FinalResult54/ /FinalProject/Top25Restaurant/FinalResult55

```
2019-08-14 21:53:09,750 WARN util.NativeCodeLoader: Unable to load native-hadoop library for y
our platform... using builtin-java classes where applicable
2019-08-14 21:53:10,263 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-08-14 21:53:10,706 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing
 not performed. Implement the Tool interface and execute your application with ToolRunner to r
emedy this.
2019-08-14 21:53:10,722 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path:
 /tmp/hadoop-yarn/staging/sarthakgoel/.staging/job_1565831423867_0007
2019-08-14 21:53:10,889 INFO input.FileInputFormat: Total input files to process : 1
2019-08-14 21:53:10,928 INFO mapreduce.JobSubmitter: number of splits:1
2019-08-14 21:53:11,033 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1565831423
867_0007
2019-08-14 21:53:11,035 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-08-14 21:53:11,168 INFO conf.Configuration: resource-types.xml not found
2019-08-14 21:53:11,168 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-08-14 21:53:11,224 INFO impl.YarnClientImpl: Submitted application application_1565831423
867_0007
2019-08-14 21:53:11,258 INFO mapreduce.Job: The url to track the job: http://Sarthaks-MacBook-
Pro.local:8088/proxy/application_1565831423867_0007/
2019-08-14 21:53:11,259 INFO mapreduce.Job: Running job: job_1565831423867_0007
2019-08-14 21:53:17,361 INFO mapreduce.Job: Job job_1565831423867_0007 running in uber mode :
false
2019-08-14 21:53:17,362 INFO mapreduce.Job:  map 0% reduce 0%
2019-08-14 21:53:22,427 INFO mapreduce.Job:  map 100% reduce 0%
2019-08-14 21:53:27,464 INFO mapreduce.Job:  map 100% reduce 7%
2019-08-14 21:53:30,486 INFO mapreduce.Job:  map 100% reduce 14%
```





Analysis # 08: Number of users who have yelp account but never provided any reviews yet:

Performing Minus Operation to find the data entries present in dataset that are not present in another dataset. There are users on yelp who haven't reviewed/rated any business yet, this MapReduce program will help find the details of such uesrs.

Mapper#01

```
public class Mapper1 extends Mapper<Object, Text, Text, Text>{

        public Text outKey = new Text();
  public Text outVal = new Text();

    @Override
     protected void map(Object key, Text value, Mapper<Object, Text, Text, Text>.Context context)
                    throws IOException, InterruptedException {

                String[] input = value.toString().split(",",-1);
  String userId = input[0];

  if(userId!= null || !userId.isEmpty()) {        outKey.set(userId);
    outVal.set("U" + value.toString());
```

```
        context.write(outKey, outVal);
    }
      }
}
```

Mapper#02

```
public class Mapper2 extends Mapper<Object, Text, Text, Text>{          public Text outKey
= new Text();     public Text outVal = new Text();

        @Override
         protected void map(Object key, Text value, Mapper<Object, Text, Text, Text>.Context context)
                                throws IOException, InterruptedException {

                               String[] input = value.toString().split(",",-1);
    String userId = input[1];

    if(userId!= null || !userId.isEmpty()) {         outKey.set(userId);
       outVal.set("R" + value.toString());
       context.write(outKey, outVal);
    }
      }
```

Reducer#01

```
public class Reducer1 extends Reducer<Text, Text, Text, Text>{

private static final Text EMPTY_TEXT = new Text(""); private ArrayList<Text> listUsers
= new ArrayList<Text>(); private ArrayList<Text> listReviews = new ArrayList<Text>();
private Text tmpVal = new Text();

private String joinType = null;

                public void setup(Reducer.Context context) {
                        joinType = context.getConfiguration().get("join.type");
      }

        public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {

                  listUsers.clear();
                  listReviews.clear();

                while (values.iterator().hasNext()) {
tmpVal = values.iterator().next();

                          if (Character.toString((char)
tmpVal.charAt(0)).equals("U")) {
                            listUsers.add(new
Text(tmpVal.toString().substring(1)));
                  }
```

```
                        if (Character.toString((char)
tmpVal.charAt(0)).equals("R")) {
                            listReviews.add(new
Text(tmpVal.toString().substring(1)));
                    }
                }
   minusLogic(context);
        }

          public void minusLogic(Context context) {              if
(joinType.equalsIgnoreCase("minus")) {                          for (Text
usr : listUsers) {
if(listReviews.isEmpty()) {
                                try {
                                        context.write(usr, EMPTY_TEXT);
                                } catch (Exception e) {

                                }
                            }
         }
                }
        }
}
```

DriverClass

```
public class DriverClass {

                    public static void main(String[] args) throws IOException,
InterruptedException, ClassNotFoundException {
     Configuration conf = new Configuration();              Job job =
Job.getInstance(conf, "AntiJoin");      job.setJarByClass(DriverClass.class);

     MultipleInputs.addInputPath(job, new Path(args[0]),
TextInputFormat.class, Mapper1.class);
     MultipleInputs.addInputPath(job, new Path(args[1]), TextInputFormat.class, Mapper2.class);
     job.getConfiguration().set("join.type", "minus");      job.setReducerClass(Reducer1.class);

     job.setOutputFormatClass(TextOutputFormat.class);      TextOutputFormat.setOutputPath(job, new
Path(args[2]));      job.setOutputKeyClass(Text.class);      job.setOutputValueClass(Text.class);

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

Output

```
[redacted]ntiJoin-0[redacted]T.jar com.neu.edu.DriverClass /FinalProject/Dataset/yelp_user.csv
/FinalProject/Top25Restaurant/Dataset/yelp_business.cs/ /FinalProject/Top25Restaurant/FinalRes
ult56/
2019-08-14 23:37:03,999 WARN util.NativeCodeLoader: Unable to load native-hadoop library for y
our platform... using builtin-java classes where applicable
2019-08-14 23:37:04,505 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-08-14 23:37:05,020 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing
 not performed. Implement the Tool interface and execute your application with ToolRunner to r
emedy this.
2019-08-14 23:37:05,034 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path:
 /tmp/hadoop-yarn/staging/sarthakgoel/.staging/job_1565831423867_0010
2019-08-14 23:37:05,206 INFO input.FileInputFormat: Total input files to process : 1
2019-08-14 23:37:05,229 INFO mapreduce.JobSubmitter: Cleaning up the staging area /tmp/hadoop-
yarn/staging/sarthakgoel/.staging/job_1565831423867_0010
Exception in thread "main" org.apache.hadoop.mapreduce.lib.input.InvalidInputException: Input
path does not exist: hdfs://localhost:9000/FinalProject/Top25Restaurant/Dataset/yelp_business.
cs
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.singleThreadedListStatus(File
InputFormat.java:332)
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.listStatus(FileInputFormat.ja
va:274)
        at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.getSplits(FileInputFormat.jav
a:396)
        at org.apache.hadoop.mapreduce.lib.input.DelegatingInputFormat.getSplits(DelegatingInp
utFormat.java:115)
        at org.apache.hadoop.mapreduce.JobSubmitter.writeNewSplits(JobSubmitter.java:310)
        at org.apache.hadoop.mapreduce.JobSubmitter.writeSplits(JobSubmitter.java:327)
        at org.apache.hadoop.mapreduce.JobSubmitter.submitJobInternal(JobSubmitter.java:200)
```

```
r-00000
2019-08-14 23:49:06,829 WARN util.NativeCodeLoader: Unable to load native-hadoop library for
our platform... using builtin-java classes where applicable
--0WZ5gklOfbUIodJuKfaQ,Scott,8,2013-02-19,0,0,0,0,None,4.2,0,0,0,0,0,0,0,0,0,0
--1mPJZdSY9KluaBYAGboQ,Bryan,5,2011-07-04,0,0,0,0,None,5.0,0,0,0,0,0,0,0,0,0,0
--3l8wysfp49Z2TLnyT0vg,Benjamin,111,2013-12-14,97,57,32,2,2016,3.43,0,0,0,0,0,0,1,2,2,3,0
--4uW4yJiRT2oXMYkCPq1Q,Ariella,50,2016-10-28,3,2,1,2,2017,4.06,0,0,0,0,0,0,0,0,1,0
--9kVKrIDkSP6lqK2PDTDw,Michelle,1,2015-04-10,0,0,0,0,None,5.0,0,0,0,0,0,0,0,0,0,0
--ERQVpqAAoi262TTbLVzQ,Regina,1,2011-04-02,0,0,0,0,None,5.0,0,0,0,0,0,0,0,0,0,0
--ET3paBtrThD95dk72Cqg,Lynda,8,2008-05-25,39,2,5,0,None,4.56,0,0,0,0,0,0,0,0,0,0
--FtMNXFtTE_KixNgPD1AQ,Ranjana,1,2017-11-06,0,0,0,0,None,5.0,0,0,0,0,0,0,0,0,0,0
--GwB-sktmoAOPBsbAaiow,Nina,2,2014-08-27,0,0,0,0,None,5.0,0,0,0,0,0,0,0,0,0,0
--I4PlFQXAQcpk_eUJEW6w,Branko,2,2010-04-22,0,0,0,0,None,2.0,0,0,0,0,0,0,0,0,0,0
--JSMB52zXJr_LB1klikyA,Diana,1,2017-06-14,0,0,0,0,None,1.0,0,0,0,0,0,0,0,0,0,0
--K1aJ5K8ZLIfDd_NjySbA,Lisa,1,2016-03-13,0,0,0,0,None,5.0,0,0,0,0,0,0,0,0,0,0
--NIc98RMssgy0mSZL3vpA,Nick,65,2013-08-15,2,0,0,1,None,3.91,0,0,0,0,0,0,1,1,0,0
--vrLQPsqckRv9TVqzg5wA,Aameemanu,17,2010-10-04,50,14,10,10,None,3.78,8,2,1,0,0,6,9,11,11,0,0
--xdSgqUJmcvJot-3OIq0g,Michelle,494,2008-08-02,14,8,1,16,"2014, 2011, 2015, 2010, 2016, 2012
2013",3.53,5,4,1,0,1,0,4,8,8,6,0
-01Cd1PEJYHhZYabHv8mIw,parag,5,2011-08-20,0,0,0,0,None,1.8,0,0,0,0,0,0,0,0,0,0
-01CztZ8o5UbarmEiQLDsQ,Kelly,2,2015-07-05,0,0,0,0,None,5.0,0,0,0,0,0,0,0,0,0,0
-04zuZ0tQoGpgG49PbY-2Q,miafri,2,2012-11-26,0,0,0,0,None,5.0,0,0,0,0,0,0,0,0,0,0
-0AyZxS5C--WySnbW_Q8yQ,Lars,507,2010-07-26,4,6,2,11,"2017, 2016, 2013, 2012, 2015, 2014",3.6
2,4,0,0,0,2,8,5,5,6,0
-0HHK0ux-abVe1mPQjyh7g,Lauren,3,2012-06-24,0,0,0,0,None,3.67,0,0,0,0,0,0,0,0,0,0
-0HHSb5xE3rNzXdqYIY3rQ,Emil,1,2013-05-20,0,0,0,0,None,2.0,0,0,0,0,0,0,0,0,0,0
-0Hbf-cgvSsu8749nt1uyg,Adina,2,2015-10-31,0,1,0,0,None,3.0,0,0,0,0,0,0,0,0,0,0
```

Analysis # 09: Secondary Sort to find out how well a business is been rated by users per year:

Composite Key – BusinessID + Year
Extract Year from the given date and sort on the basis of ratings given by users to a business.
Following classes are created to perform secondary sort on the data.
1. CompositeKeyClass
2. MapperClass
3. Custom Partitioner Class
4. GroupComapartor Class
5. Secondary Sort Class
6. ReducerClas

CompositeKeyClass:

```
public class CompositeKeyClass implements Writable,
WritableComparable<CompositeKeyClass>{

                private String businessYear;
               private Float rating;

                 public String getBusinessYear() {
                     return businessYear;
        }

          public void setBusinessYear(String businessYear) {                    this.businessYear                    =
businessYear;
        }

                public Float getRating() {
                    return rating;
        }

          public void setRating(Float rating) {                    this.rating = rating;
        }

          public void readFields(DataInput in) throws IOException {                    businessYear                =
in.readUTF();
                        rating = in.readFloat();
        }

          public void write(DataOutput out) throws IOException {
out.writeUTF(businessYear);                    out.writeFloat(rating);
        }

                  public int compareTo(CompositeKeyClass o) {
                    int result = this.businessYear.compareTo(o.getBusinessYear());                if (result == 0) {
```

```
                return this.rating.compareTo(o.getRating());
            }
                return result;
        }

        @Override
            public String toString() {

                return businessYear;
        }
}
```

Mapper

```
public class MapperClass extends Mapper<Object, Text,
CompositeKeyClass, FloatWritable>{

        private final static SimpleDateFormat frmt = new SimpleDateFormat("MM/DD/yy");
   private FloatWritable outVal = new FloatWritable();

        @Override
        protected void map(Object key, Text value, Mapper<Object, Text, CompositeKeyClass,
FloatWritable>.Context context)
                        throws IOException, InterruptedException {

            try {
                    String input[] = value.toString().split("\t");                    if(input.length >
5) {

                            Calendar cal = Calendar.getInstance();
                        String strDate = input[4];
                            String businessId = input[2];
                    String ratings = input[3];
        try {
                                cal.setTime(frmt.parse(strDate));
                            } catch (ParseException e) {

                        }

                                String businessYear = businessId + " - " +
String.valueOf(cal.get(Calendar.YEAR));
                            CompositeKeyClass compKey = new
CompositeKeyClass();
                            compKey.setBusinessYear(businessYear);
        compKey.setRating(Float.parseFloat(ratings));
                            outVal.set(Float.parseFloat(ratings));

                        context.write(compKey, outVal);
```

```
                    }
                } catch(Exception e) {

            }
        }
}
```

Partition Class:

```
public class PartitionerClass extends Partitioner<CompositeKeyClass,
FloatWritable> {

        @Override
         public int getPartition(CompositeKeyClass key, FloatWritable text, int numberOfPartitions) {
                // TODO Auto-generated method stub
            return Math.abs(key.getBusinessYear().hashCode() % numberOfPartitions);
        }

}
```

SecondarySortComparator:

```
public class SecondarySortComparator extends WritableComparator {

            protected SecondarySortComparator() {
                super(CompositeKeyClass.class, true);
        }

        @Override
                public int compare(WritableComparable a, WritableComparable b) {

                    CompositeKeyClass ckw1 = (CompositeKeyClass) a;
                    CompositeKeyClass ckw2 = (CompositeKeyClass) b;

                int result =
ckw1.getBusinessYear().compareTo(ckw2.getBusinessYear());

                    if (result == 0) {
                            return -ckw1.getRating().compareTo(ckw2.getRating());
            }

                return result;
        }
}
```

Group Comparator:

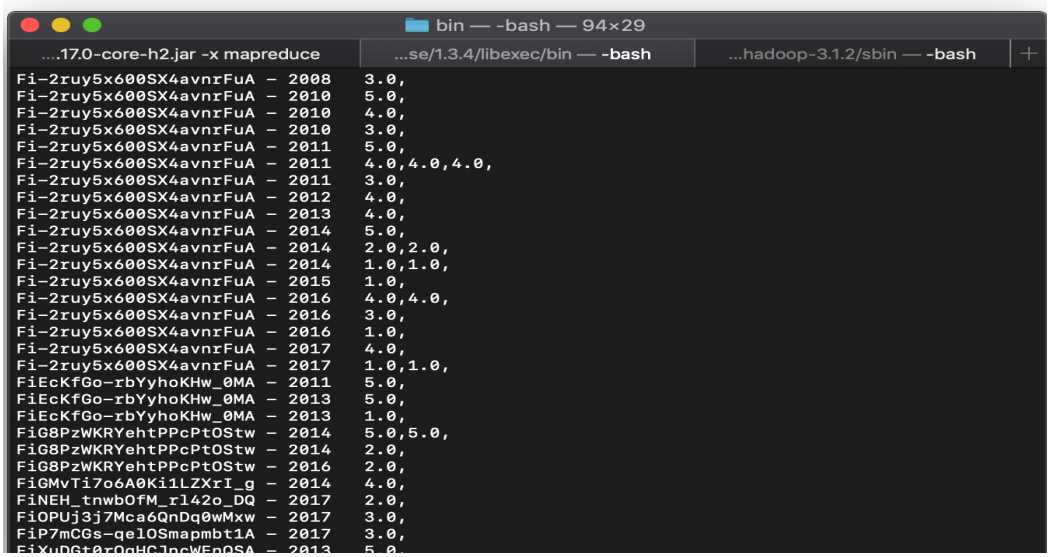
```
public class GroupComparator extends WritableComparator{

        protected GroupComparator() {
                super(CompositeKeyClass.class, true);
    }

     @Override
            public int compare(Object a, Object b) {

                CompositeKeyClass ckw1 = (CompositeKeyClass)a;
                CompositeKeyClass ckw2 = (CompositeKeyClass)b;

            return
ckw1.getBusinessYear().compareTo(ckw2.getBusinessYear());
    }
}
```

Reducer:

```
public class ReducerClass extends Reducer<CompositeKeyClass,
FloatWritable, CompositeKeyClass, Text> {

            Text outVal = new Text();

        @Override
              protected void reduce(CompositeKeyClass key,
Iterable<FloatWritable> values,
                                    Reducer<CompositeKeyClass, FloatWritable,
CompositeKeyClass, Text>.Context context)
                                      throws IOException, InterruptedException {

                                StringBuilder sortedRating = new StringBuilder();
                for(FloatWritable val : values){                    sortedRating.append(val.get());
                                sortedRating.append(",");
                }
                            outVal.set(sortedRating.toString());
                        context.write(key,outVal);
        }
}
```

```
/target/SecondarySorting-0.0.1-SNAPSHOT.jar com.neu.edu.DriverClass /FinalProject/Dataset/yelp
_review_tsv.tsv /FinalProject/Top25Restaurant/FinalResult59
2019-08-15 18:39:56,403 WARN util.NativeCodeLoader: Unable to load native-hadoop library for y
our platform... using builtin-java classes where applicable
2019-08-15 18:39:57,016 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-08-15 18:39:57,360 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing
 not performed. Implement the Tool interface and execute your application with ToolRunner to r
emedy this.
2019-08-15 18:39:57,377 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path:
 /tmp/hadoop-yarn/staging/sarthakgoel/.staging/job_1565908449557_0002
2019-08-15 18:39:57,541 INFO input.FileInputFormat: Total input files to process : 1
2019-08-15 18:39:57,586 INFO mapreduce.JobSubmitter: number of splits:1
2019-08-15 18:39:57,692 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1565908449
557_0002
2019-08-15 18:39:57,694 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-08-15 18:39:57,831 INFO conf.Configuration: resource-types.xml not found
2019-08-15 18:39:57,832 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-08-15 18:39:57,882 INFO impl.YarnClientImpl: Submitted application application_1565908449
557_0002
2019-08-15 18:39:57,920 INFO mapreduce.Job: The url to track the job: http://Sarthaks-MacBook-
Pro.local:8088/proxy/application_1565908449557_0002/
2019-08-15 18:39:57,920 INFO mapreduce.Job: Running job: job_1565908449557_0002
2019-08-15 18:40:02,995 INFO mapreduce.Job: Job job_1565908449557_0002 running in uber mode :
false
2019-08-15 18:40:02,997 INFO mapreduce.Job:  map 0% reduce 0%
2019-08-15 18:40:09,060 INFO mapreduce.Job:  map 100% reduce 0%
2019-08-15 18:40:14,105 INFO mapreduce.Job:  map 100% reduce 100%
2019-08-15 18:40:14,113 INFO mapreduce.Job: Job job_1565908449557_0002 completed successfully
```

hadoop fs -cat /FinalProject/Top25Restaurant/FinalResult59/part-r-00000



Analysis # 10: To Find the top 25 most rated business:

MapReduce Chaining

MapReduce Job 1: To find the count of no. of times each business is rated
MapReduce Job 2: Apply Top N Filtering Technique to find the top 25 rated businesse
Mapper # 01

```
public class Mapper1 extends Mapper<Object, Text, Text, IntWritable>{

        Text outKey = new Text();
        IntWritable outVal = new IntWritable();
         @Override
        protected void map(Object key, Text value, Mapper<Object, Text, Text, IntWritable>.Context
context)
                                throws IOException, InterruptedException {

                  try {
                     String input[] = value.toString().split("\t");
outKey.set(input[2]);                outVal.set(1);
                        context.write(outKey, outVal);
                    } catch(Exception e) {

                }
        }
}
```

Reducer #01

```
public class Reducer1 extends Reducer<Text, IntWritable, Text,
IntWritable> {

                    IntWritable result = new IntWritable();

          @Override
           protected void reduce(Text key, Iterable<IntWritable> values,                    Reducer<Text,
IntWritable, Text, IntWritable>.Context context) throws IOException, InterruptedException {

                   int sum = 0;
                         for(IntWritable val : values) {
                            sum += val.get();
                }
                    result.set(sum);
context.write(key, result);
        }
}
```

Step 02 to implement top n filtering pattern

Top25Mapper

```
public class Top25Mapper extends Mapper<Object, Text, NullWritable,
Text>{

        private TreeMap<Integer,Text>  tm = new TreeMap<Integer, Text>();  @Override
        protected void map(Object key, Text value, Mapper<Object, Text, NullWritable, Text>.Context
context)
                                throws IOException, InterruptedException {

                                String input[] = value.toString().split("\t");
                try {
                                String business = input[0];
                                    int noOfRating = Integer.parseInt(input[1]);
                                 tm.put(noOfRating, new Text(value));

                               if(tm.size()>25) {
                                        tm.remove(tm.firstKey());
                            }

                        }catch(Exception e) {

                }
        }


         @Override
          protected void cleanup(Mapper<Object, Text, NullWritable, Text>.Context context)
                                throws IOException, InterruptedException {

                        for(Text t: tm.values()) {
                                context.write(NullWritable.get(), t);
                }
        }

}
```

Top25Reducer

```
public class Top25Reducer extends Reducer<NullWritable, Text,
NullWritable, Text>{

                        private TreeMap<Integer, Text> tm = new TreeMap<Integer, Text>();
```

```
        @Override
        protected void reduce(NullWritable key, Iterable<Text> values, Reducer<NullWritable, Text, NullWritable, Text>.Context context) throws IOException, InterruptedException {


                        for(Text value : values) {
                                String input[] = value.toString().split("\t");
          int count = Integer.parseInt(input[1]);              tm.put(count,              new
        Text(value));                if(tm.size()> 25) {
                                tm.remove(tm.firstKey());
                        }
                }
        }

        @Override
        protected void cleanup(Reducer<NullWritable, Text, NullWritable, Text>.Context context)
                        throws IOException, InterruptedException {              for(Text t :
tm.descendingMap().values()) {                          context.write(NullWritable.get(), t);
                }
        }
}
```

Driver Class

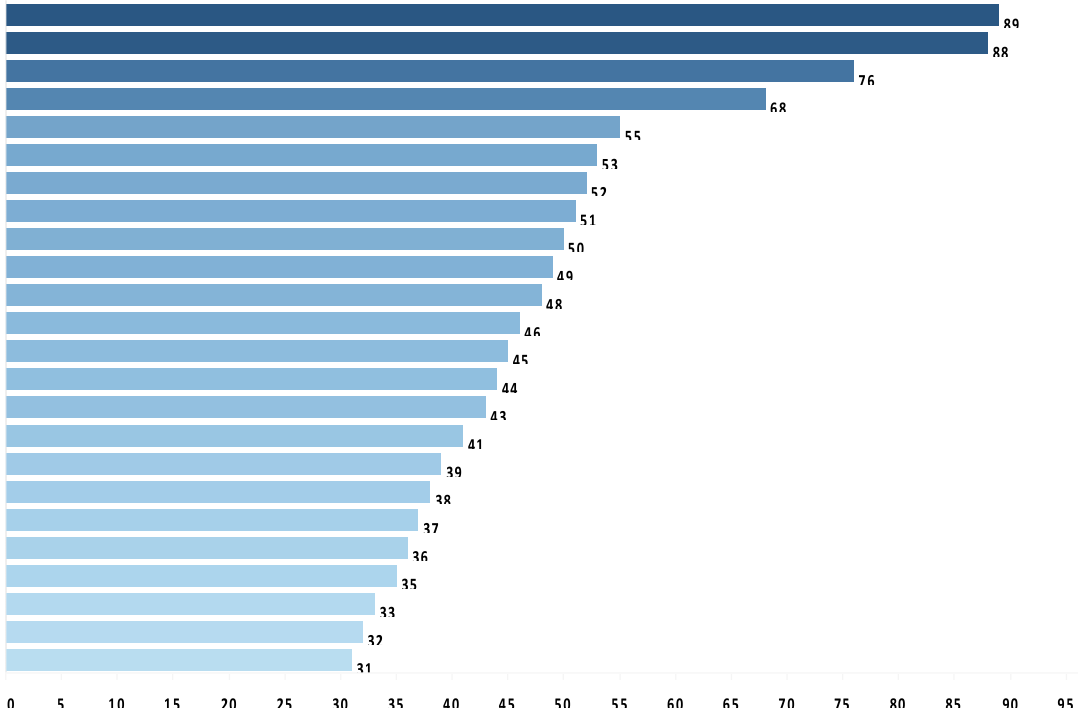
```
public class DriverClass {

                public static void main(String[] args) throws IOException,
ClassNotFoundException, InterruptedException {

                Configuration conf = new Configuration();              Job job =
Job.getInstance(conf, "Top25Business");
job.setJarByClass(DriverClass.class);

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

      job.setInputFormatClass(TextInputFormat.class);
job.setOutputFormatClass(TextOutputFormat.class);        job.setMapOutputKeyClass(Text.class);
job.setMapOutputValueClass(IntWritable.class);

      job.setMapperClass(Mapper1.class);       job.setReducerClass(Reducer1.class);

      job.setOutputKeyClass(Text.class);       job.setOutputValueClass(IntWritable.class);


          boolean result = job.waitForCompletion(true);

          if(result) {
           Job job1 = Job.getInstance(conf, "Top25Business");     job1.setJarByClass(DriverClass.class);
```

```
        job1.setInputFormatClass(TextInputFormat.class);
                job1.setOutputFormatClass(TextOutputFormat.class);
            job1.setMapperClass(Top25Mapper.class);
            job1.setReducerClass(Top25Reducer.class);

            job1.setOutputKeyClass(NullWritable.class);
            job1.setOutputValueClass(Text.class);

            FileInputFormat.addInputPath(job1, new Path(args[1]));
            FileOutputFormat.setOutputPath(job1, new Path(args[2]));

            result = job1.waitForCompletion(true);
        }
    }
}
```

```
rget/NoOfRestaurants-0.0.1-SNAPSHOT.jar com.neu.edu.DriverClass /FinalProject/yelp_business_tsv.tsv /FinalProject/Top25Restaurant/FinalResult80/ /FinalProject/Top25Restaurant/FinalResult81/
2019-08-16 00:21:30,394 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2019-08-16 00:21:30,941 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-08-16 00:21:31,577 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2019-08-16 00:21:31,592 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/sarthakgoel/.staging/job_1565925804834_0016
2019-08-16 00:21:31,782 INFO input.FileInputFormat: Total input files to process : 1
2019-08-16 00:21:31,823 INFO mapreduce.JobSubmitter: number of splits:1
2019-08-16 00:21:31,942 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1565925804834_0016
2019-08-16 00:21:31,944 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-08-16 00:21:32,090 INFO conf.Configuration: resource-types.xml not found
2019-08-16 00:21:32,090 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-08-16 00:21:32,143 INFO impl.YarnClientImpl: Submitted application application_1565925804834_0016
2019-08-16 00:21:32,187 INFO mapreduce.Job: The url to track the job: http://Sarthaks-MacBook-Pro.local:8088/proxy/application_1565925804834_0016/
2019-08-16 00:21:32,187 INFO mapreduce.Job: Running job: job_1565925804834_0016
2019-08-16 00:21:38,266 INFO mapreduce.Job: Job job_1565925804834_0016 running in uber mode : false
2019-08-16 00:21:38,268 INFO mapreduce.Job:  map 0% reduce 0%
2019-08-16 00:21:42,318 INFO mapreduce.Job:  map 100% reduce 0%
2019-08-16 00:21:47,351 INFO mapreduce.Job:  map 100% reduce 100%
```

```
4JNXUYY8wbaaDmk3BPzlWw  89
RESDUcs7fIiihp38-d6_6g  88
K7lWdNUhCbcnEvI0NhGewg  76
cYwJA2A6I12KNkm2rtXd5g  68
iCQpiavjjPzJ5_3gPD5Ebg  55
eoHdUeQDNgQ6WYEnP2aiRw  53
ujHiaprwCQ5ewziu0Vi9rw  52
f4x1YBxkLrZg652xt2KR5g  51
5LNZ67Yw9RD6nf4_UhXOjw  50
El4FC8jcawUVgw_0EIcbaQ  49
hihud--QRriCYZw1zZvW4g  48
KskYqH1Bi7Z_61pH6Om8pg  46
SMPbvZLSMMb7KU76YNYMGg  45
rcaPajgKOJC2vo_l3xa42A  44
FaHADZARwnY4yvlvpnsfGA  43
yfxDa8RFOvJPQh0rNtakHA  41
eAc9Vd6loOgRQolMXQt6FA  39
uanCi4OGc1mHLGl_AT4JhQ  38
ii8sAGBexBOJoYRFafF9XQ  37
MpmFFw0GE_2iRFPdsRpJbA  36
XXW_OFaYQkkGOGniujZFHg  35
QJatAcxYgK1Zp9BRZMAx7g  33
pH0BLkL4cbxKzu471VZnuA  32
na4Th5DrNauOv-c43QQFvA  31
```

Most Reviewed Business By Users



Analysis # 11: Recommendation Using Mahout

Step 1 Pig

A =LOAD 'hdfs://localhost:9000/FinalProject/Dataset/yelp_review_tsv.tsv' USING org.apache.pig.piggybank.storage.CSVExcelStorage('\t', 'NO_MULTILINE', 'UNIX', 'SKIP_INPUT_HEADER');

fltrvw = FOREACH A GENERATE (chararray) $1 as userId, (chararray) $2 as businessID, (chararray) $3 as rating;

STORE fltrvw INTO 'hdfs://localhost:9000/FinalProject/Hive/MahoutInputUsers/' Using PigStorage('\t');

Step 02 Copy the file to local

hadoop fs -copyToLocal /FinalProject/Hive/MahoutInputUsers/part-m-00000 /Users/poojithamuppalla/Desktop/Project/Analysis/Mahout/Input/

Step 03 Program to create a Mahout Input

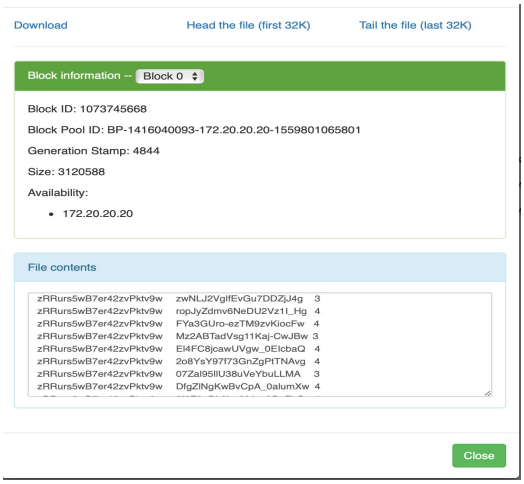
```
static HashMap<String, Integer> um = new HashMap<String, Integer>();
static HashMap<String, Integer> bm = new HashMap<String, Integer>();
private static int getUser(String user)
{
if (!um.containsKey(user))
um.put(user, um.size() + 1);
return um.get(user);
}
private static int getBusiness(String business)
{
if (!bm.containsKey(business))
bm.put(business, bm.size() + 1);
return bm.get(business);
}
private static void saveMapping(HashMap<String, Integer> mapping,
String fileName) throws IOException
{
int limitPrint = mapping.size();
FileWriter fstream;
BufferedWriter out;
fstream = new
FileWriter("/Users/poojithamuppalla/Desktop/Project/Analysis/Mahout/Input/"+ fileName);
out = new BufferedWriter(fstream);
for (Map.Entry<String,Integer> entry : mapping.entrySet()) {
//System.out.println("Key = " + entry.getKey() + ", Value= " + entry.getValue());
out.write(entry.getKey() + "\t");
out.write(entry.getValue() + "\n");
}
out.close();
}
public static void mahoutInput() throws IOException {
try {
File file = new
File("/Users/poojithamuppalla/Desktop/Project/Analysis/Mahout/Input/InputFile.csv");
FileWriter fstream = new
FileWriter("/Users/poojithamuppalla/Desktop/Project/Analysis/Mahout/Input/MahoutInput.csv");
BufferedReader br = new BufferedReader(new FileReader(file));
BufferedWriter out = new BufferedWriter(fstream);
String st;
while ((st = br.readLine()) != null) {
if(!st.isEmpty()) {
String input[] = st.split(",");
int user = getUser(input[0]);
int business = getBusiness(input[1]);
String ratings = input[2];
if(!String.valueOf(user).isEmpty() &&
!String.valueOf(business).isEmpty() && !ratings.isEmpty())
out.write(String.valueOf(user));
out.write(",");
```

```
out.write(String.valueOf(business));
out.write(",");
out.write(ratings);
out.write("\n"); }
}
br.close();
out.close();
} catch(Exception e){
}
//writer.Close();
saveMapping(um, "usersMap.csv");
saveMapping(bm, "songMapping.csv");
}
public static void main(String[] args) throws IOException {mahoutInput();
}
```

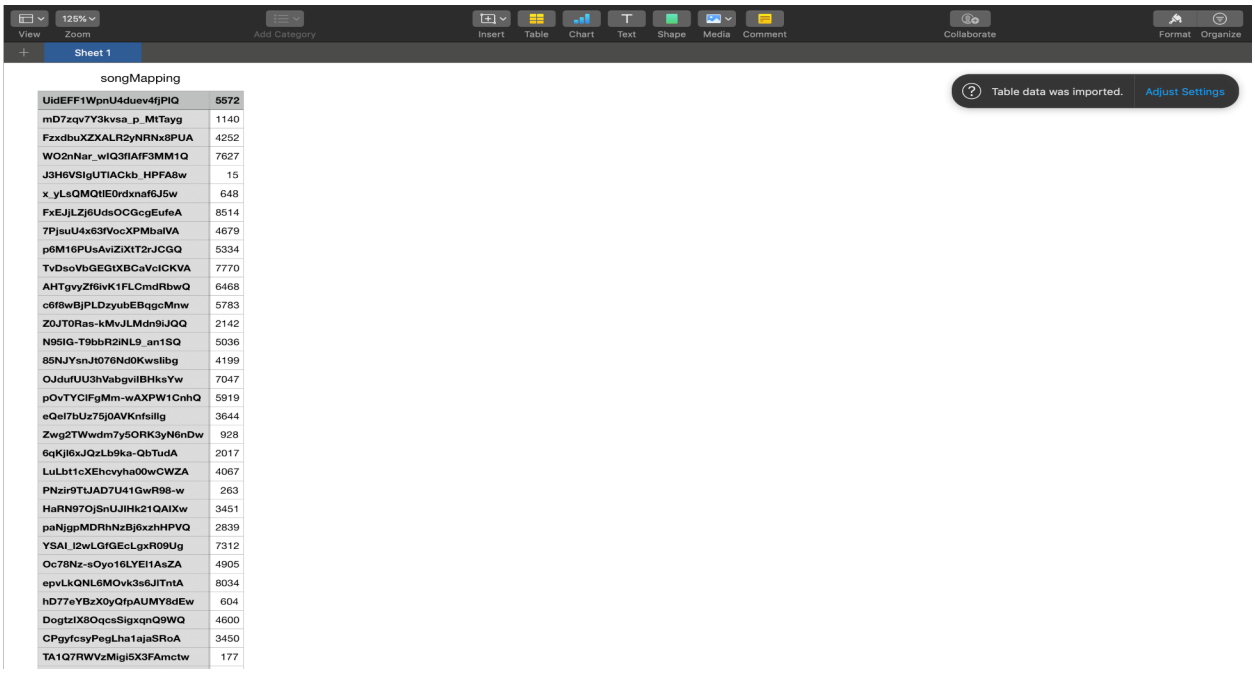
Step 04 Mahout Recommendation

```
public class MahoutRecommender {
public static void main(String[] args) throws IOException,
TasteException{
File userPreferencesFile = new
File("/Users/poojithamuppalla/Desktop/Project/Analysis/Mahout/Input/MahoutInput.csv");
DataModel dataModel = new
FileDataModel(userPreferencesFile);
```

Step01 Pig Commands to get the userId, businessId, Ratings



Step02 to create a mapping file, since the output file is unstructured and Mahout needs a proper structured file to perform recommendation so we are creating a hashmap for businessID and userID and save their mapping Business Mapping

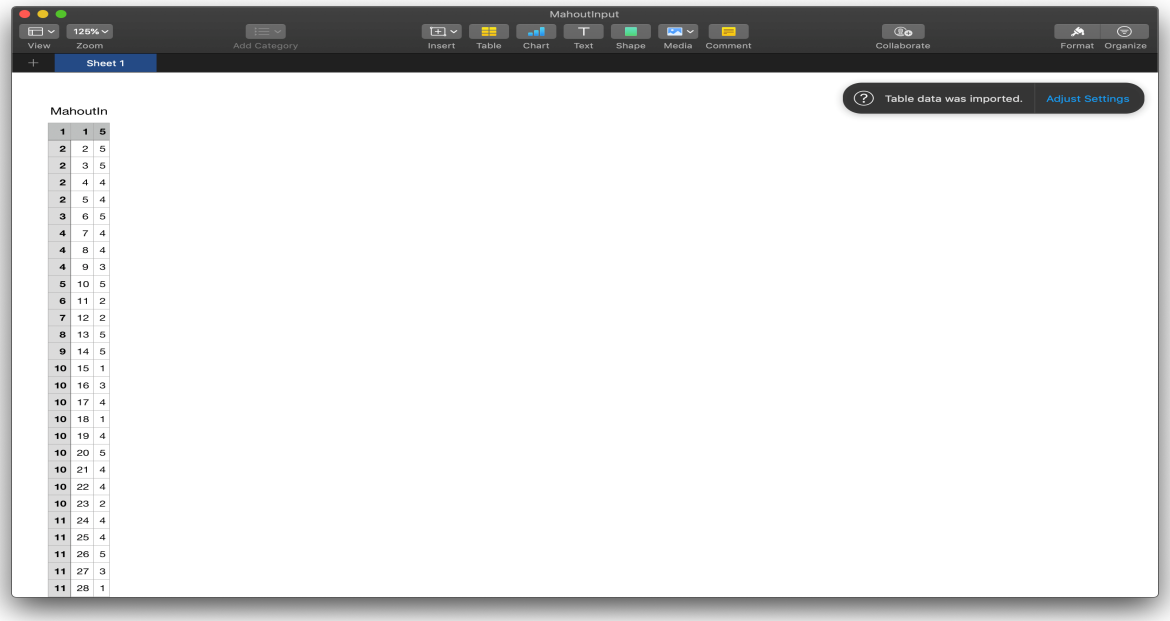


UidEFF1WpnU4duev4fjPlQ	5572
mD7zqv7Y3kvsa_p_MtTayg	1140
FzxdbuXZXALR2yNRNx8PUA	4252
WO2nNar_wIQ3fIAfF3MM1Q	7627
J3H6VSlgUTlACkb_HPFA8w	15
x_yLsQMQtlE0rdxnaf6J5w	648
FxEJjLZj6UdsOCGcgEufeA	8514
7PjsuU4x63fVocXPMbaIVA	4679
p6M16PUsAviZiXtT2rJCGQ	5334
TvDsoVbGEGtXBCaVcICKVA	7770
AHTgvyZf6ivK1FLCmdRbwQ	6468
c6f8wBjPLDzyubEBqgcMnw	5783
Z0JT0Ras-kMvJLMdn9iJQQ	2142
N95lG-T9bbR2iNL9_an1SQ	5036
85NJYsnJt076Nd0Kwslibg	4199
OJdufUU3hVabgvilBHksYw	7047
pOvTYClFgMm-wAXPW1CnhQ	5919
eQel7bUz75j0AVKnfsillg	3644
Zwg2TWwdm7y5ORK3yN6nDw	928
6qKjl6xJQzLb9ka-QbTudA	2017
LuLbt1cXEhcvyha00wCWZA	4067
PNzir9TtJAD7U41GwR98-w	263
HaRN97OjSnUJIHk21QAIXw	3451
paNjgpMDRhNzBj6xzhHPVQ	2839
YSAI_l2wLGfGEcLgxR09Ug	7312
Oc78Nz-sOyo16LYEl1AsZA	4905
epvLkQNL6MOvk3s6JlTntA	8034
hD77eYBzX0yQfpAUMY8dEw	604
DogtzlX8OqcsSigxqnQ9WQ	4600
CPgyfcsyPegLha1ajaSRoA	3450
TA1Q7RWVzMigi5X3FAmctw	177

User Mapping

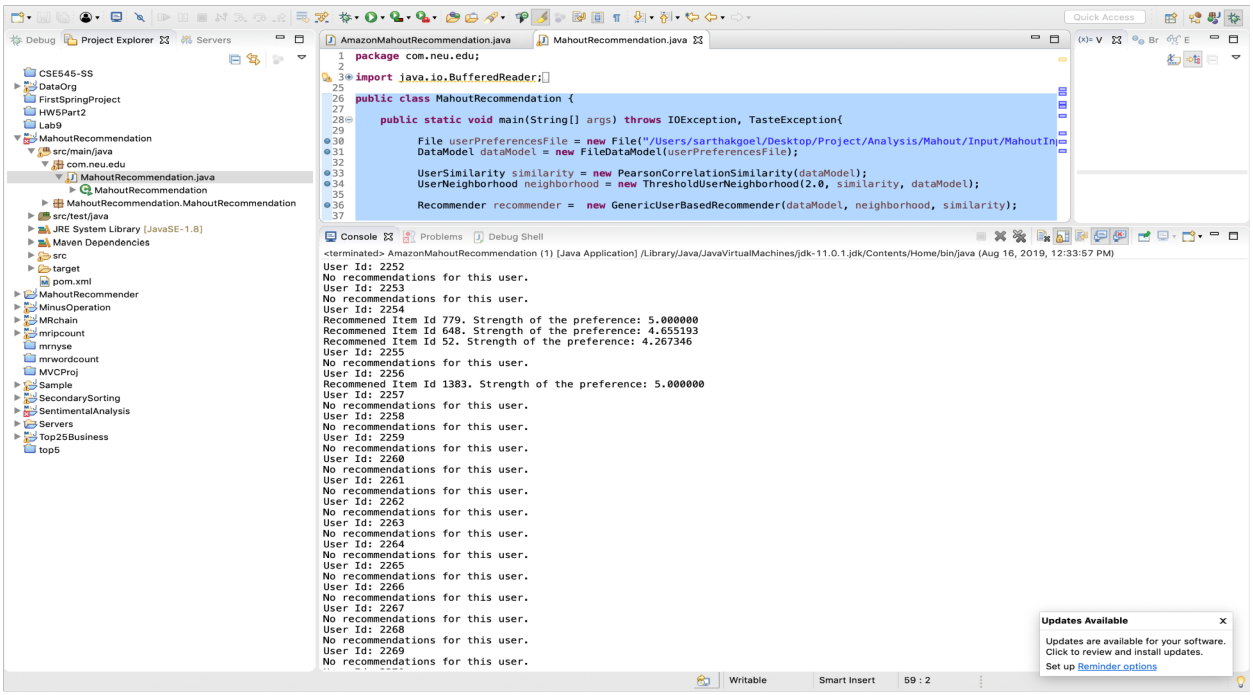
hXxN8h9_XXt4vqWVD6VJNg	87
m7UE-3UEVH0xzjzN9jZ2rQ	850
CVtF-KfhXlh5GOQAUDMTrQ	1822
vl1FXMGvlvA3wF78f6u-aw	979
1BKX1sVSaVpNaXMGxOyjUA	1579
iSvvhQDAL8T_bDheKaVgNA	1242
9b7zJ0opAxhqdqMKTAmgOA	1526
irykVoCgH393QZTXYbgdwA	568
07x1nr2hla5ODaJeLLEPIw	1324
NVoSQgeP-EEstYql2jAFEA	23
8Q15MkHpK9szYg_8WAaZSw	150
nxZm9O5yju_tSTj0sI6dqw	408
iew5lGWb-kGRRDSMii_QTw	644
0JYWsiBQfHnrEjKJhliULA	275
UKGVelzb9j2mJiLIBmLhrg	1903
Xz-ab3qbgiWgl2xFNt2DIQ	370
8VuoElbt6CUIs-IFg6ICWw	1676
xyVht40ggkb3xNf8AdQ5rA	654
5ehlqZy6iIjNbNB58Cf53A	1344
gu__VhXlhPsVGY4nmDu9lA	638
qTs_GibdRA1daay6D3rrbQ	795
haYMlQAttei_ZaIMDZZ3TQ	1562
QxeGfwMFcD8VL6-TwVTibg	606
RWXAq51RpvCNw8kYKK0BEw	1458
4sCNqow8kIlbIhzZN1CqYQ	2414
ltFjEnf5vN6m_Fxlwy6aZA	584
AGWHeXz_g9x9pBbfIJ0QDQ	1338
mmqmkfMhieEwFEOqx7h8QQ	2234
2l2rwgkdluT9JwP8R_Cvuw	2351
pk0cnv1-qslWGUSlKdRLqQ	1771

Mahout Input Passed to Mahout Recommendation Engine



1	1	5
2	2	5
2	3	5
2	4	4
2	5	4
3	6	5
4	7	4
4	8	4
4	9	3
5	10	5
6	11	2
7	12	2
8	13	5
9	14	5
10	15	1
10	16	3
10	17	4
10	18	1
10	19	4
10	20	5
10	21	4
10	22	4
10	23	2
11	24	4
11	25	4
11	26	5
11	27	3
11	28	1

Step 04 Mahout Recommendation



Conclusion:
I performed multiple analysis on this dataset by running Hadoop on single machine.

Apache Hive provides a SQL dialect that enables us to perform analysis on underlying data. These underlying queries are converted to MapReduce jobs and further you could store the output on HDFS.

On similar line, I used Apache Pig Latin and accessed the Pig using CLI and wrote Data Flow Queries to perform MapReduce operations on yelp dataset.

Apache Mahout is a powerful machine learning library for recommendation system, and I was able run the recommendations program over a large dataset very quickly.

Citations:

1. http://pig.apache.org/docs/latest/api/org/apache/pig/builtin/PigStorage.html
2. https://pig.apache.org/docs/r0.17.0/api/org/apache/pig/piggybank/storage/package-summary.html
3. https://pig.apache.org/docs/r0.16.0/udf.pdf
4. https://hive.apache.org/
5. https://medium.com/@sudarshan_sreenivasan/what-is-secondary-sort-in-hadoop-and-how-does-it-work-fe35609b5319
6. http://blog.ditullio.fr/2015/12/28/hadoop-basics-secondary-sort-in-mapreduce/
7. https://github.com/goelsarthak
8. https://github.com/AmiGandhi