INFO7250 Final Project Report: Yelp Dataset Analysis

Dataset Overview:

Source: https://www.yelp.com/dataset
Example entries: https://www.yelp.com/dataset/documentation/main

The Yelp dataset provides information on businesses, reviews, users, check-ins, and tips making it ideal for large-scale analysis. The dataset includes:

- **business.json:** This file has information about businesses, including location, categories, ratings and hours when it is open.
- **review.json:** It contains text reviews of businesses, with star ratings and votes (useful, funny, cool).
- **user.json:** This contains user profiles, review counts, compliment metrics and number of friend connections.
- **checkin.json:** Check-in data for businesses with timestamps.
- **tip.json:** Tips written by a user on a business.

Apache Hadoop Analysis:

1) **Sentiment Analysis:**
   The SentimentAnalysisMapper computes the sentiment scores for each review using the **AFINN-111** lexicon in the **DistributedCache**. While the BusinessInfoMapper extracts business details. Both mappers emit business_id as key, enabling the SentimentAnalysisReducer to perform a **reduce-side join** and aggregating the sentiment scores.

   Output:

```
zz0l4dUf28wzPAaTdGqsSw  El Pique - Wilmington | Wilmington, DE | Stars: 4.5 | Reviews: 181 | Normalized Sentiment Score: 8.03867403314917
zz18xL0peOpD-ENMEF6hcQ  Pretty Nails | Moorestown, NJ | Stars: 2.5 | Reviews: 13 | Normalized Sentiment Score: 4.153846153846154
zz3E7kmJI2r2JseE6LAnrw  Hung Vuong Super Market | Philadelphia, PA | Stars: 3.5 | Reviews: 103 | Normalized Sentiment Score: 8.747572815598
zz6_dk1S63QQNBSaq3iXEg  SPECS St. Petersburg Eye Care Specialists | St. Petersburg, FL | Stars: 5.0 | Reviews: 8 | Normalized Sentiment Score: 11.75
zzFCdBSW27eKFg-xG7cqAg  Suburbia Seafood | King of Prussia, PA | Stars: 4.0 | Reviews: 19 | Normalized Sentiment Score: 6.105263157894737
zzG-EObaHskhFLy6suavpA  G&H Tile Masters | Skippack, PA | Stars: 5.0 | Reviews: 6 | Normalized Sentiment Score: 11.333333333333334
zzHtFjfM7NvuVM1HTsCLGA  Nom Nom Japanese Kitchen | Paoli, PA | Stars: 4.5 | Reviews: 35 | Normalized Sentiment Score: 11.914285714285715
zzIF9qp2UoHN48EeZH_IDg  Domino's Pizza | Tarpon Springs, FL | Stars: 3.0 | Reviews: 20 | Normalized Sentiment Score: 1.45
zzKVqizyl3QCb3GZmg_cNg  New Castle Court House Museum | New Castle, DE | Stars: 4.5 | Reviews: 6 | Normalized Sentiment Score: 13.166666666666
zzO2zgfqP9ANmEWt-EZFWg  Seasons American Cuisine | Chesterfield, MO | Stars: 5.0 | Reviews: 13 | Normalized Sentiment Score: 28.6923076927693
zzQWjZ_1Dr7kkDYlk17qRw  The Stockyard | Holiday, FL | Stars: 2.5 | Reviews: 15 | Normalized Sentiment Score: 6.666666666666667
zzRZMOmhjgUbzZSSWJT5aw  Countryside Christian Church | Clearwater, FL | Stars: 4.5 | Reviews: 10 | Normalized Sentiment Score: 13.6
zzUxvYE-8Fj_dWqsPcagNg  The Cleaning Authority - Tucson | Tucson, AZ | Stars: 2.0 | Reviews: 26 | Normalized Sentiment Score: 2.6538461536153 7
zzW99n4VJr1Atte1Uhub1A  Atlas Transmissions | Tucson, AZ | Stars: 3.0 | Reviews: 15 | Normalized Sentiment Score: 5.533333333333333
zzXDi0Pdv0s84M-oQaIa_g  Radnor Lake State Park | Nashville, TN | Stars: 4.5 | Reviews: 211 | Normalized Sentiment Score: 10.4691943127962
zzXRdzrVhfNWPHD2MeyWeA  Royal Farms | Lansdale, PA | Stars: 2.0 | Reviews: 15 | Normalized Sentiment Score: 4.4
zzZqlYfZZIcN02C8SLcuBw  Skyline Printing | Tucson, AZ | Stars: 5.0 | Reviews: 5 | Normalized Sentiment Score: 6.8
zzbZtgPYZS8sTIWQH6DwEw  F & M Patio Bar | New Orleans, LA | Stars: 3.0 | Reviews: 87 | Normalized Sentiment Score: 3.103448275862069
zzbpcMZXHoZxUr9JZdH6wg  Eden Medical Spa | Wayne, PA | Stars: 1.5 | Reviews: 23 | Normalized Sentiment Score: -1.7391304347826086
zzfj1-iPfw0cwnOjY0yUgA  Ruby's Kitchen | Nashville, TN | Stars: 3.0 | Reviews: 25 | Normalized Sentiment Score: 7.68
zzg-Il9zxsaVXlCDrcG7hg  North End Organic Nursery | Garden City, ID | Stars: 5.0 | Reviews: 16 | Normalized Sentiment Score: 12.0625
zziDpuuJw-Km1J4BaGpBKA  Honey Baked Ham Company | St. Louis, MO | Stars: 3.5 | Reviews: 6 | Normalized Sentiment Score: 5.166666666666667
zzjCxn89a7RQo8keIOO_Ag  Thriftway | Pottstown, PA | Stars: 2.0 | Reviews: 5 | Normalized Sentiment Score: 1.8
zzjFdJwXuxBOGe9JeY_EMw  Taste of the Islands | Norristown, PA | Stars: 4.0 | Reviews: 48 | Normalized Sentiment Score: 4.645833333333333
zznJox6-nmXlGYNWgTDwQQ  Dunkin' | Clearwater, FL | Stars: 1.5 | Reviews: 30 | Normalized Sentiment Score: -0.1
zznZqH9CiAznbkV6fXyHWA  Que Pasta Nola | New Orleans, LA | Stars: 5.0 | Reviews: 12 | Normalized Sentiment Score: 14.25
zztOG2cKm87I6Iw_tleZsQ  Project Management Academy | King of Prussia, PA | Stars: 5.0 | Reviews: 6 | Normalized Sentiment Score: 8.5
zzu6_r3DxBJuXcjnOYVdTw  Cafe Diblasi | Gretna, LA | Stars: 3.5 | Reviews: 8 | Normalized Sentiment Score: 12.75
zzw66H6hVjXQEt0Js3Mo4A  Sullivan Farms Christmas Trees | Ballwin, MO | Stars: 3.5 | Reviews: 5 | Normalized Sentiment Score: 7.2
zzyx5x0Z7xXWWvWnZFuxlQ  Walnut Street Pizza | Philadelphia, PA | Stars: 2.5 | Reviews: 8 | Normalized Sentiment Score: 8.125
```

2) **Top Business Categories by State:**
This analysis identifies the top business categories in each state based on a composite score derived from review count and star ratings. The *StateCategoryKey* class defines a composite key combining state and composite score, so that we can perform **Secondary Sort**. *StatePartitioner* ensures all records for a state are sent to the same reducer, while *StateGroupingComparator* groups records by state.

Output:

```
~/INFO7250/final-project/Yelp-Analysis/SentimentAnalysis main
❯ hadoop fs -cat /yelp_output_2/part-r-00000/
2024-12-13T09:54:02,856 WARN [main] org.apache.hadoop.util.Na
ng builtin-java classes where applicable
AB       Restaurants | Composite Score: 341.55
AB       Cafes | Composite Score: 341.55
AB      Patisserie/Cake Shop | Composite Score: 341.55
AZ       Cocktail Bars | Composite Score: 1489.55
AZ       Nightlife | Composite Score: 1489.55
AZ       Bars | Composite Score: 1489.55
CA      Mexican | Composite Score: 2685.15
CA       Restaurants | Composite Score: 2685.15
CA      Cocktail Bars | Composite Score: 2059.20
CO       Restaurants | Composite Score: 9.45
CO      Buffets | Composite Score: 9.45
CO       Outdoor Gear | Composite Score: 8.20
DE       Cocktail Bars | Composite Score: 520.60
DE       Bars | Composite Score: 520.60
DE      Cajun/Creole | Composite Score: 520.60
FL       Nightlife | Composite Score: 2283.20
FL       Delis | Composite Score: 2283.20
FL      Burgers | Composite Score: 2283.20
HI       Pizza | Composite Score: 12.95
HI      Cosmetic Surgeons | Composite Score: 12.95
HI       Health & Medical | Composite Score: 12.95
ID       Burgers | Composite Score: 1268.20
ID      American (New) | Composite Score: 1268.20
ID       Restaurants | Composite Score: 1268.20
IL      Restaurants | Composite Score: 409.45
IL       Barbeque | Composite Score: 409.45
IL       Southern | Composite Score: 386.05
IN       Restaurants | Composite Score: 1564.45
IN       Comfort Food | Composite Score: 1564.45
IN      Nightlife | Composite Score: 1564.45
LA       Cajun/Creole | Composite Score: 5298.80
LA      Live/Raw Food | Composite Score: 5298.80
```

3) **User Engagement Analysis:**
This analysis identifies the top 10 most engaged users based on the engagement score derived from review count, fan count, and total compliments. The mapper processes user data, calculates the score and then emits the details. The reducer maintains a *TreeMap* to track the top users.

Output:

```
❯ hadoop fs -cat /yelp_output_3/part-r-00000/
2024-12-13T09:53:23,045 WARN [main] org.apache.hadoop.util.NativeCodeLoader - Unable to load native-hadoop library for your platform... us
ng builtin-java classes where applicable
Engagement Score: 98659.00      User ID: JjXuiru1_ONzDkYVrHN0aw, Name: Richard, Review Count: 1424, Total Compliments: 324328, Fans: 3243
Engagement Score: 84256.00      User ID: Tqm7Wu7IBJ1td3Ab5ZpUhw, Name: Brian, Review Count: 1996, Total Compliments: 277314, Fans: 319
Engagement Score: 46912.30      User ID: ax7SnXOTIpatbsmqHLqVow, Name: Rohlin, Review Count: 972, Total Compliments: 154351, Fans: 605
Engagement Score: 40948.30      User ID: --2vR0DIsmQ6WfcSzKWigw, Name: Harald, Review Count: 1534, Total Compliments: 133351, Fans: 880
Engagement Score: 40348.90      User ID: h4oOQdnfjpEHbygEJDsFbg, Name: John, Review Count: 1157, Total Compliments: 132032, Fans: 804
Engagement Score: 37962.20      User ID: NOUfyJW-BAo_-Cbfo8edww, Name: Lolia, Review Count: 2516, Total Compliments: 121940, Fans: 611
Engagement Score: 31827.70      User ID: Ggx8iUdJ7lsQSqXRuclXtg, Name: Nadine, Review Count: 3499, Total Compliments: 99694, Fans: 850
Engagement Score: 27090.30      User ID: HH7iiWvBqV-20lA7JlSRWQ, Name: Lyla, Review Count: 1660, Total Compliments: 87271, Fans: 395
Engagement Score: 26783.60      User ID: JRAy4P4op3PCISZaMRA9_w, Name: Carissa, Review Count: 463, Total Compliments: 88363, Fans: 216
Engagement Score: 26680.90      User ID: giJqvuH8EC62Hi5WboZ08w, Name: Dawne, Review Count: 123, Total Compliments: 87736, Fans: 1493
```

4) **Staff Activity Analysis:**
   This analysis tracks staff checkin patterns by analyzing the timestamps of past business checkins. The mapper processes each timestamp, extracting the business ID and day of the week, and emits these as keys with a count of one. The reducer aggregates the values to determine the total activity occurences for each business and day combination.

   Output:

```
zzjFdJwXuxBOGe9JeY_EMw:6        9
zzjFdJwXuxBOGe9JeY_EMw:7        4
zznJox6-nmXlGYNWgTDwQQ:1        10
zznJox6-nmXlGYNWgTDwQQ:2        11
zznJox6-nmXlGYNWgTDwQQ:3        10
zznJox6-nmXlGYNWgTDwQQ:4        7
zznJox6-nmXlGYNWgTDwQQ:5        9
zznJox6-nmXlGYNWgTDwQQ:6        8
zznJox6-nmXlGYNWgTDwQQ:7        12
zznZqH9CiAznbkV6fXyHWA:7        1
zzu6_r3DxBJuXcjnOYVdTw:1        5
zzu6_r3DxBJuXcjnOYVdTw:3        3
zzu6_r3DxBJuXcjnOYVdTw:4        4
zzu6_r3DxBJuXcjnOYVdTw:5        2
zzu6_r3DxBJuXcjnOYVdTw:6        1
zzu6_r3DxBJuXcjnOYVdTw:7        8
zzw66H6hVjXQEt0Js3Mo4A:1        1
zzw66H6hVjXQEt0Js3Mo4A:7        1
zzyx5x0Z7xXWWvWnZFuxlQ:3        1
```

Apache Pig Analysis:

Processing data using *Elephant Bird libraries* to extract data from JSON files.

1) **Average Review Count by State:**

This analysis identifies the top 10 states with the highest average review counts. Extracting key attributes like state and review counts. Businesses are grouped by state and the average review count for each state is calculated.

Output:

```
~/INF07250/final-project/Yelp-Analysis/SentimentAnalysi
❯ hadoop fs -cat /output/top_10_states/part-r-00000/
2024-12-13T09:55:56,664 WARN [main] org.apache.hadoop.u
ng builtin-java classes where applicable
LA,74.88673921805723
CA,65.27714779934654
NV,53.13674659753727
TN,49.61803251493033
PA,45.26543082934281
MO,44.341336021259046
FL,42.53421952145841
IN,42.01698230639282
SD,42.0
AZ,41.63024616626311
```

2) **Distribution of Business by Categories:**
This analysis identifies the top 10 business categories with the highest number of businesses. Extracting and splitting the categories field into individual entries. Businesses are grouped by category, and the number of businesses in each category is calculated.

Output:

```
~/INF07250/final-project/Yelp-Analysis/SentimentAnalysis main*
❯ hadoop fs -cat /output/top_10_categories/part-r-00000/
2024-12-13T09:57:03,780 WARN [main] org.apache.hadoop.util.Nativ
ng builtin-java classes where applicable
Restaurants,15290
Food,6783
Shopping,5480
Beauty & Spas,4385
Home Services,3793
Automotive,3449
Health & Medical,3058
Local Services,2642
Nightlife,2291
Event Planning & Services,2067
```

3) **Distribution of Review Ratings:**

This analysis examines the distribution of review ratings across different star levels. It extracts key attributes like star ratings and review counts. Reviews are grouped by their star ratings, and the total number of reviews for each rating is calculated.

Output:

```
~/INF07250/final-project/Yelp-Analysis/SentimentAnalysis main*
❯ hadoop fs -cat /output/ordered_star_ratings/part-r-00000/
2024-12-13T09:57:46,003 WARN [main] org.apache.hadoop.util.Nat
ng builtin-java classes where applicable
1.0,1069561
2.0,544240
3.0,691934
4.0,1452918
5.0,3231627
```

4) **Most Active Users:**
This analysis identifies the top 10 users who have written the most reviews. It extracts key attributes like user ID and review count. Reviews are grouped by user ID, and the total number of reviews for each user is calculated.

Output:

```
~/INF07250/final-project/Yelp-Analysis/SentimentAnalysis
❯ hadoop fs -cat /output/top_10_users/part-r-00000
2024-12-13T09:58:34,680 WARN [main] org.apache.hadoop.ut
ng builtin-java classes where applicable
_BcWyKQL16ndpBdggh2kNA,3048
Xw7ZjaGfr0WNVt6s_5KZfA,1840
0Igx-a1wAstiBDerGxXk2A,1747
-G7Zkl1wIWBBmD0KRy_sCw,1682
ET8n-r7glWYqZhuR6GcdNw,1653
bYENop4BuQepBjM1-BI3fA,1578
1HM81n6n4iPIFU5d2Lokhw,1554
fr1Hz2acAb30aL3l6DyKNg,1447
wXdbkFZsfDR7utJvbWElyA,1396
Um5bfs5DH6eizgjH3xZsvg,1391
```

Apache Hive Analysis:

1) **Number of Businesses by City and State:**
In this analysis I have summarized the number of businesses in each city and state. The query groups data from the business table by state and city and calculates the count of businesses for each group.

Output:

NJ	Westampton	36
NJ	Westampton Township	1
FL	Westchase	17
IN	Westfield	1
NJ	Westhampton	1
NJ	Westmont	31
NJ	Westmont - Haddon Towsship	1
PA	Westtown	2
NJ	Westville	31
LA	Westwego	56
TN	White House	74
TN	Whitehouse	1
IN	Whiteland	18
TN	Whites Creek	7
IN	Whitestown	45
DE	Wilimington	1
PA	Williamsport	1
NJ	Williamstown	112
NJ	Willingboro	63
NJ	Willingboro Township	3
PA	Willow Grove	311
DE	Wilmington	1445
PA	Wilmington	1
DE	Wilmington	2
DE	Wilmington Manor	3
FL	Wimauma 23	
MO	Winchester	2
NJ	Winslow 3	
NJ	Winslow Township	6
DE	Winterthur	1
IL	Wood River	42
PA	Woodbourne	1
NJ	Woodbury	171

2) **Top 10 Most Reviewed Businesses Per State:**
This analysis identifies the top 10 most reviewed businesses in each state. Using the ROW_NUMBER() function, I ranked businesses within each state based on their review_count in descending order. The results were filtered to include only businesses with a rank of 10 or less and stored in a table.

Output:

```
AB      Esf3-D_44pArPd9GqysoCg  Duchess Bake Shop       486
AB      fM6XQeGW70a4EFbjqAAy3Q  SugarBowl       435
AB      jEvoDXtF3xMJsiPPbqGSmw  Tres Carnales Taqueria  429
AB      pqZYSvF_qrmCjkr6frf4RQ  Meat    420
AB      hc0KmC42EzjXbC83eMIDMg  Padmanadi Vegetarian Restaurant 271
AB      HCBSkwnMPHuOVKrTuGd6DA  The Next Act    261
AB      F0XiRcSbcLF4GwA2A2TDKQ  El Cortez Mexican Kitchen + Tequila Bar 258
AB      tt4lgEAXQcS_Cdy8cbb3rw  Blue Plate Diner        237
AB      f9_TLVlUHBv0869CygEbZg  Dadeo   237
AB      Eq5w0ZAW0PV30nNkJxJY_A  CRAFT Beer Market       226
AZ      UCMSWPqzXjd7QHq7v8PJjQ  Prep & Pastry   2126
AZ      LZzDvgfpkd4nI3E4L9wF1w  El Charro Cafe  1583
AZ      tV46IhCfHbsx_af-pMupiw  Cafe Poca Cosa  1306
AZ      zZ01WQlcpI1_n806WKV3bA  Culinary Dropout        1295
AZ      muxda1cSVtplETqTfYVgZA  HUB Restaurant & Ice Creamery   1270
AZ      WSx9-iYYyST_umny9sJBFg  The Parish      1210
AZ      Rv8bW3pkzpi5dZu5ckbgtA  Guadalajara Original Grill      1117
AZ      hyeo7JQr5uLp-St1MibYAA  Baja Cafe       1074
AZ      j8feOxyJqlIJWOi8su2qzw  Serial Grillers 986
AZ      wj8XtPyuREj8_0GQz3LZ6w  Wild Garlic Grill       968
CA      yPSejq3_erxo9zdVYTBnZA  Los Agaves      3834
CA      U3grYFIeu6RgAAQgdriHww  Brophy Bros - Santa Barbara     2940
CA      skY6r8WAkYqpV7_TxNm23w  Boathouse at Hendry's Beach     2536
CA      SZU9c8V2GuREDN5KgyHFJw  Santa Barbara Shellfish Company 2404
CA      GuzbBFraIq-fbkjfvaTRvg  Mesa Verde      1796
CA      6RBZfirnzE4NahJTn1UPNA  La Super-Rica Taqueria  1759
CA      EtM079Cj7-B3G7jPsGYb_Q  Sandbar 1546
CA      znlrxrY4InSx7ekPZxoHRw  McConnell's Fine Ice Creams     1537
CA      oGDGlUbOjHxmmCh8ZYcDCg  The Lark        1520
CA      edJoBsse6nsF0BYh6pATAg  The Palace Grill        1500
CO      1H9WYGFJ4AFwXq7WCOOMtg  Two Elk Restaurant      12
CO      g0mm9BzcG7AdpTbJf2gzgA  Breeze Ski Rentals      10
CO      C_k12ohqzRrF0P9aJVwocQ  Travis Darnell Photography      8
DE      UBX73ZWgCdgom4nv0UeGvg  Border Cafe     742
DE      S6ACtNZ5wY0EjmK07iWUgg  Iron Hill Brewery & Restaurant  512
DE      EP9K6qXOyHFq_0tp93wh9Q  Big Fish Grill - Riverfront     456
DE      luUGLqA3Gxv5raCJOAzblw  Firebirds Wood Fired Grill      430
DE      yK8LWJ4VRTYEoGxVYzcjQg  Metro Diner     381
DE      ujUb29BOYcUyhMcnqo4bWA  Two Stones Pub  328
DE      L7r4uOZT58z1Vv-rGlSZBg  Scrumptious     305
DE      qFsh80AAL90tkOc0n98bqg  El Diablo Burritos      303
DE      R5yCU9dJnGcXw3voSB7pKg  Del Pez Mexican Gastropub       290
DE      CFBnmxqAm4vX6v4BFye70w  Banks' Seafood Kitchen and Raw Bar      287
FL      QHWYlmVbLC3K6eglWoHVvA  Datz    3260
FL      L5LLN0RafiV1Z9cddzvuCw  Ulele   3064
```

Source Code:

Hadoop MapReduce:

Sentiment Analysis:

SentimentAnalysisDriver.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.sentimentanalysis;

import java.net.URI;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.MultipleInputs;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

/**
 *
 * @author nehadevarapalli
 */
public class SentimentAnalysisDriver extends Configured implements Tool {
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new SentimentAnalysisDriver(), args);
        System.exit(res);
```

```
    }

    @Override
    public int run(String[] args) throws Exception {
        Configuration conf = getConf();

        if (args.length != 3) {
            System.err.println("Usage: SentimentAnalysis <reviews-input-path> <business-input-path>
<output-path>");
            System.exit(-1);
        }

        // Adding AFINN file to Distributed Cache
        DistributedCache.addCacheFile(new URI("/AFINN-111.txt"), conf);
        Job job = Job.getInstance(conf, "Sentiment Analysis and Business Info");

        job.setJarByClass(SentimentAnalysisDriver.class);

        job.setReducerClass(SentimentAnalysisReducer.class);

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

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

        MultipleInputs.addInputPath(job, new Path(args[0]), TextInputFormat.class,
SentimentAnalysisMapper.class); //Reviews Data
        MultipleInputs.addInputPath(job, new Path(args[1]), TextInputFormat.class,
BusinessInfoMapper.class); //Business Data

        FileOutputFormat.setOutputPath(job, new Path(args[2]));
```

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

        return 0;
    }
}
```

SentimentAnalysisMapper.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.sentimentanalysis;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.URI;
import java.util.HashMap;

import org.apache.hadoop.filecache.DistributedCache;
import org.apache.hadoop.fs.FSDataInputStream;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.json.JSONObject;

/**
 *
 * @author nehadevarapalli
 */
```

```
public class SentimentAnalysisMapper extends Mapper<LongWritable, Text, Text, Text> {
    private final Text businessId = new Text();
    private final Text sentimentScore = new Text();
    private final HashMap<String, Integer> afinnMap = new HashMap<>();
    private URI[] files;

    @Override
    protected void setup(Context context) throws IOException {
        files = DistributedCache.getCacheFiles(context.getConfiguration());
        System.out.println("files:"+ files);
        Path path = new Path(files[0]);
        FileSystem fs = FileSystem.get(context.getConfiguration());
        FSDataInputStream in = fs.open(path);
        BufferedReader br = new BufferedReader(new InputStreamReader(in));
        String line = "";
        while ((line = br.readLine()) != null) {
            String parts[] = line.split("\t");
            if (parts.length == 2) {
            afinnMap.put(parts[0].toLowerCase(), Integer.parseInt(parts[1]));
            }
        }
        br.close();
        in.close();
    }

    @Override
    protected void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
        String businessIdStr;
        String reviewText;
        String line = value.toString();
        try{
            JSONObject obj = new JSONObject(line);
```

```
            businessIdStr = obj.getString("business_id");
            reviewText = obj.getString("text");

            businessId.set(businessIdStr);
            int score = 0;

            for (String word : reviewText.toLowerCase().replaceAll("[^a-zA-Z ]", "").split("\\s+")) {
                score += afinnMap.getOrDefault(word, 0);
            }

            sentimentScore.set("SCORE: " + score);
            context.write(businessId, sentimentScore);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

BusinessInfoMapper.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.sentimentanalysis;

import java.io.IOException;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.json.JSONObject;

/**
 *
```

```
 * @author nehadevarapalli
 */
public class BusinessInfoMapper extends Mapper<Object, Text, Text, Text> {
    private final Text businessId = new Text();
    private final Text businessInfo = new Text();

    @Override
    protected void map(Object key, Text value, Context context) throws IOException, InterruptedException
{
        String line = value.toString();
        try {
            JSONObject obj = new JSONObject(line);
            String businessIdStr = obj.getString("business_id");
            String businessName = obj.getString("name");
            String businessCity = obj.getString("city");
            String businessState = obj.getString("state");
            double stars = obj.getDouble("stars");
            int reviewCount = obj.getInt("review_count");

            businessId.set(businessIdStr);
            businessInfo.set("INFO:" + businessName + " | " + businessCity + ", " + businessState + " | Stars:
" + stars);
            context.write(businessId, businessInfo);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

SentimentAnalysisReducer.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
```

```java
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.sentimentanalysis;

import java.io.IOException;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nehadevarapalli
 */
public class SentimentAnalysisReducer extends Reducer<Text, Text, Text, Text> {
    private final Text result = new Text();

    @Override
    protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {
        String businessInfo = null;
        int totalSentimentScore = 0;
        int reviewCount = 0;

        for (Text val : values) {
            String value = val.toString();
            if (value.startsWith("INFO:")) {
                businessInfo = value.substring(5);
            } else if (value.startsWith("SCORE:")) {
                totalSentimentScore += Integer.parseInt(value.substring(6).trim());
                reviewCount++;
            }
        }
```

```
        if (businessInfo != null && reviewCount > 0) {
            double normalizedScore = (double) totalSentimentScore / reviewCount;
            result.set(businessInfo
                    + " | Reviews: " + reviewCount
                    + " | Normalized Sentiment Score: " + normalizedScore);
            context.write(key, result);
        }
    }
}
```

Top Business Categories by State:

TopBusinessCategoriesByLocation.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 */

package com.mycompany.topbusinesscategoriesbylocation;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

/**
 *
 * @author nehadevarapalli
```

```
*/
public class TopBusinessCategoriesByLocation extends Configured implements Tool {
   public static void main(String[] args) throws Exception {
      int res = ToolRunner.run(new Configuration(), new TopBusinessCategoriesByLocation(), args);

   }

   @Override
   public int run(String[] args) throws Exception{
      Configuration conf = getConf();

      if (args.length != 2) {
         System.err.println("Usage: TopBusinessCategories <business-input-path> <output-path>");
         System.exit(-1);
      }

      Job job = Job.getInstance(conf, "Top Business Categories by Location");

      job.setJarByClass(TopBusinessCategoriesByLocation.class);
      job.setMapperClass(BusinessMapper.class);
      job.setReducerClass(CategoryReducer.class);

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

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

      job.setPartitionerClass(StatePartitioner.class);
      job.setGroupingComparatorClass(StateGroupingComparator.class);
      job.setSortComparatorClass(StateCategoryKeyComparator.class);

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

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

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

        return 0;
    }
}
```

BusinessMapper.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.topbusinesscategoriesbylocation;

import java.io.IOException;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.json.JSONObject;

/**
 *
 * @author nehadevarapalli
 */

// Mapper to process business.json
public class BusinessMapper extends Mapper<LongWritable, Text, StateCategoryKey, Text> {
    @Override
    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
        try {
```

```
            JSONObject business = new JSONObject(value.toString());

            // Extracting location information
            String state = business.optString("state", "Unknown").trim();
            String categoriesStr = business.optString("categories", null);

            int reviewCount = business.optInt("review_count", 0);
            double stars = business.optDouble("stars", 0.0);
            double compositeScore = (0.7 * reviewCount) + (0.3 * stars);

            if (categoriesStr != null && !categoriesStr.isEmpty()) {
                String[] categories = categoriesStr.split(",");
                for (String category : categories) {
                    context.write(new StateCategoryKey(state, compositeScore), new Text(category));
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

StateCategoryKey.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.topbusinesscategoriesbylocation;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
```

```java
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.WritableComparable;

public class StateCategoryKey implements WritableComparable<StateCategoryKey> {
    private Text state;
    private Double compositeScore;

    public StateCategoryKey() {
        this.state = new Text();
        this.compositeScore = 0.0;
    }

    public StateCategoryKey (String state, double compositeScore) {
        this.state = new Text(state);
        this.compositeScore = compositeScore;
    }

    @Override
    public void write(DataOutput out) throws IOException {
        state.write(out);
        out.writeDouble(compositeScore);
    }

    @Override
    public void readFields(DataInput in) throws IOException {
        state.readFields(in);
        compositeScore = in.readDouble();
    }

    @Override
    public int compareTo(StateCategoryKey o) {
        int cmp = this.state.compareTo(o.state);
        if (cmp == 0) {
```

```
            return -Double.compare(this.compositeScore, o.compositeScore); //Descending by score
        }
        return cmp;
    }

    public Text getState() { return state; }
    public double getCompositeScore() { return compositeScore; }
}
```

StateCategoryKeyComparator.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.topbusinesscategoriesbylocation;

import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.io.WritableComparator;

/**
 *
 * @author nehadevarapalli
 */
public class StateCategoryKeyComparator extends WritableComparator {
    protected StateCategoryKeyComparator() {
        super(StateCategoryKey.class, true);
    }

    @Override
    public int compare(WritableComparable w1, WritableComparable w2) {
        StateCategoryKey k1 = (StateCategoryKey) w1;
        StateCategoryKey k2 = (StateCategoryKey) w2;
```

```
        int cmp = k1.getState().compareTo(k2.getState());
        if (cmp == 0) {
            return -Double.compare(k1.getCompositeScore(), k2.getCompositeScore()); // Descending by
score
        }
        return cmp; // Ascending by state
    }
}
```

StateGroupingComparator.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.topbusinesscategoriesbylocation;

import org.apache.hadoop.io.WritableComparable;
import org.apache.hadoop.io.WritableComparator;

/**
 *
 * @author nehadevarapalli
 */
public class StateGroupingComparator extends WritableComparator {
    protected StateGroupingComparator() {
        super(StateCategoryKey.class, true);
    }

    @Override
    public int compare(WritableComparable w1, WritableComparable w2) {
        StateCategoryKey k1 = (StateCategoryKey) w1;
        StateCategoryKey k2 = (StateCategoryKey) w2;
```

```
        return k1.getState().compareTo(k2.getState());
    }
}
```

StatePartitioner.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.topbusinesscategoriesbylocation;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Partitioner;

/**
 *
 * @author nehadevarapalli
 */
public class StatePartitioner extends Partitioner<StateCategoryKey, Text> {
    @Override
    public int getPartition(StateCategoryKey key, Text value, int numPartitions) {
        return (key.getState().hashCode() & Integer.MAX_VALUE) % numPartitions;
    }
}
```

CategoryReducer.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.topbusinesscategoriesbylocation;
```

```java
import java.io.IOException;
import java.util.HashMap;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nehadevarapalli
 */
public class CategoryReducer extends Reducer<StateCategoryKey, Text, Text, Text> {
    @Override
    public void reduce (StateCategoryKey key, Iterable<Text> values, Context context) throws
IOException, InterruptedException {
        HashMap<String, Integer> stateCount = new HashMap<>();
        Text currentState = key.getState();
        int categoryCount = 0;

        for (Text category : values) {
            if (categoryCount < 3) {
                String formattedScore = String.format("%.2f", key.getCompositeScore());
                context.write(new Text(currentState), new Text(category.toString() + " | Composite Score: " +
formattedScore));
                categoryCount++;
            } else {
                break;
            }
        }
    }
}
```

User Engagement Analysis:

UserEngagement.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.userengagement;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

/**
 *
 * @author nehadevarapalli
 */
public class UserEngagement {
    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.err.println("Usage: UserEngagementDriver <input path> <output path>");
            System.exit(-1);
        }

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "User Engagement Analysis");

        job.setJarByClass(UserEngagement.class);
        job.setMapperClass(UserEngagementMapper.class);
        job.setReducerClass(UserEngagementReducer.class);
```

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

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

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

UserEngagementMapper.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.userengagement;

import java.io.IOException;

import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.json.JSONObject;

/**
 *
 * @author nehadevarapalli
 */
public class UserEngagementMapper extends Mapper<LongWritable, Text, Text, Text> {
    @Override
```

```
    protected void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
        try {
            JSONObject jsonObject = new JSONObject(value.toString());

            String userId = jsonObject.getString("user_id");
            String name = jsonObject.optString("name", "Unknown");
            int reviewCount = jsonObject.optInt("review_count", 0);
            int fans = jsonObject.optInt("fans", 0);

            // Calculate total compliments
            int totalCompliments = 0;
            for (String field : jsonObject.keySet()) {
                if (field.startsWith("compliment_")) {
                    totalCompliments += jsonObject.optInt(field, 0);
                }
            }

            // Compute engagement score
            double engagementScore = 0.5 * reviewCount + 0.3 * totalCompliments + 0.2 * fans;

            String userInfo = "User ID: " + userId + ", Name: " + name + ", Review Count: " + reviewCount + ",
Total Compliments: " + totalCompliments + ", Fans: " + fans;
            context.write(new Text(String.valueOf(engagementScore)), new Text(userInfo));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

UserEngagementReducer.java

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.userengagement;

import java.io.IOException;
import java.util.TreeMap;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

/**
 *
 * @author nehadevarapalli
 */
public class UserEngagementReducer extends Reducer<Text, Text, Text, Text> {
    private TreeMap<Double, String> topUsersMap = new TreeMap<>();

    @Override
    protected void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {
        for (Text value : values) {
            double score = Double.parseDouble(key.toString());
            topUsersMap.put(score, value.toString());

            // Keep only top N users in the TreeMap
            if (topUsersMap.size() > 10) {
                topUsersMap.remove(topUsersMap.firstKey());
            }
        }
    }
```

```
    @Override
    protected void cleanup(Context context) throws IOException, InterruptedException {
        for (Double score : topUsersMap.descendingKeySet()) {
            String userInfo = topUsersMap.get(score);
            context.write(new Text(String.format("Engagement Score: %.2f", score)), new Text(userInfo));
        }
    }
}
```

Staff Activity Analysis:

StaffActivityDriver.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.staffactivitydriver;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.conf.Configured;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.util.Tool;
import org.apache.hadoop.util.ToolRunner;

/**
 *
 * @author nehadevarapalli
 */
```

```java
public class StaffActivityDriver extends Configured implements Tool {
    public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new StaffActivityDriver(), args);
        System.exit(res);
    }

    @Override
    public int run(String[] args) throws Exception {
        if (args.length != 2) {
            System.err.println("Usage: StaffActivity <input-path> <output-path>");
            System.exit(-1);
        }

        Configuration conf = getConf();
        Job job = Job.getInstance(conf, "Staff Activity Analysis");

        job.setJarByClass(StaffActivityDriver.class);
        job.setMapperClass(StaffActivityMapper.class);
        job.setReducerClass(StaffActivityReducer.class);

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

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

        TextInputFormat.addInputPath(job, new Path(args[0]));
        TextOutputFormat.setOutputPath(job, new Path(args[1]));

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

StaffActivityMapper.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.staffactivitydriver;

import java.io.IOException;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.json.JSONObject;

/**
 *
 * @author nehadevarapalli
 */
public class StaffActivityMapper extends Mapper<Object, Text, Text, IntWritable> {
    private final Text businessDayHourKey = new Text();
    private final IntWritable one = new IntWritable(1);
    private final SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

    @Override
    protected void map(Object key, Text value, Context context) throws IOException, InterruptedException
{
        try {
            JSONObject obj = new JSONObject(value.toString());
            String businessId = obj.getString("business_id");
            String date = obj.getString("date");
```

```
            String[] timestamps = date.split(", ");

            for (String timestamp : timestamps) {
                Calendar calendar = Calendar.getInstance();
                calendar.setTime(dateFormat.parse(timestamp));

                // Get the day of the week (1 = Sunday, 7 = Saturday)
                int dayOfWeek = calendar.get(Calendar.DAY_OF_WEEK);

                // Create the key using businessId, dayOfWeek
                businessDayHourKey.set(businessId + ":" + dayOfWeek);

                context.write(businessDayHourKey, one);
            }
        } catch (ParseException e) {
            e.printStackTrace();
        }
    }
}
```

StaffActivityReducer.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.staffactivitydriver;

import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
```

```java
/**
 *
 * @author nehadevarapalli
 */
public class StaffActivityReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
    private final IntWritable result = new IntWritable();

    @Override
    protected void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,
InterruptedException {
        int totalCount = 0;

        for (IntWritable val : values) {
            totalCount += val.get();
        }

        result.set(totalCount);
        context.write(key, result);
    }
}
```

Pig Analysis

Average Review Count by State:

average_review_count_by_state.pig

```
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-
core-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-
hadoop-compat-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-
pig-4.17.jar;
```

```
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/json-simple-1.1.1.jar;

-- Load data
raw_data = load '/yelp_data/yelp_academic_dataset_business.json' using com.twitter.elephantbird.pig.load.JsonLoader();
extracted_data = foreach raw_data generate (chararray)$0#'business_id' as business_id, (chararray)$0#'name' as name, (chararray)$0#'address' as address, (chararray)$0#'city' as city, (chararray)$0#'state' as state, (chararray)$0#'postal_code' as postal_code, (float)$0#'latitude' as latitude, (float)$0#'longitude' as longitude, (float)$0#'stars' as stars, (int)$0#'review_count' as review_count, (chararray)$0#'is_open' as is_open, (chararray)$0#'attributes' as attributes, (chararray)$0#'categories' as categories, (chararray)$0#'hours' as hours;

-- Group businesses by state
grouped_by_state = GROUP extracted_data BY state;

-- Calculate the average review count for each state
avg_reviews_by_state = FOREACH grouped_by_state GENERATE
    group AS state,
    AVG(extracted_data.review_count) AS avg_review_count;

-- Order states by average review count in descending order
ordered_states = ORDER avg_reviews_by_state BY avg_review_count DESC;

-- Display the top 10 states with the highest average review counts
top_10_states = LIMIT ordered_states 10;
DUMP top_10_states;

STORE top_10_states INTO '/output/top_10_states' USING PigStorage(',');
```

Distribution of Business by Categories:

distribution_of_business_by_category.pig

```
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-core-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-hadoop-compat-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-pig-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/json-simple-1.1.1.jar;

-- Load data
raw_data = load '/yelp_data/yelp_academic_dataset_business.json' using com.twitter.elephantbird.pig.load.JsonLoader();
extracted_data = foreach raw_data generate (chararray)$0#'business_id' as business_id, (chararray)$0#'name' as name, (chararray)$0#'address' as address, (chararray)$0#'city' as city, (chararray)$0#'state' as state, (chararray)$0#'postal_code' as postal_code, (float)$0#'latitude' as latitude, (float)$0#'longitude' as longitude, (float)$0#'stars' as stars, (int)$0#'review_count' as review_count, (chararray)$0#'is_open' as is_open, (chararray)$0#'attributes' as attributes, (chararray)$0#'categories' as categories, (chararray)$0#'hours' as hours;

-- Split the categories field into individual categories
categories_split = FOREACH extracted_data GENERATE
    business_id,
    FLATTEN(STRSPLIT(categories, ',')) AS category;

-- Group by category
grouped_by_category = GROUP categories_split BY category;

-- Count the number of businesses in each category
business_count_by_category = FOREACH grouped_by_category GENERATE
    group AS category,
    COUNT(categories_split) AS business_count;

-- Order categories by business count in descending order
```

```
ordered_categories = ORDER business_count_by_category BY business_count DESC;

-- Display the top 10 categories with the most businesses
top_10_categories = LIMIT ordered_categories 10;
DUMP top_10_categories;

STORE top_10_categories INTO '/output/top_10_categories' USING PigStorage(',');
```

Distribution of Review Ratings:

distribution_of_review_ratings.pig:

```
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-core-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-hadoop-compat-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-pig-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/json-simple-1.1.1.jar;

-- Load data
raw_data = load '/yelp_data/yelp_academic_dataset_review.json' using com.twitter.elephantbird.pig.load.JsonLoader();
extracted_data = foreach raw_data generate (chararray)$0#'review_id' as review_id, (chararray)$0#'user_id' as user_id, (chararray)$0#'business_id' as business_id, (float)$0#'stars' as stars, (int)$0#'useful' as useful, (int)$0#'funny' as funny, (int)$0#'cool' as cool, (chararray)$0#'text' as text, (chararray)$0#'date' as date;

-- Group reviews by star rating
grouped_by_stars = GROUP extracted_data BY stars;

-- Count the number of reviews for each star rating
review_count_by_stars = FOREACH grouped_by_stars GENERATE
```

```
    group AS star_rating,
    COUNT(extracted_data) AS review_count;

-- Order star ratings in ascending order
ordered_star_ratings = ORDER review_count_by_stars BY star_rating ASC;

-- Display the distribution of review ratings
DUMP ordered_star_ratings;

STORE ordered_star_ratings INTO '/output/ordered_star_ratings' USING PigStorage(',');
```

Most Active Users:

most_active_users.pig

```
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-core-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-hadoop-compat-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/elephant-bird-pig-4.17.jar;
REGISTER /Users/nehadevarapalli/INFO7250/final-project/Yelp-Analysis/elephantBirdLib/json-simple-1.1.1.jar;

-- Load data
raw_data = load '/yelp_data/yelp_academic_dataset_review.json' using com.twitter.elephantbird.pig.load.JsonLoader();
extracted_data = foreach raw_data generate (chararray)$0#'review_id' as review_id,
(chararray)$0#'user_id' as user_id, (chararray)$0#'business_id' as business_id,
(float)$0#'stars' as stars, (int)$0#'useful' as useful, (int)$0#'funny' as funny, (int)$0#'cool' as cool,
(chararray)$0#'text' as text,
(chararray)$0#'date' as date;

-- Group reviews by user_id
grouped_by_user = GROUP extracted_data BY user_id;
```

```
-- Count the number of reviews per user
review_count_per_user = FOREACH grouped_by_user GENERATE
    group AS user_id,
    COUNT(extracted_data) AS review_count;

-- Order users by review count in descending order
ordered_users = ORDER review_count_per_user BY review_count DESC;

-- Display the top 10 most active users
top_10_users = LIMIT ordered_users 10;
DUMP top_10_users;

STORE top_10_users INTO '/output/top_10_users' USING PigStorage(',');
```

Hive Analysis

Number of Businesses by City and State:

number_of_businesses_each_city_and_state.hql

```
use my_database;
drop table if exists number_of_businesses_each_city_and_state;
create table number_of_businesses_each_city_and_state
row format delimited
fields terminated by ','
lines terminated by '\n'
as
select state, city, count(*) as count
from business
group by state, city;
select * from number_of_businesses_each_city_and_state;
```

Top 10 Most Reviewed Businesses per State:

top_10_reviewed_businesses.hql

```
-- Find the top 10 most reviewed businesses in each state
use my_database;
drop table if exists top10_reviewed_businesses_per_state;
CREATE TABLE top10_reviewed_businesses_per_state
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
AS
SELECT b.state, b.business_id, b.name, b.review_count
FROM (
    SELECT state, business_id, name, review_count,
        ROW_NUMBER() OVER (PARTITION BY state ORDER BY review_count DESC) AS rank
    FROM business
) b
WHERE b.rank <= 10;

select * from top10_reviewed_businesses_per_state;
```