PIG Assignment-

--LOAD jars

REGISTER 'hdfs://dumbo/user/as11445/elephant-bird-hadoop-compat-4.15.jar'

REGISTER 'hdfs://dumbo/user/as11445/elephant-bird-pig-4.15.jar'

REGISTER 'hdfs://dumbo/user/as11445/json-simple-1.1.1.jar'

REGISTER 'hdfs://dumbo/user/as11445/elephant-bird-core-4.15.jar'

REGISTER 'hdfs://dumbo/user/as11445/piggybank-0.15.0.jar'

--LOAD the json files

data = LOAD 'hdfs://dumbo/user/as11445/business.json' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') as (json:map[]);

data1= LIMIT data 1;

--dump data1;

review = LOAD 'hdfs://dumbo/user/as11445/review.json' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') as (rev:map[]);

rev1= LIMIT review 1;

--dump rev1;

userz= LOAD 'hdfs://dumbo/user/as11445/user.json' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') as (usr:map[]);

usr1= LIMIT userz 1;

--dump usr1;

uscity= LOAD 'hdfs://dumbo/user/as11445/file.json' USING com.twitter.elephantbird.pig.load.JsonLoader('-nestedLoad') as (uscity:map[]);

uscity1= LIMIT uscity 1;

--dump uscity1;

**-- Q1. Summarize the number of unique reviewers by US city, by business category.**

business\_data = FOREACH data GENERATE json#'city' as city, json#'categories' as categories, json#'business\_id' as business\_id;

review\_data = FOREACH review GENERATE rev#'business\_id' as business\_id,rev#'user\_id' as user\_id;

joined= JOIN business\_data by business\_id, review\_data by business\_id;

flatten\_business\_data = FOREACH joined GENERATE user\_id, city, FLATTEN(categories);

grouped\_business\_data = GROUP flatten\_business\_data BY (city,categories);

final\_df = FOREACH grouped\_business\_data GENERATE group.city as city , group.categories as category, COUNT(flatten\_business\_data.user\_id);

dump final\_df;

**--Q2. Rank all cities by # of stars descending, for each category**

business\_data = FOREACH data GENERATE json#'city' as city, json#'categories' as categories, json#'business\_id' as business\_id;

review\_data = FOREACH review GENERATE rev#'business\_id' as business\_id, (int)rev#'stars' as stars;

joined= JOIN business\_data by business\_id, review\_data by business\_id;

flatten\_data = FOREACH joined GENERATE stars, city, FLATTEN(categories);

grouped= GROUP flatten\_data by (categories, city);

res= FOREACH grouped GENERATE FLATTEN(group) AS (categories, city), AVG(flatten\_data.stars) AS rankings;

outputer= ORDER res BY categories, rankings DESC;

dump outputer;

**--Q3. What is the average rank (# stars) for businesses within 15 km of Edinburgh Castle, Scotland, by type of business (category)?**

dist = FOREACH data GENERATE (double)json#'stars' as stars, json#'latitude' as latitude, json#'longitude' as longitude, (double)json#'review\_count' as reviews, json#'categories' as categories:bag{a:tuple(b:chararray)} ;

dist1 = FOREACH dist GENERATE stars,latitude,longitude,reviews, FLATTEN(categories) as categories;

dist2 = FOREACH dist1 GENERATE \*, (111.045\* org.apache.pig.piggybank.evaluation.math.toDegrees(ACOS(COS(org.apache.pig.piggybank.evaluation.math.toRadians(55.9469753))\* COS(org.apache.pig.piggybank.evaluation.math.toRadians(latitude))\* COS(org.apache.pig.piggybank.evaluation.math.toRadians(-3.2096308) - org.apache.pig.piggybank.evaluation.math.toRadians(longitude))+ SIN(org.apache.pig.piggybank.evaluation.math.toRadians(55.9469753))\* SIN(org.apache.pig.piggybank.evaluation.math.toRadians(latitude))))) as distance:double;

dist3 = FILTER dist2 BY (distance<15);

dist4 = GROUP dist3 BY categories;

dist5 = FOREACH dist4 GENERATE group as category, AVG(dist3.stars);

dist5\_order= ORDER dist5 BY category;

dump dist5\_order;

**--Q4 Rank reviewers in Q3 by their number of reviews. For the top 10 reviewers, show their average number of stars, by category.**

users= FOREACH userz GENERATE usr#'user\_id' as user\_id, usr#'name' as name, (int)usr#'review\_count' as review\_count;

dist = FOREACH data GENERATE (double)json#'stars' as stars, json#'business\_id' as business\_id, json#'latitude' as latitude, json#'longitude' as longitude, (double)json#'review\_count' as reviews, json#'categories' as categories:bag{a:tuple(b:chararray)} ;

revs= FOREACH review GENERATE rev#'business\_id' as business\_id, (int)rev#'stars' as stars, rev#'business\_id' as business\_id, rev#'user\_id' as user\_id;

dist1 = FOREACH dist GENERATE business\_id,stars,latitude,longitude,reviews, FLATTEN(categories) as categories;

dist2 = FOREACH dist1 GENERATE \*, (111.045\* org.apache.pig.piggybank.evaluation.math.toDegrees(ACOS(COS(org.apache.pig.piggybank.evaluation.math.toRadians(55.9469753))\* COS(org.apache.pig.piggybank.evaluation.math.toRadians(latitude))\* COS(org.apache.pig.piggybank.evaluation.math.toRadians(-3.2096308) - org.apache.pig.piggybank.evaluation.math.toRadians(longitude))+ SIN(org.apache.pig.piggybank.evaluation.math.toRadians(55.9469753))\* SIN(org.apache.pig.piggybank.evaluation.math.toRadians(latitude))))) as distance:double;

dist3 = FILTER dist2 BY (distance<15);

join\_3rev= JOIN dist3 by business\_id, revs by business\_id;

dist4 = GROUP dist3 BY categories;

dist5 = FOREACH dist4 GENERATE group as category, AVG(dist3.stars);

dump dist5;

**--Q5 For the top 10 and bottom 10 category Food businesses in Q3, (in terms of stars), summarize star rating for reviews in January through May only.**

df\_1 = FOREACH data GENERATE (double)json#'stars' as stars,json#'business\_id' as id, json#'latitude' as latitude, json#'longitude' as longitude, json#'categories' as categories:bag{a:tuple(b:chararray)} ;

df\_2 = FOREACH df\_1 GENERATE stars,id,latitude,longitude, FLATTEN(categories) as categories;

df\_3 = FOREACH df\_2 GENERATE \*, (111.045\* org.apache.pig.piggybank.evaluation.math.toDegrees(ACOS(COS(org.apache.pig.piggybank.evaluation.math.toRadians(55.9469753))\* COS(org.apache.pig.piggybank.evaluation.math.toRadians(latitude))\* COS(org.apache.pig.piggybank.evaluation.math.toRadians( -3.2096308) - org.apache.pig.piggybank.evaluation.math.toRadians(longitude))+ SIN(org.apache.pig.piggybank.evaluation.math.toRadians(55.9469753))\* SIN(org.apache.pig.piggybank.evaluation.math.toRadians(latitude))))) as distance:double;

df\_4 = FILTER df\_3 BY (distance<15) AND (categories=='Food');

df\_5 = GROUP df\_4 BY id;

df\_6 = FOREACH df\_5 GENERATE group as id, AVG(df\_4.stars) as stars;

dump df\_6;

df\_top = ORDER df\_6 BY stars DESC;

top10 = limit df\_top 10;

--dump top10;

df\_bottom= ORDER df\_6 BY stars ASC;

bottom10 = limit df\_bottom 10;

--dump bottom10;

review1= FOREACH review GENERATE rev#'stars' as stars, rev#'business\_id' as id, rev#'date' as date;

total\_df = UNION top10, bottom10;

dump total\_df;

joined\_review = JOIN review1 by id, total\_df by id;

final\_required\_data = FOREACH joined\_review GENERATE total\_df::id as bid, (double)review1::stars as star,SUBSTRING(review1::date,5,7) as month;

final\_data\_by\_month = FILTER final\_required\_data BY (month matches '01|02|03|04|05');

--dump final\_data\_by\_month;

grouped\_data\_by\_business = GROUP final\_data\_by\_month by bid;

avg\_rating = FOREACH grouped\_data\_by\_business GENERATE group, AVG(final\_data\_by\_month.star) as avg\_stars;

dump avg\_rating;