Contributors

Member 1

- Name : Kavin Kumar Arumugam
- Class Id : 1

Member 2

- Name : Alper Erel
- Class Id : 8

Member 3

- Name : Jayden Tran
- Class Id : 27

Table of Contents:

Objectives:

- ### Question-1: Use Case: Hadoop Map-Reduce Algorithm:

Finding Facebook common friends: Facebook has an archive of friends (Consider that if "A" is a friend of "B", that means "B" is a friend of "A"). Assume disk space is not a constraint because they execute potentially thousands of requests everyday. They want to do the calculations beforehand so that the processing time for the future requests will be significantly decreased. One of the most common processing demand is the feature that shows number of mutual friends. When you look at somebody's profile, you see the list for the mutual friends. We will be using the Map-Reduce algorithm so that we will have the data for mutual friends at the beginning of each day and have the results ready. Afterwards, it's just a rapid search.

- ### Question-2: Use Case: Use Map-Reduce algorithm to analyze You Tube data set given.

Task 1: Determine the top 5 categories that has the maximum number of videos.
Task 2: Determine the top 10 high-rated videos on You tube.

- ### Question-3: Use Case: Hive and Solr

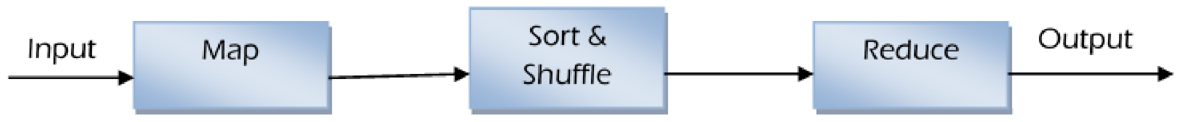
1. Hive Usecase
a. Create a new Hive Table and also include Complex Data Types.
b. Use built-in functions from Hive.
c. Perform 10 queries on the dataset.
2. Solr Usecase
a. Create a new Solr Collection.
b. Perform 10 queries on the created collection.
c. Record the execution time for the created 10 queries.

Approaches/Methods:

- ### Question 1 & 2:

Map-Reduce algorithm is very advantageous when it comes to process extensive numbers of lines of data. It splits input task into littler and reasonable sub-assignments to manage them in parallel. Map-Reduce algorithm basically sends the processing node to where the data stands.

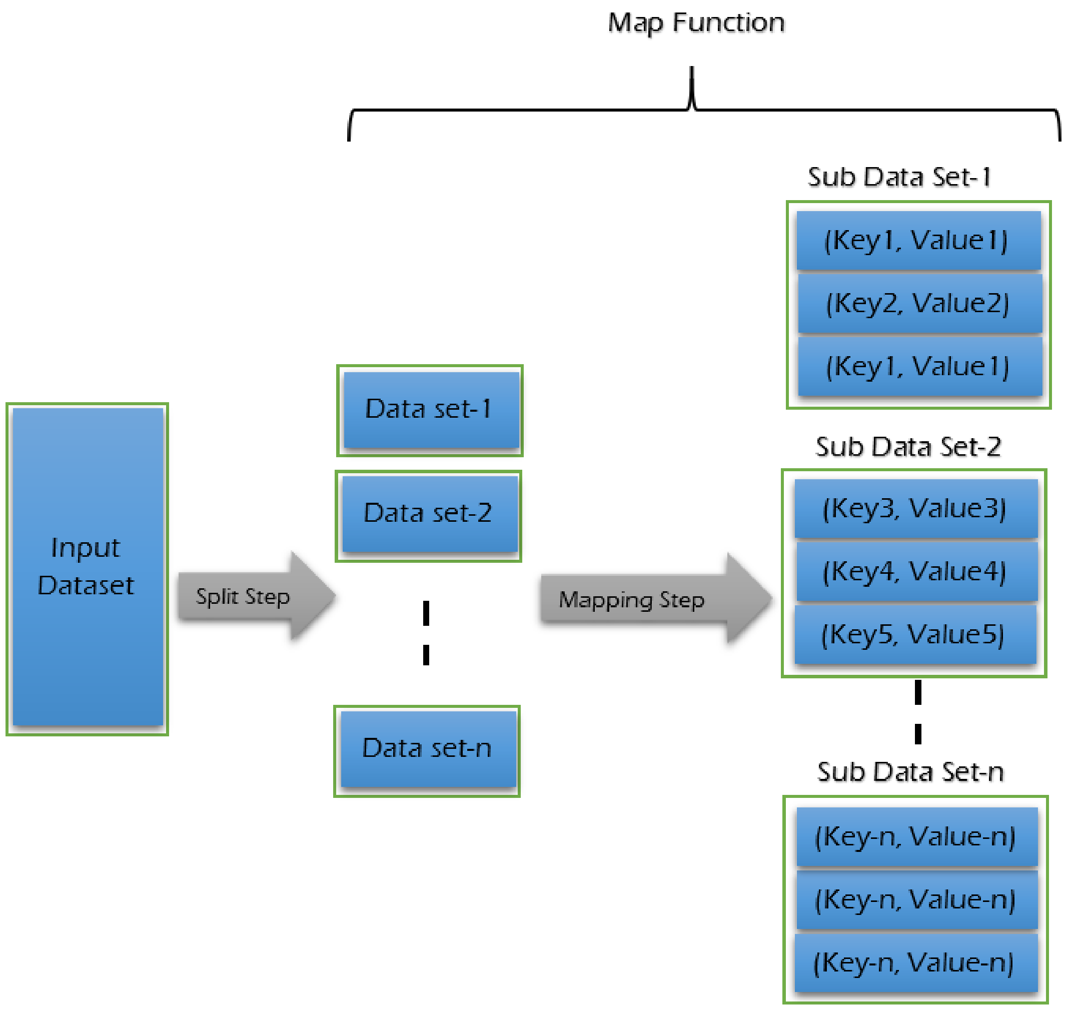
Map-Reduce Process Consists of 3 Stages:



1) Map Stage
Map Function is the initial phase in MapReduce Algorithm. At Map stage will take a shot at key and value sets as ianput.

A list of data is given to mapper class called mapper **Splitting** - Takes input dataset and divide the input dataset into small groups.
Mapping - Takes the splitted dataset and perform required computation or action on each of them.

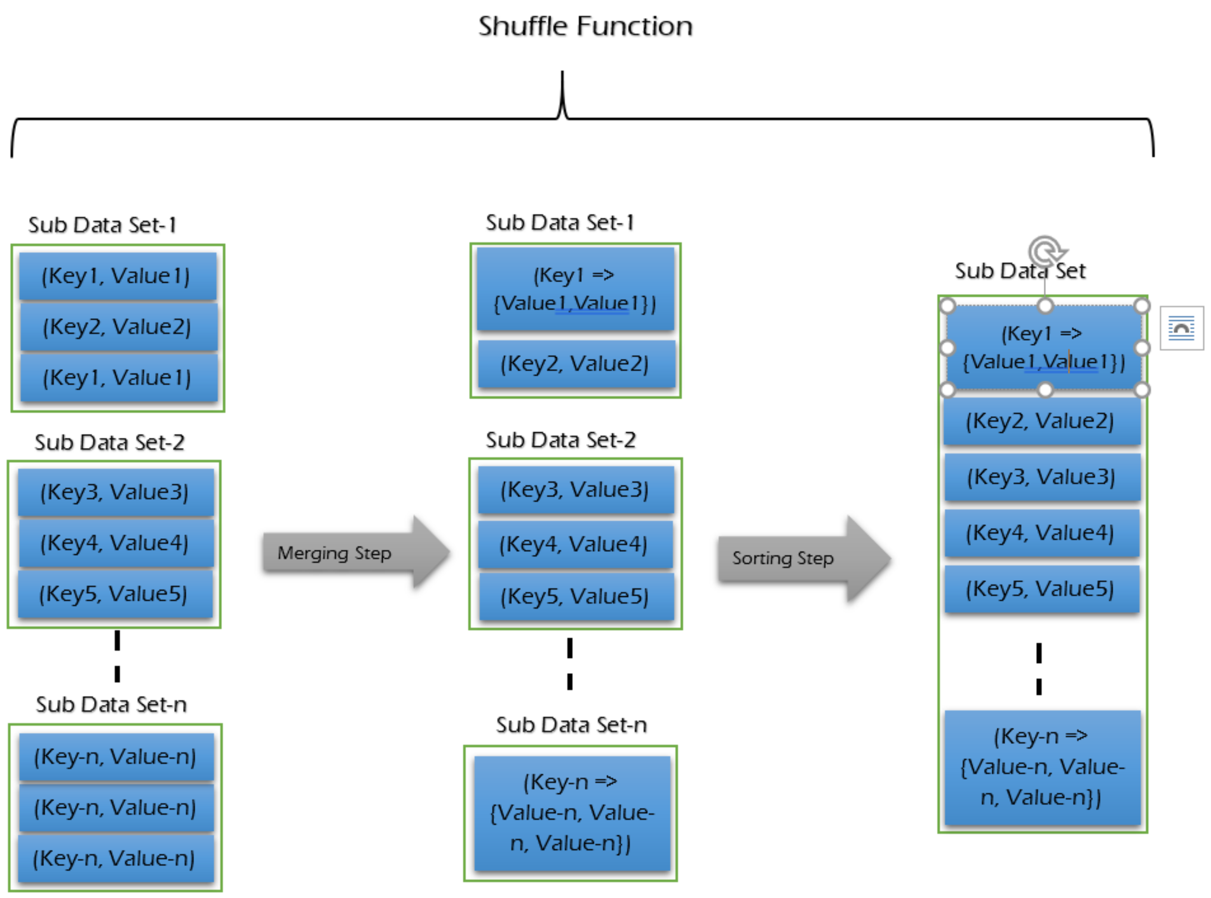
OUTPUT: set of key and value pairs as <Key, Value>.



2) Sort & Shuffle Stage
Shuffle and Sort is the second phase in MapReduce Algorithm.
This Shuffle and Sort is also called as “Combine Function”
The output from the mapper class is taken as input and sort and shuffle them.

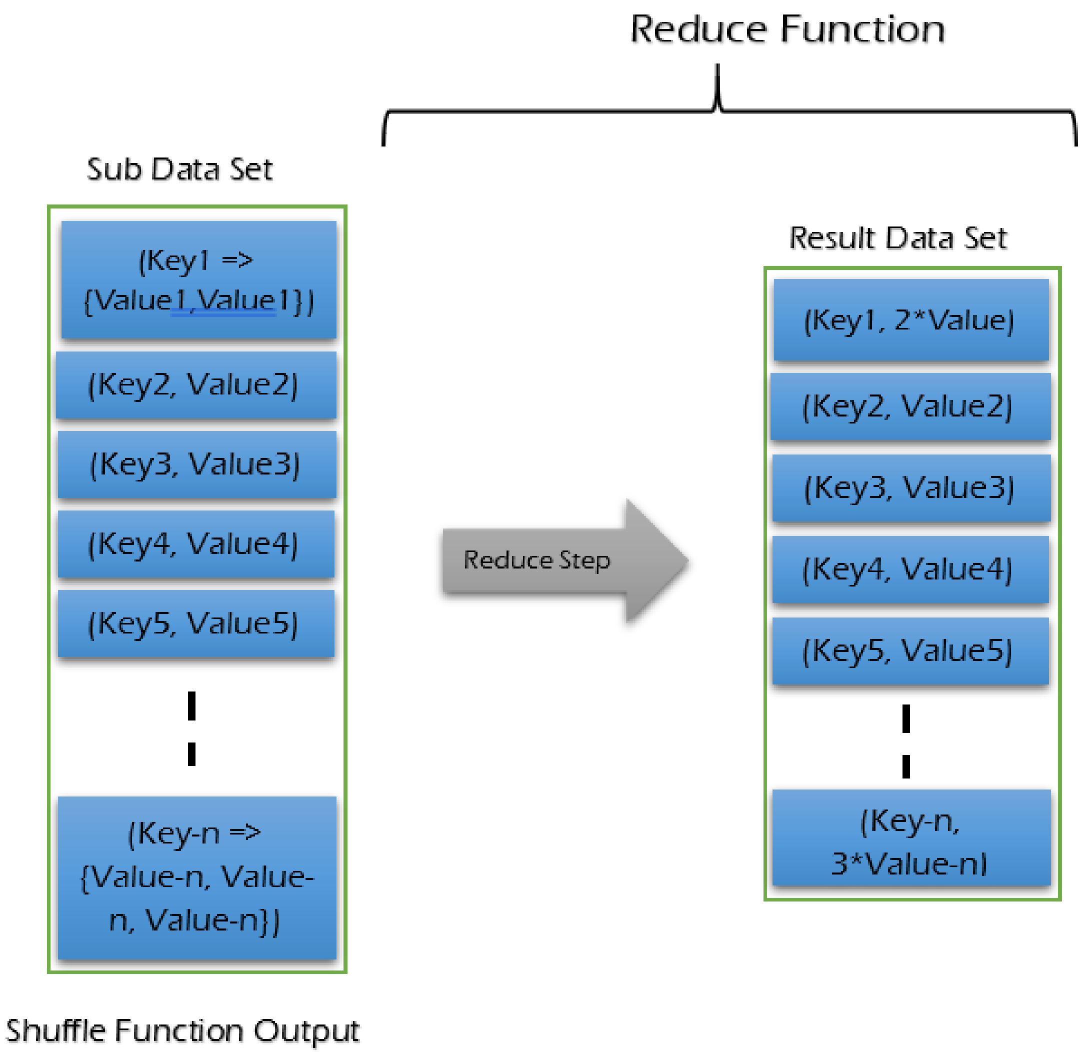
Merging - Find and merge all key and value pairs which all have same key.
Sorting - Sort all of the key and value pairs by keys.

OUTPUT: group of key and value pairs as <Key, List>



3) Reduce Stage
Reducer is the final stage in MapReduce Algorithm.
Takes list of sorted pairs of <Key, List>
After finishing the reducer part the cluster collects the data and send the data back to hadoop server.

OUTPUT: Result as <Key, Value>



- **Question 3:**

1. Hive Usecase

- Create a new table in hive.
- Use built-in hive functions.
- Create 10 queries

2. Solr Usecase

- Create a new Solr Collection.

- Make 10 queries.
- Record execution time for the queries.

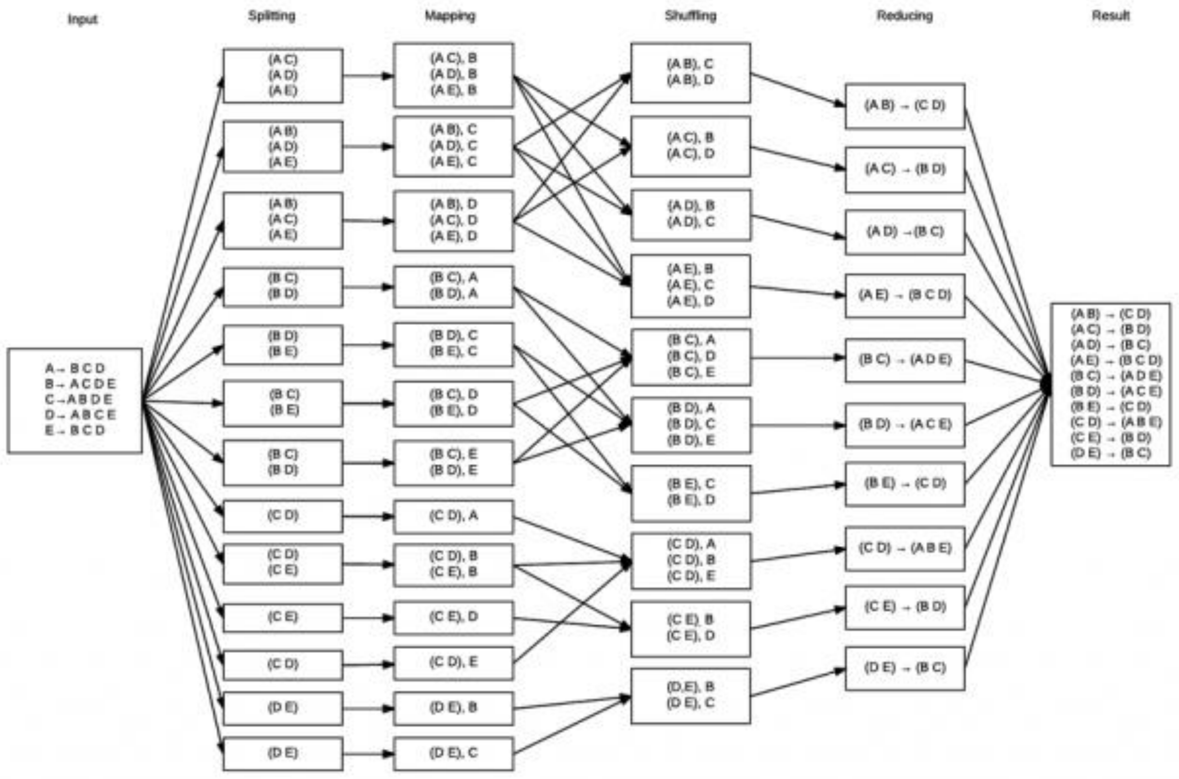
Datasets:

- For Question-1: Facebook Dataset-1, Facebook Dataset - 2
- For Question-2: Youtube Dataset-before pre-processing, Youtube Dataset-after pre-processing
- For Question-3: Zomato Dataset

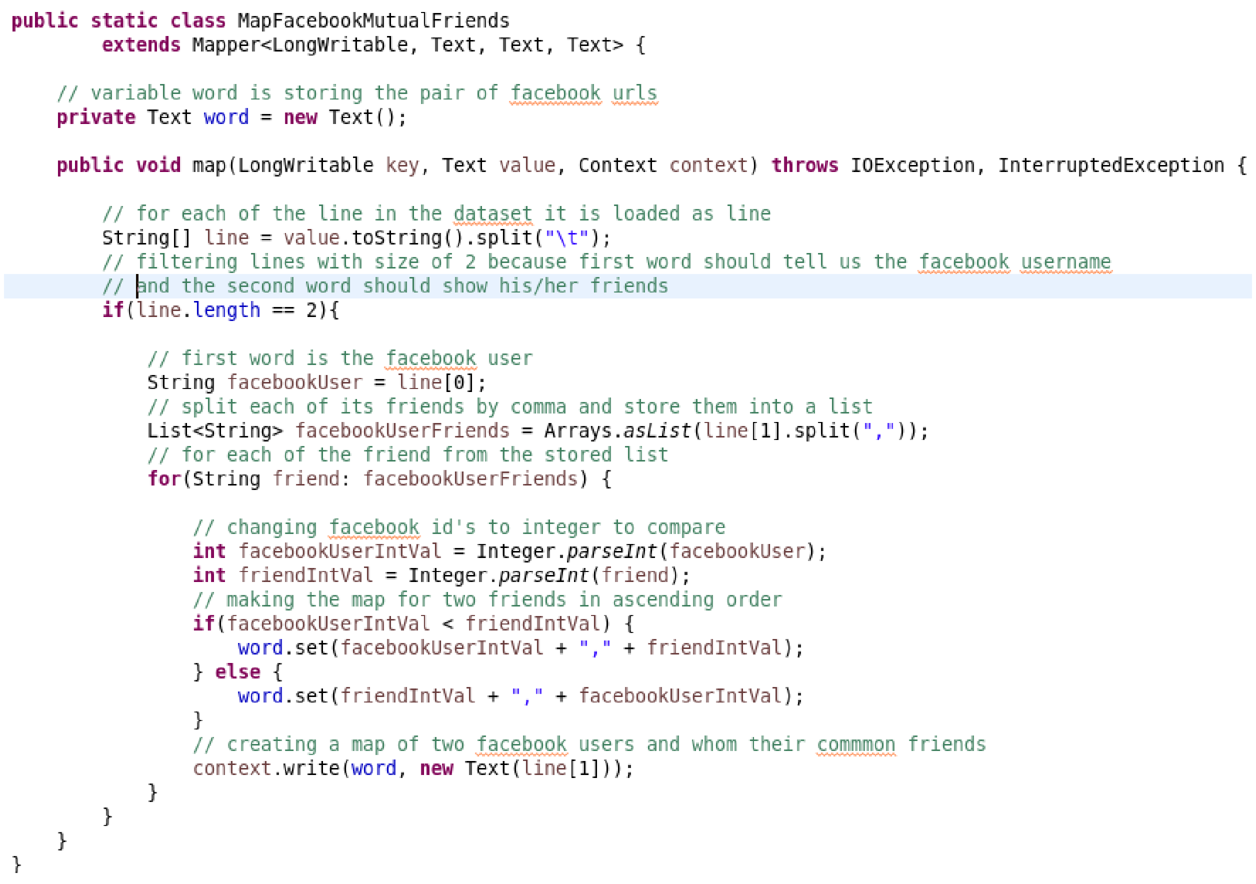
Code Screenshots:

- **Question-1:**

Map Reduce Diagram:

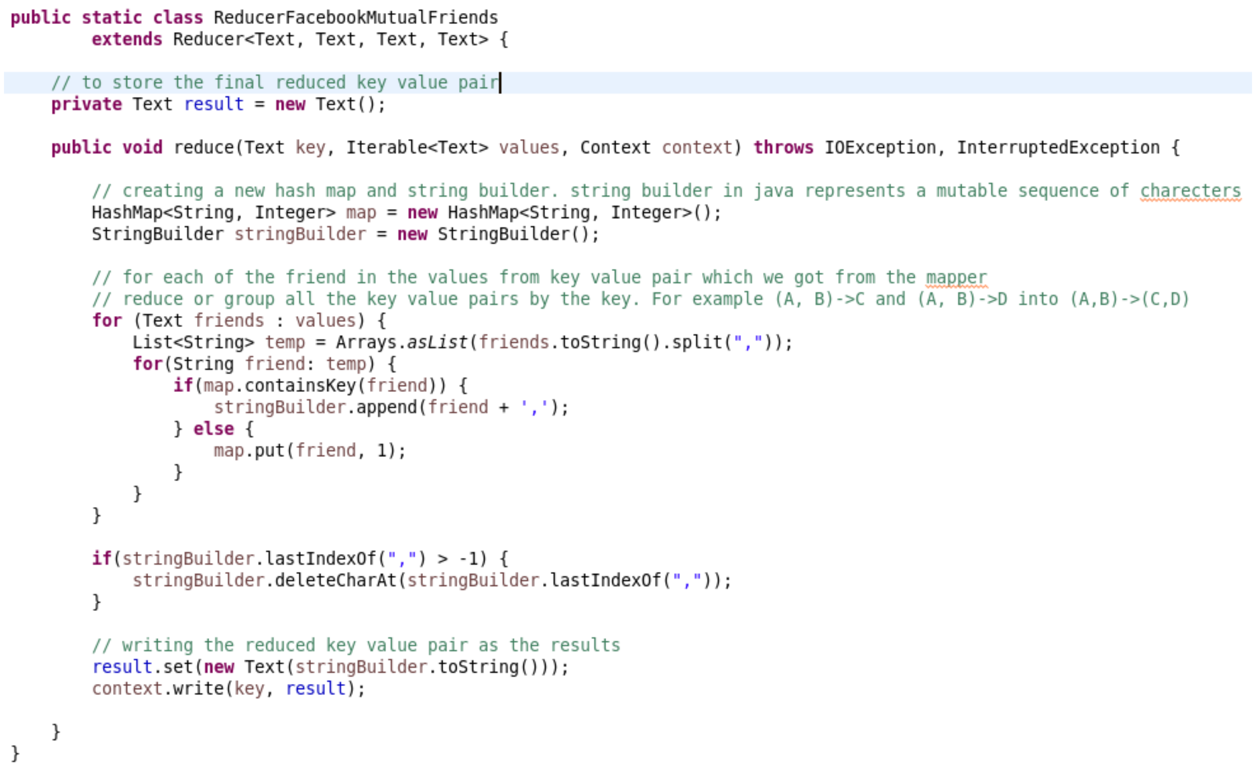


1. Mapper (To see the code -> click here)



```java
public static class MapFacebookMutualFriends
        extends Mapper<LongWritable, Text, Text, Text> {

    // variable word is storing the pair of facebook urls
    private Text word = new Text();

    public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

        // for each of the line in the dataset it is loaded as line
        String[] line = value.toString().split("\t");
        // filtering lines with size of 2 because first word should tell us the facebook username
        // and the second word should show his/her friends
        if(line.length == 2){

            // first word is the facebook user
            String facebookUser = line[0];
            // split each of its friends by comma and store them into a list
            List<String> facebookUserFriends = Arrays.asList(line[1].split(","));
            // for each of the friend from the stored list
            for(String friend: facebookUserFriends) {

                // changing facebook id's to integer to compare
                int facebookUserIntVal = Integer.parseInt(facebookUser);
                int friendIntVal = Integer.parseInt(friend);
                // making the map for two friends in ascending order
                if(facebookUserIntVal < friendIntVal) {
                    word.set(facebookUserIntVal + "," + friendIntVal);
                } else {
                    word.set(friendIntVal + "," + facebookUserIntVal);
                }
                // creating a map of two facebook users and whom their commmon friends
                context.write(word, new Text(line[1]));
            }
        }
    }
}
```

2. Reducer (To see the code -> click here)



```java
public static class ReducerFacebookMutualFriends
        extends Reducer<Text, Text, Text, Text> {

    // to store the final reduced key value pair
    private Text result = new Text();

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

        // creating a new hash map and string builder. string builder in java represents a mutable sequence of charecters
        HashMap<String, Integer> map = new HashMap<String, Integer>();
        StringBuilder stringBuilder = new StringBuilder();

        // for each of the friend in the values from key value pair which we got from the mapper
        // reduce or group all the key value pairs by the key. For example (A, B)->C and (A, B)->D into (A,B)->(C,D)
        for (Text friends : values) {
            List<String> temp = Arrays.asList(friends.toString().split(","));
            for(String friend: temp) {
                if(map.containsKey(friend)) {
                    stringBuilder.append(friend + ',');
                } else {
                    map.put(friend, 1);
                }
            }
        }

        if(stringBuilder.lastIndexOf(",") > -1) {
            stringBuilder.deleteCharAt(stringBuilder.lastIndexOf(","));
        }

        // writing the reduced key value pair as the results
        result.set(new Text(stringBuilder.toString()));
        context.write(key, result);

    }
}
```

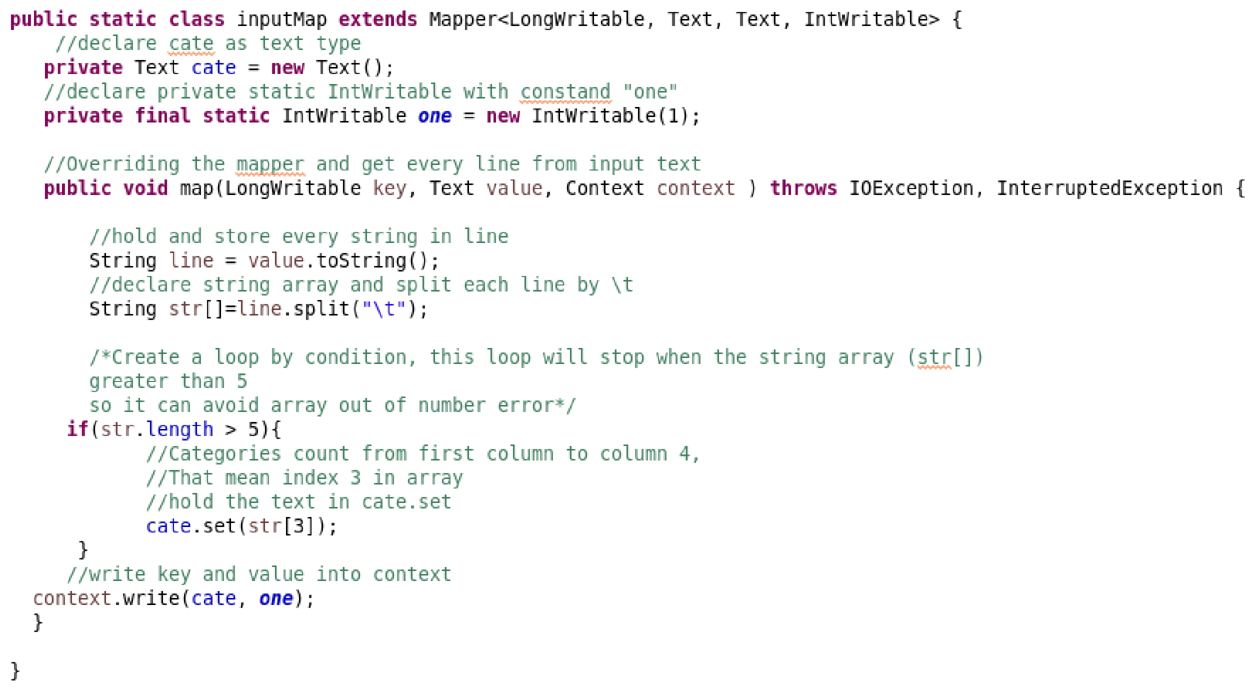
3. Main Class (To see the code -> click here)

```
public static void main(String[] args) throws Exception {

    // number of arguments should be exactly 2
    if(args.length != 2){
        System.err.println("Ivalid Arguments!!");
    }

    // configuration setup
    Configuration conf = new Configuration();

    // set job instance
    Job job = Job.getInstance(conf, "MutualFriends");
    // class name
    job.setJarByClass(FacebookMutualFriends.class);

    // what is this?
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);

    // set mapper class and reducer class
    job.setMapperClass(MapFacebookMutualFriends.class);
    job.setReducerClass(ReducerFacebookMutualFriends.class);

    // set input format and output format
    job.setOutputFormatClass(TextOutputFormat.class);
    job.setInputFormatClass(TextInputFormat.class);

    // first argument will be the input path and second will be output path
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
}
```

- ## Question-2:

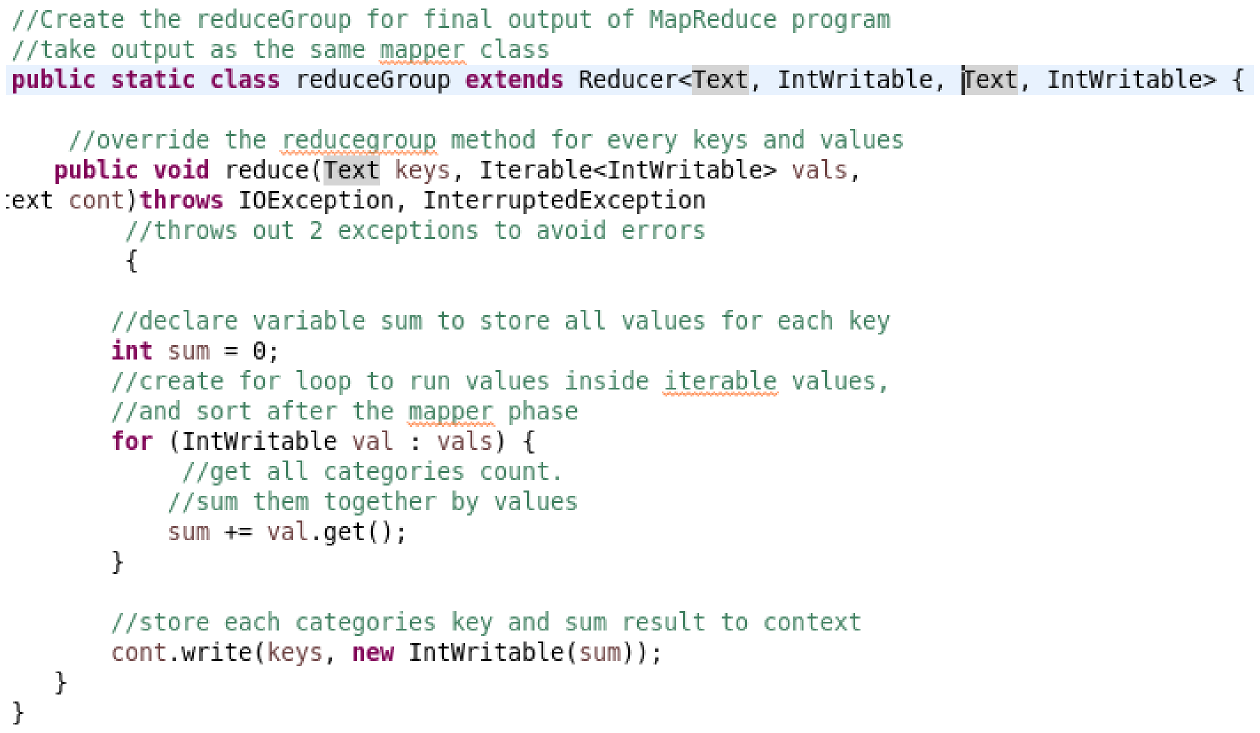
Part-1

1. Mapper (To see the code -> click here)



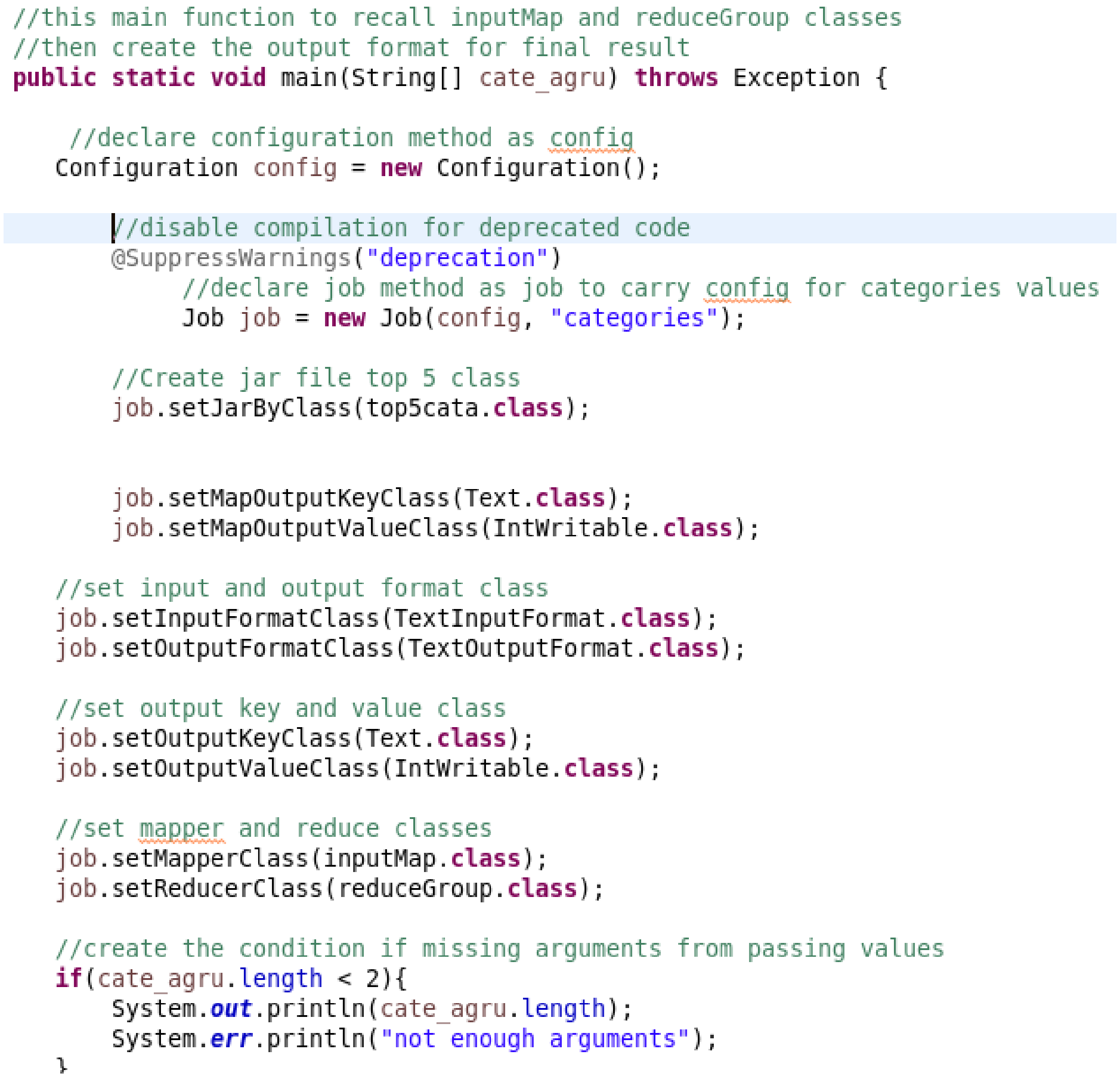
```java
public static class inputMap extends Mapper<LongWritable, Text, Text, IntWritable> {
    //declare cate as text type
   private Text cate = new Text();
   //declare private static IntWritable with constand "one"
   private final static IntWritable one = new IntWritable(1);

   //Overriding the mapper and get every line from input text
   public void map(LongWritable key, Text value, Context context ) throws IOException, InterruptedException {

       //hold and store every string in line
       String line = value.toString();
       //declare string array and split each line by \t
       String str[]=line.split("\t");

       /*Create a loop by condition, this loop will stop when the string array (str[])
       greater than 5
       so it can avoid array out of number error*/
     if(str.length > 5){
            //Categories count from first column to column 4,
            //That mean index 3 in array
            //hold the text in cate.set
            cate.set(str[3]);
      }
     //write key and value into context
  context.write(cate, one);
  }

}
```

2. Reducer (To see the code -> click here)



```
//Create the reduceGroup for final output of MapReduce program
//take output as the same mapper class
public static class reduceGroup extends Reducer<Text, IntWritable, Text, IntWritable> {

    //override the reducegroup method for every keys and values
   public void reduce(Text keys, Iterable<IntWritable> vals,
ext cont)throws IOException, InterruptedException
        //throws out 2 exceptions to avoid errors
        {

       //declare variable sum to store all values for each key
       int sum = 0;
       //create for loop to run values inside iterable values,
       //and sort after the mapper phase
       for (IntWritable val : vals) {
            //get all categories count.
           //sum them together by values
           sum += val.get();
       }

       //store each categories key and sum result to context
       cont.write(keys, new IntWritable(sum));
   }
}
```

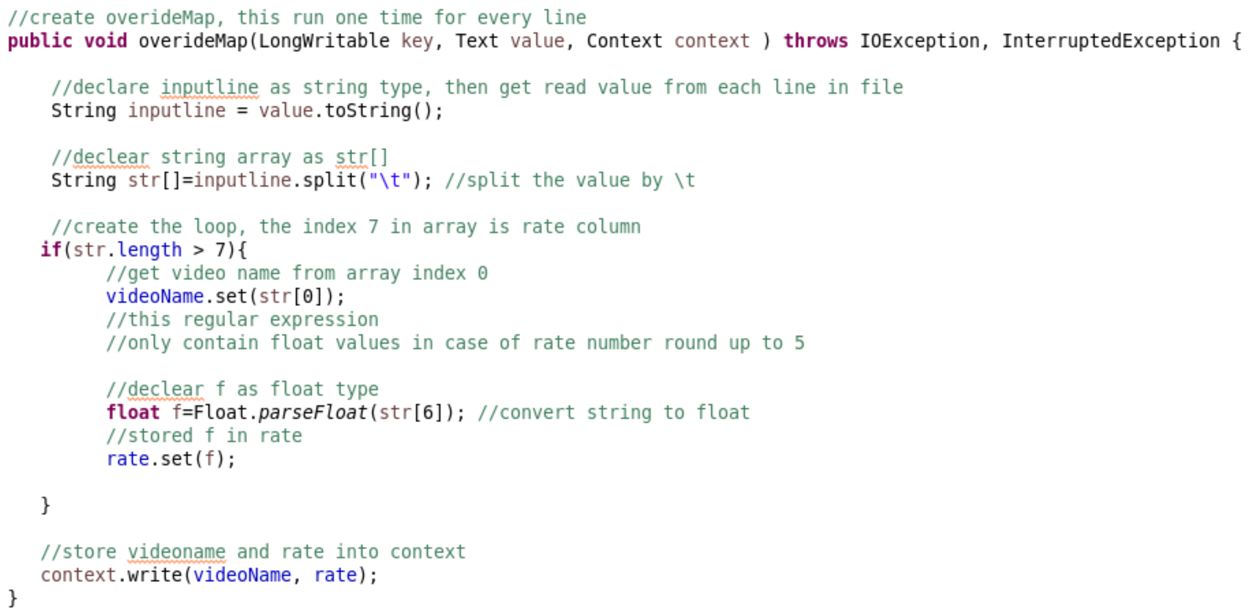
3. Main Class (To see the code -> click here)



```java
//this main function to recall inputMap and reduceGroup classes
//then create the output format for final result
public static void main(String[] cate_agru) throws Exception {

    //declare configuration method as config
   Configuration config = new Configuration();

       //disable compilation for deprecated code
       @SuppressWarnings("deprecation")
            //declare job method as job to carry config for categories values
            Job job = new Job(config, "categories");

       //Create jar file top 5 class
       job.setJarByClass(top5cata.class);


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

   //set input and output format class
   job.setInputFormatClass(TextInputFormat.class);
   job.setOutputFormatClass(TextOutputFormat.class);

   //set output key and value class
   job.setOutputKeyClass(Text.class);
   job.setOutputValueClass(IntWritable.class);

   //set mapper and reduce classes
   job.setMapperClass(inputMap.class);
   job.setReducerClass(reduceGroup.class);

   //create the condition if missing arguments from passing values
   if(cate_agru.length < 2){
       System.out.println(cate_agru.length);
       System.err.println("not enough arguments");
   }
```

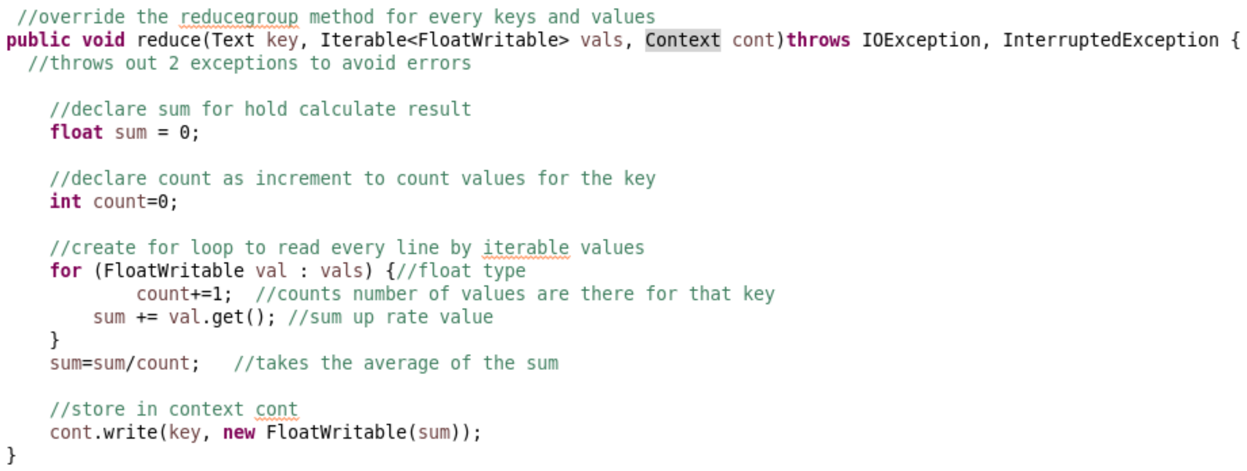
Part-2

1. Mapper (To see the code -> click here)



```
//create overideMap, this run one time for every line
public void overideMap(LongWritable key, Text value, Context context ) throws IOException, InterruptedException {

    //declare inputline as string type, then get read value from each line in file
    String inputline = value.toString();

    //declear string array as str[]
    String str[]=inputline.split("\t"); //split the value by \t

    //create the loop, the index 7 in array is rate column
   if(str.length > 7){
        //get video name from array index 0
        videoName.set(str[0]);
        //this regular expression
        //only contain float values in case of rate number round up to 5

        //declear f as float type
        float f=Float.parseFloat(str[6]); //convert string to float
        //stored f in rate
        rate.set(f);

   }

   //store videoname and rate into context
   context.write(videoName, rate);
}
```

2. Reducer (To see the code -> click here)



```
 //override the reducegroup method for every keys and values
public void reduce(Text key, Iterable<FloatWritable> vals, Context cont)throws IOException, InterruptedException {
  //throws out 2 exceptions to avoid errors

    //declare sum for hold calculate result
    float sum = 0;

    //declare count as increment to count values for the key
    int count=0;

    //create for loop to read every line by iterable values
    for (FloatWritable val : vals) {//float type
            count+=1;  //counts number of values are there for that key
        sum += val.get(); //sum up rate value
    }
    sum=sum/count;   //takes the average of the sum

    //store in context cont
    cont.write(key, new FloatWritable(sum));
}
```

3. Main Class (To see the code -> click here)

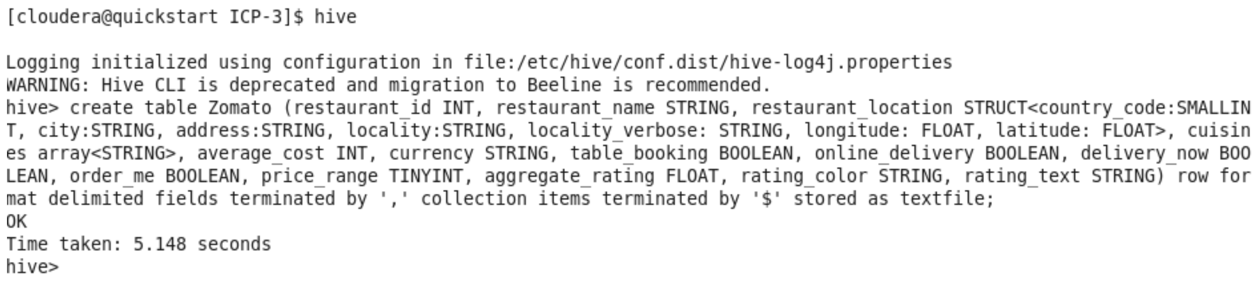
```
//this main function to recall inputMap and reduceGroup classes
//then create the output format for final result
public static void main(String[] argument) throws Exception {
    //declare configuration method as config
   Configuration config = new Configuration();

 //disable compilation for deprecated code
      @SuppressWarnings("deprecation")
     //declare job method as job to carry config for categories values
           Job job = new Job(config, "videorating");

       //Create jar file for videoRating class
       job.setJarByClass(videoRating.class);


       //set map output key and value
       job.setMapOutputKeyClass(Text.class);
       job.setMapOutputValueClass(FloatWritable.class);

   //set output key and value
   job.setOutputKeyClass(Text.class);
   job.setOutputValueClass(FloatWritable.class);

   //recall inputMap and reduceGroup and set them as
   //mapper and reducer class
   job.setMapperClass(inputMap.class);
   job.setReducerClass(reduceGroup.class);


   //set input and output format
   job.setInputFormatClass(TextInputFormat.class);
   job.setOutputFormatClass(TextOutputFormat.class);
```

- ## Question-3:

Part-1

Start Hive and Create a new table called Zomato.

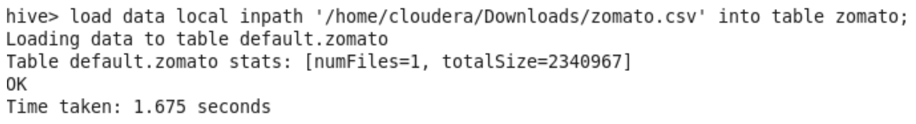
Command: create table Zomato (restaurant_id INT, restaurant_name STRING, restaurant_location STRUCT<country_code:SMALLINT, city:STRING, address:STRING, locality:STRING, locality_verbose: STRING, longitude: FLOAT, latitude: FLOAT>, cuisines array, average_cost INT, currency STRING, table_booking BOOLEAN, online_delivery BOOLEAN, delivery_now BOOLEAN, order_me BOOLEAN, price_range TINYINT, aggregate_rating FLOAT, rating_color STRING, rating_text STRING) row format delimited fields terminated by ',' collection items terminated by '$' stored as textfile;



```
[cloudera@quickstart ICP-3]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> create table Zomato (restaurant_id INT, restaurant_name STRING, restaurant_location STRUCT<country_code:SMALLIN
T, city:STRING, address:STRING, locality:STRING, locality_verbose: STRING, longitude: FLOAT, latitude: FLOAT>, cuisin
es array<STRING>, average_cost INT, currency STRING, table_booking BOOLEAN, online_delivery BOOLEAN, delivery_now BOO
LEAN, order_me BOOLEAN, price_range TINYINT, aggregate_rating FLOAT, rating_color STRING, rating_text STRING) row for
mat delimited fields terminated by ',' collection items terminated by '$' stored as textfile;
OK
Time taken: 5.148 seconds
hive>
```

Load the csv file into the created table.

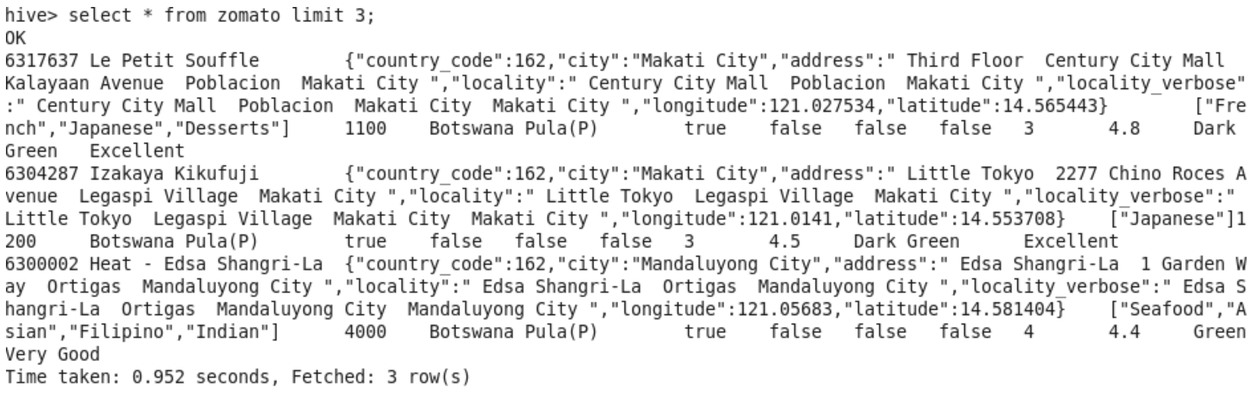
Command: load data local inpath '/home/cloudera/Downloads/zomato.csv' into table zomato;



```
hive> load data local inpath '/home/cloudera/Downloads/zomato.csv' into table zomato;
Loading data to table default.zomato
Table default.zomato stats: [numFiles=1, totalSize=2340967]
OK
Time taken: 1.675 seconds
```

To verify the data insertion.

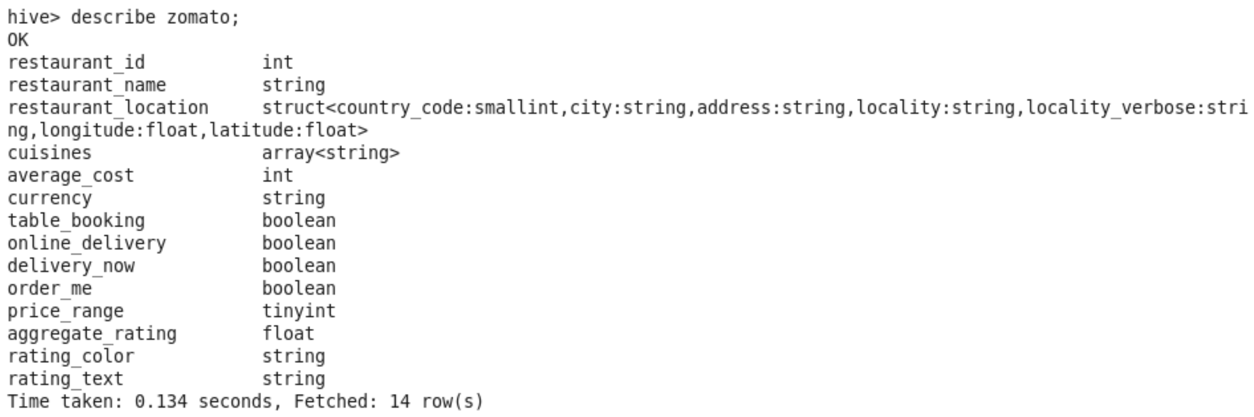
Command: select * from zomato limit 3;



```
hive> select * from zomato limit 3;
OK
6317637 Le Petit Souffle        {"country_code":162,"city":"Makati City","address":" Third Floor  Century City Mall
Kalayaan Avenue  Poblacion  Makati City ","locality":" Century City Mall  Poblacion  Makati City ","locality_verbose"
:" Century City Mall  Poblacion  Makati City  Makati City ","longitude":121.027534,"latitude":14.565443}        ["Fre
nch","Japanese","Desserts"]     1100    Botswana Pula(P)        true    false   false   false   3       4.8     Dark
Green   Excellent
6304287 Izakaya Kikufuji        {"country_code":162,"city":"Makati City","address":" Little Tokyo  2277 Chino Roces A
venue  Legaspi Village  Makati City ","locality":" Little Tokyo  Legaspi Village  Makati City ","locality_verbose":"
Little Tokyo  Legaspi Village  Makati City  Makati City ","longitude":121.0141,"latitude":14.553708}    ["Japanese"]1
200     Botswana Pula(P)        true    false   false   false   3       4.5     Dark Green      Excellent
6300002 Heat - Edsa Shangri-La  {"country_code":162,"city":"Mandaluyong City","address":" Edsa Shangri-La  1 Garden W
ay  Ortigas  Mandaluyong City ","locality":" Edsa Shangri-La  Ortigas  Mandaluyong City ","locality_verbose":" Edsa S
hangri-La  Ortigas  Mandaluyong City  Mandaluyong City ","longitude":121.05683,"latitude":14.581404}    ["Seafood","A
sian","Filipino","Indian"]      4000    Botswana Pula(P)        true    false   false   false   4       4.4     Green
Very Good
Time taken: 0.952 seconds, Fetched: 3 row(s)
```

To describe the created table.

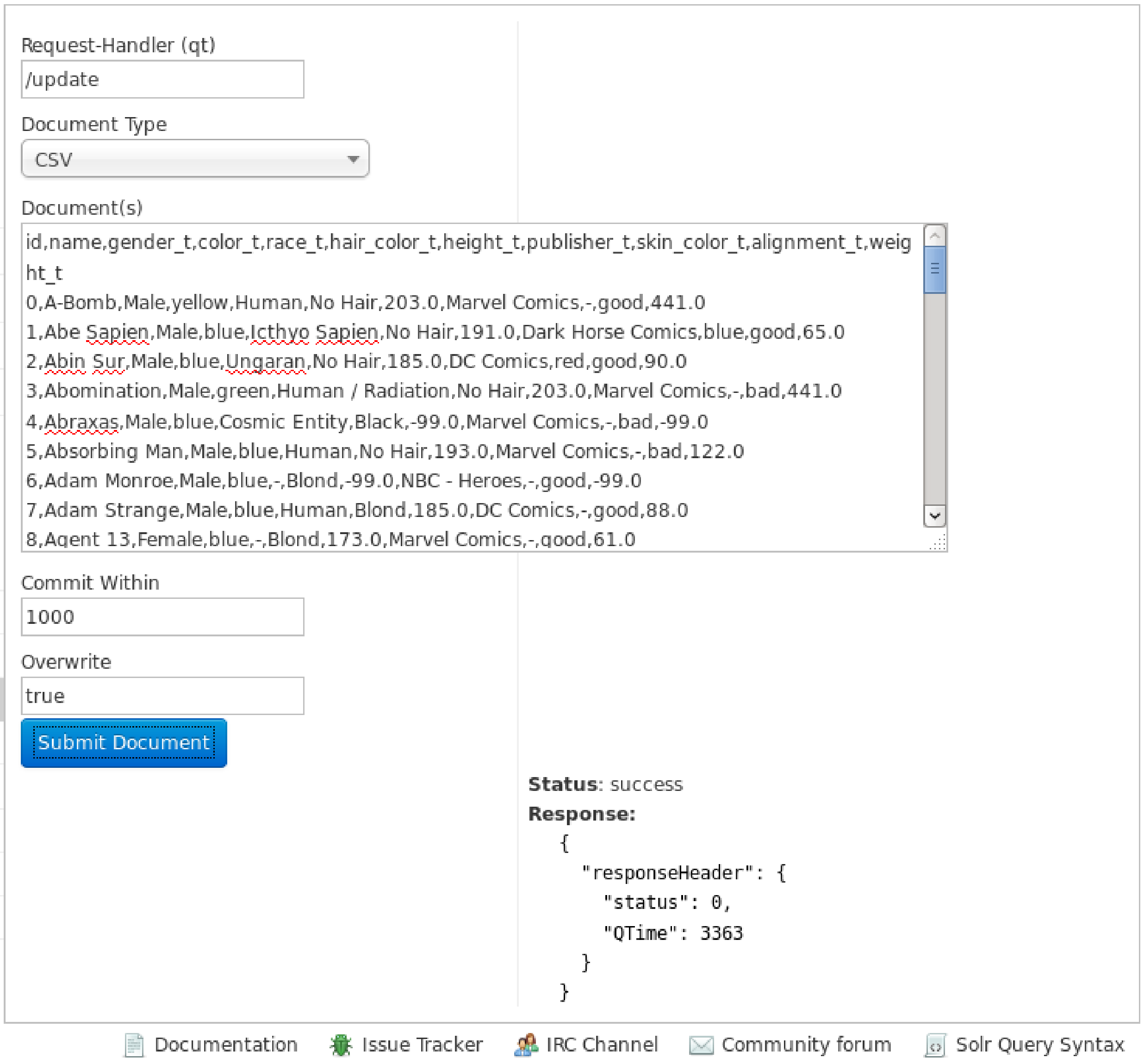
Command: select restaurant_location.city, count(*) from zomato group by restaurant_location.city;



```
hive> describe zomato;
OK
restaurant_id           int
restaurant_name         string
restaurant_location     struct<country_code:smallint,city:string,address:string,locality:string,locality_verbose:stri
ng,longitude:float,latitude:float>
cuisines                array<string>
average_cost            int
currency                string
table_booking           boolean
online_delivery         boolean
delivery_now            boolean
order_me                boolean
price_range             tinyint
aggregate_rating        float
rating_color            string
rating_text             string
Time taken: 0.134 seconds, Fetched: 14 row(s)
```

Part-2

Data Insertion in solr.



Request-Handler (qt)
/update

Document Type
CSV

Document(s)

```
id,name,gender_t,color_t,race_t,hair_color_t,height_t,publisher_t,skin_color_t,alignment_t,weig
ht_t
0,A-Bomb,Male,yellow,Human,No Hair,203.0,Marvel Comics,-,good,441.0
1,Abe Sapien,Male,blue,Icthyo Sapien,No Hair,191.0,Dark Horse Comics,blue,good,65.0
2,Abin Sur,Male,blue,Ungaran,No Hair,185.0,DC Comics,red,good,90.0
3,Abomination,Male,green,Human / Radiation,No Hair,203.0,Marvel Comics,-,bad,441.0
4,Abraxas,Male,blue,Cosmic Entity,Black,-99.0,Marvel Comics,-,bad,-99.0
5,Absorbing Man,Male,blue,Human,No Hair,193.0,Marvel Comics,-,bad,122.0
6,Adam Monroe,Male,blue,-,Blond,-99.0,NBC - Heroes,-,good,-99.0
7,Adam Strange,Male,blue,Human,Blond,185.0,DC Comics,-,good,88.0
8,Agent 13,Female,blue,-,Blond,173.0,Marvel Comics,-,good,61.0
```

Commit Within
1000

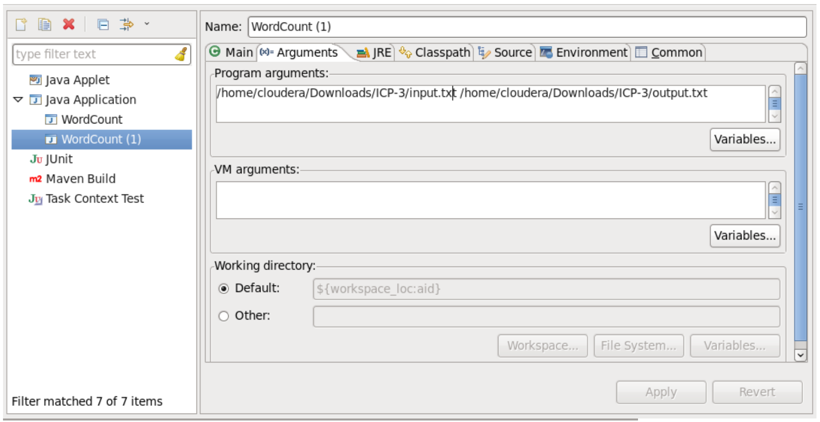
Overwrite
true

Status: success
Response:

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 3363
  }
}
```

Results/Output Screenshots:

- **Question-1:**



Name: WordCount (1)

Program arguments:
```
/home/cloudera/Downloads/ICP-3/input.txt /home/cloudera/Downloads/ICP-3/output.txt
```

VM arguments:

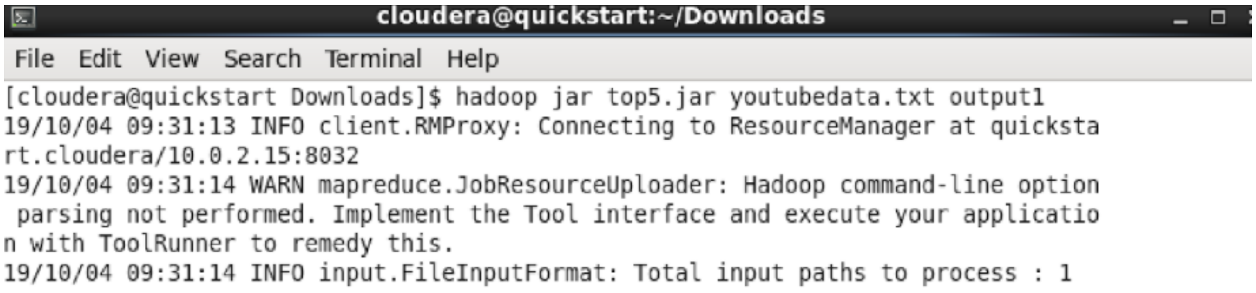
Working directory:
Default: [✓] ${workspace_loc:aid}
Other:

```
0,1     5,20
0,10    12,16,30
0,11
0,12    3,10,16,29,30,38,41,55,83,85,89
0,13    27,37
0,14    4,19
0,15    4,27,80
0,16    10,12,18,30,38,89,53,83
0,17    19,26,28,53
0,18    4,16,30,89
0,19    14,17,50
0,2
0,20    1,5
0,21    6,52,63,91
0,22    29
0,23
0,24    28,38,53,83,85
0,25
0,26    17
0,27    4,15,13
0,28    17,24,38,53,83,85,89
0,29    12,22,38
0,3     12,41,55
0,30    10,16,18,12,83
0,31
0,32    90,92
0,33
0,34
0,35
0,36    39,43
0,37    13
0,38    8,12,16,24,28,29,46,89
0,39    36
0,4     8,14,15,18,27,72,74,77,80
0,40    43
0,41    3,12
0.42
```

- **Question-2:**

Part-1

Command: Hadoop jar top5.jar youtubedata.txt output1

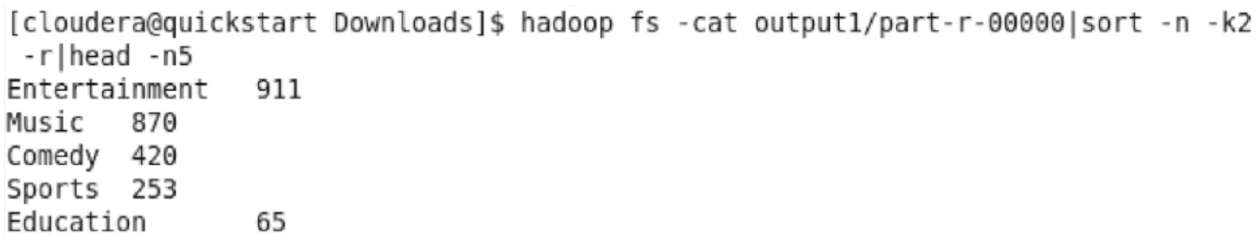


cloudera@quickstart:~/Downloads

File Edit View Search Terminal Help

```
[cloudera@quickstart Downloads]$ hadoop jar top5.jar youtubedata.txt output1
19/10/04 09:31:13 INFO client.RMProxy: Connecting to ResourceManager at quicksta
rt.cloudera/10.0.2.15:8032
19/10/04 09:31:14 WARN mapreduce.JobResourceUploader: Hadoop command-line option
 parsing not performed. Implement the Tool interface and execute your applicatio
n with ToolRunner to remedy this.
19/10/04 09:31:14 INFO input.FileInputFormat: Total input paths to process : 1
```

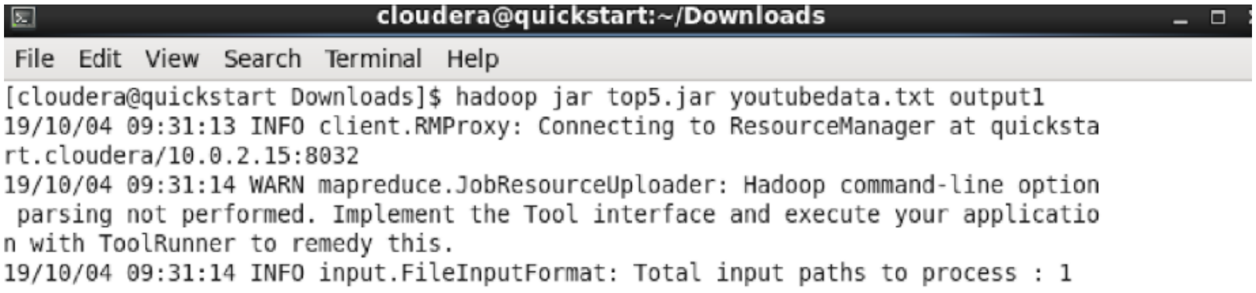
Command: Hadoop fs -cat /output1/part-r-00000|sort -n -k2 -r|head -n



```
[cloudera@quickstart Downloads]$ hadoop fs -cat output1/part-r-00000|sort -n -k2
 -r|head -n5
Entertainment   911
Music   870
Comedy  420
Sports  253
Education       65
```

Part-2

Command: Hadoop jar top10.jar youtubedata.txt output2

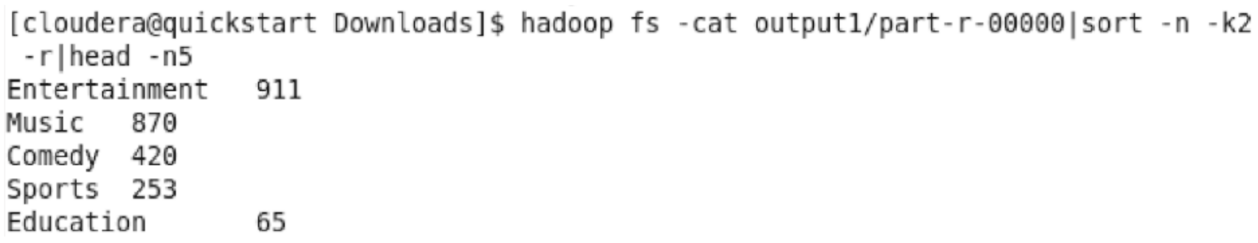


cloudera@quickstart:~/Downloads

File Edit View Search Terminal Help

```
[cloudera@quickstart Downloads]$ hadoop jar top5.jar youtubedata.txt output1
19/10/04 09:31:13 INFO client.RMProxy: Connecting to ResourceManager at quicksta
rt.cloudera/10.0.2.15:8032
19/10/04 09:31:14 WARN mapreduce.JobResourceUploader: Hadoop command-line option
 parsing not performed. Implement the Tool interface and execute your applicatio
n with ToolRunner to remedy this.
19/10/04 09:31:14 INFO input.FileInputFormat: Total input paths to process : 1
```

Command: Hadoop fs -cat output2/part-r-00000 |sort -n -k2 -r|head -n10



```
[cloudera@quickstart Downloads]$ hadoop fs -cat output1/part-r-00000|sort -n -k2
 -r|head -n5
Entertainment   911
Music   870
Comedy  420
Sports  253
Education       65
```

Explaination

****Hadoop fs -cat /output1/part-r-00000 ****

- fs: specify an operation related to Hadoop
- -cat: use to view the content
- /output1/part-r-00000 : directory to the output file stored

****sort -n -k2 -r|head -n ****

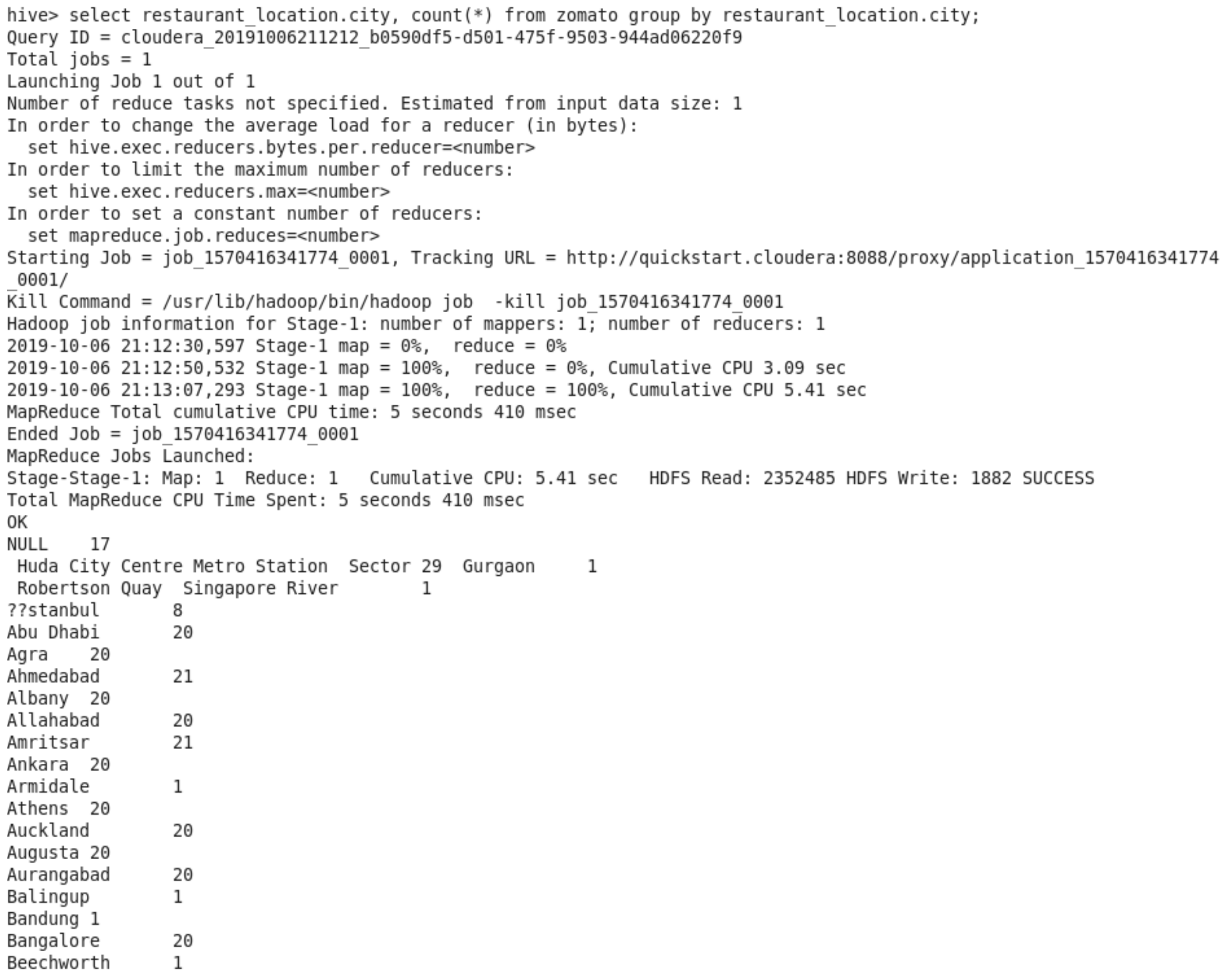
- sort: sort the data
- -n: sort by numerically
- -k2: second column
- -r : recursive operation
- head -n5 : take the first 5 values after sorting.

- ## Question-3:

Part-1

Query-1: Find number of restaurants in each city.

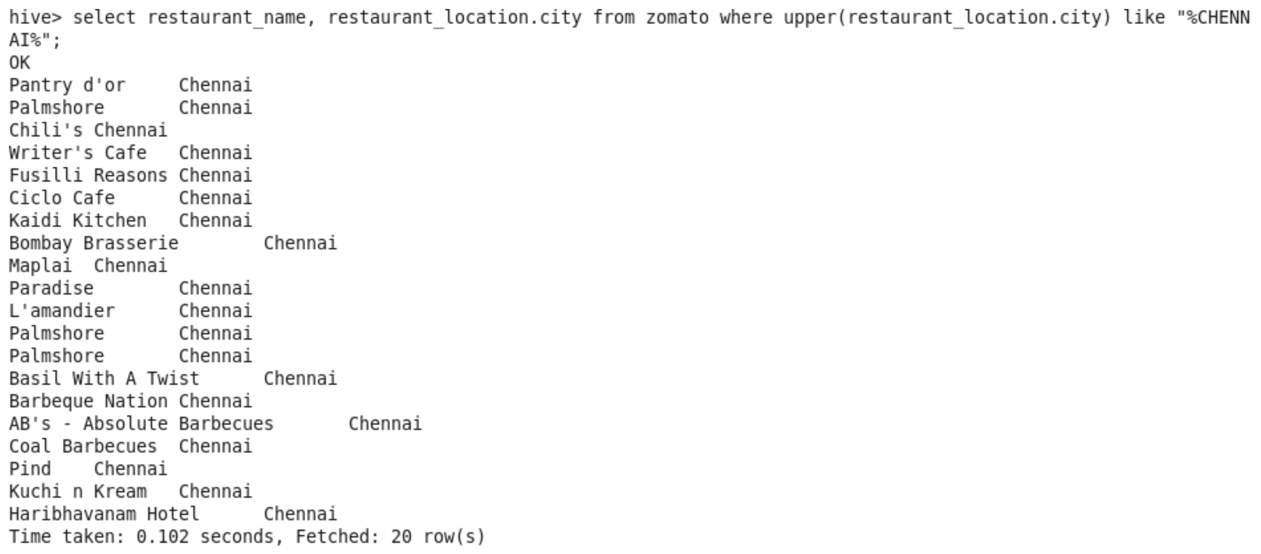
Command: select restaurant_location.city, count(*) from zomato group by restaurant_location.city;



```
hive> select restaurant_location.city, count(*) from zomato group by restaurant_location.city;
Query ID = cloudera_20191006211212_b0590df5-d501-475f-9503-944ad06220f9
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>
Starting Job = job_1570416341774_0001, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1570416341774
_0001/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1570416341774_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-10-06 21:12:30,597 Stage-1 map = 0%,  reduce = 0%
2019-10-06 21:12:50,532 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.09 sec
2019-10-06 21:13:07,293 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.41 sec
MapReduce Total cumulative CPU time: 5 seconds 410 msec
Ended Job = job_1570416341774_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.41 sec   HDFS Read: 2352485 HDFS Write: 1882 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 410 msec
OK
NULL    17
 Huda City Centre Metro Station  Sector 29  Gurgaon     1
 Robertson Quay  Singapore River        1
??stanbul       8
Abu Dhabi       20
Agra    20
Ahmedabad       21
Albany  20
Allahabad       20
Amritsar        21
Ankara  20
Armidale        1
Athens  20
Auckland        20
Augusta 20
Aurangabad      20
Balingup        1
Bandung 1
Bangalore       20
Beechworth      1
```

Query-2: Print restaurant names from chennai.

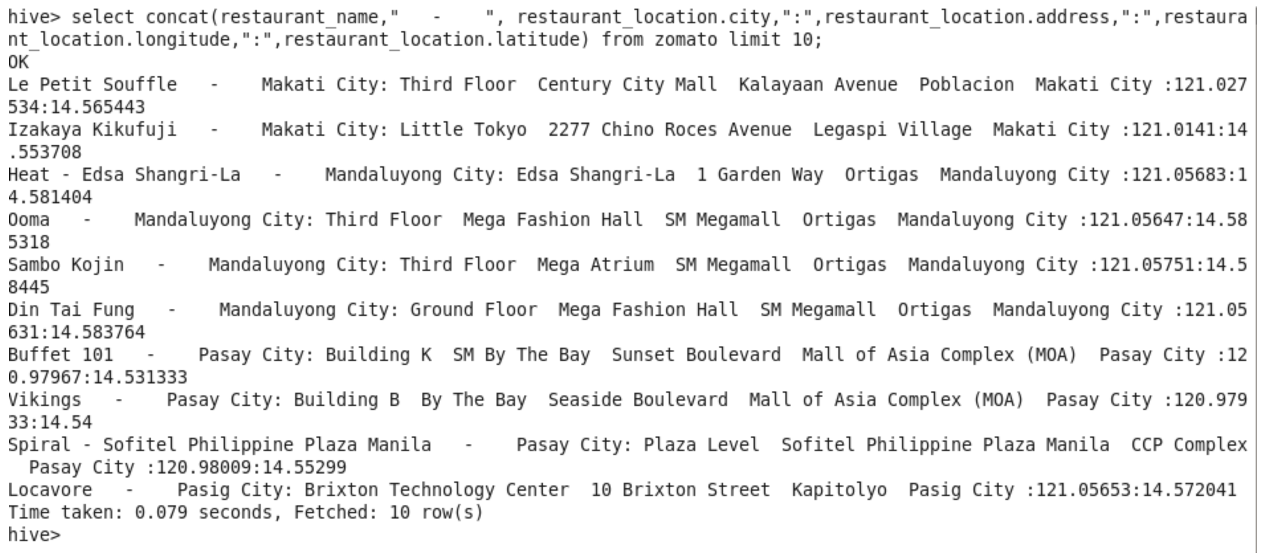
Command: select restaurant_name, restaurant_location.city from zomato where upper(restaurant_location.city) like "%CHENNAI%";



```
hive> select restaurant_name, restaurant_location.city from zomato where upper(restaurant_location.city) like "%CHENN
AI%";
OK
Pantry d'or     Chennai
Palmshore       Chennai
Chili's Chennai
Writer's Cafe   Chennai
Fusilli Reasons Chennai
Ciclo Cafe      Chennai
Kaidi Kitchen   Chennai
Bombay Brasserie        Chennai
Maplai  Chennai
Paradise        Chennai
L'amandier      Chennai
Palmshore       Chennai
Palmshore       Chennai
Basil With A Twist      Chennai
Barbeque Nation Chennai
AB's - Absolute Barbecues       Chennai
Coal Barbecues  Chennai
Pind    Chennai
Kuchi n Kream   Chennai
Haribhavanam Hotel      Chennai
Time taken: 0.102 seconds, Fetched: 20 row(s)
```

Query-3: Concat restaurant name with address, latitude, and longitude.

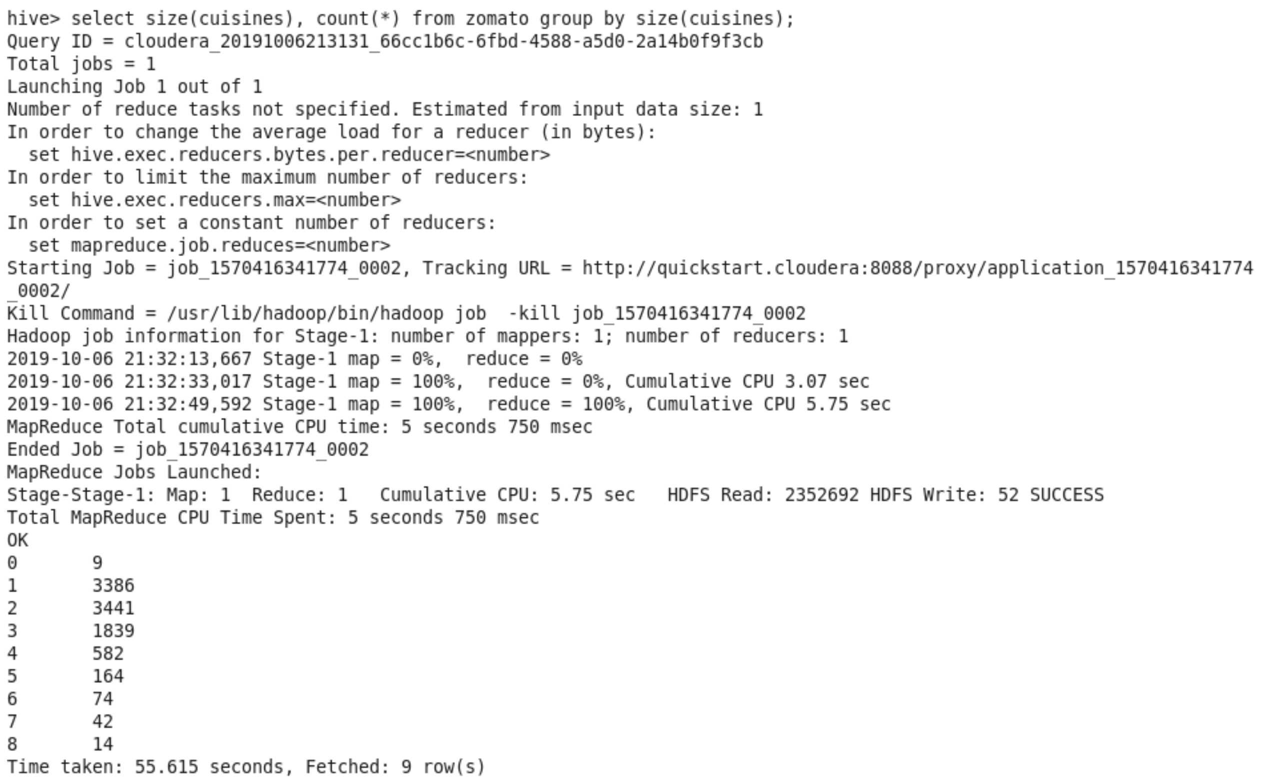
Command: select concat(restaurant_name," - ", restaurant_location.city,":",restaurant_location.address,":",restaurant_location.longitude,":",restaurant_ ocation.latitude) from zomato limit 10;



```
hive> select concat(restaurant_name,"   -    ", restaurant_location.city,":",restaurant_location.address,":",restaura
nt_location.longitude,":",restaurant_location.latitude) from zomato limit 10;
OK
Le Petit Souffle   -    Makati City: Third Floor  Century City Mall  Kalayaan Avenue  Poblacion  Makati City :121.027
534:14.565443
Izakaya Kikufuji   -    Makati City: Little Tokyo  2277 Chino Roces Avenue  Legaspi Village  Makati City :121.0141:14
.553708
Heat - Edsa Shangri-La   -    Mandaluyong City: Edsa Shangri-La  1 Garden Way  Ortigas  Mandaluyong City :121.05683:1
4.581404
Ooma   -    Mandaluyong City: Third Floor  Mega Fashion Hall  SM Megamall  Ortigas  Mandaluyong City :121.05647:14.58
5318
Sambo Kojin   -    Mandaluyong City: Third Floor  Mega Atrium  SM Megamall  Ortigas  Mandaluyong City :121.05751:14.5
8445
Din Tai Fung   -    Mandaluyong City: Ground Floor  Mega Fashion Hall  SM Megamall  Ortigas  Mandaluyong City :121.05
631:14.583764
Buffet 101   -    Pasay City: Building K  SM By The Bay  Sunset Boulevard  Mall of Asia Complex (MOA)  Pasay City :12
0.97967:14.531333
Vikings   -    Pasay City: Building B  By The Bay  Seaside Boulevard  Mall of Asia Complex (MOA)  Pasay City :120.979
33:14.54
Spiral - Sofitel Philippine Plaza Manila   -    Pasay City: Plaza Level  Sofitel Philippine Plaza Manila  CCP Complex
  Pasay City :120.98009:14.55299
Locavore   -    Pasig City: Brixton Technology Center  10 Brixton Street  Kapitolyo  Pasig City :121.05653:14.572041
Time taken: 0.079 seconds, Fetched: 10 row(s)
hive>
```

Query-4: To print size of cuisines with number of restaurants with that cuisines.

Command: select size(cuisines), count(*) from zomato group by size(cuisines);

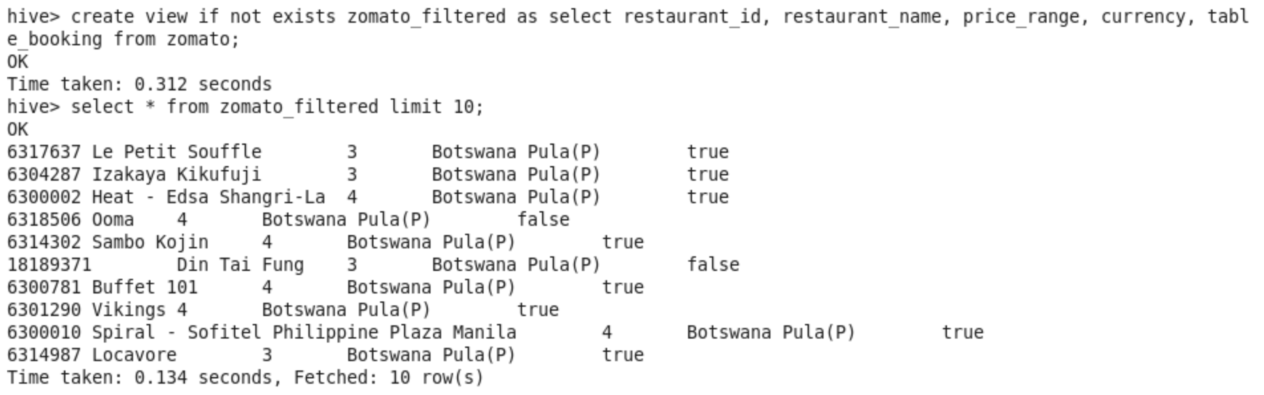


```
hive> select size(cuisines), count(*) from zomato group by size(cuisines);
Query ID = cloudera_20191006213131_66cc1b6c-6fbd-4588-a5d0-2a14b0f9f3cb
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for 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>
Starting Job = job_1570416341774_0002, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1570416341774
_0002/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1570416341774_0002
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-10-06 21:32:13,667 Stage-1 map = 0%,  reduce = 0%
2019-10-06 21:32:33,017 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.07 sec
2019-10-06 21:32:49,592 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.75 sec
MapReduce Total cumulative CPU time: 5 seconds 750 msec
Ended Job = job_1570416341774_0002
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.75 sec   HDFS Read: 2352692 HDFS Write: 52 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 750 msec
OK
0       9
1       3386
2       3441
3       1839
4       582
5       164
6       74
7       42
8       14
Time taken: 55.615 seconds, Fetched: 9 row(s)
```

Query-5: Creating a new views in a database can hide personal information and users who have access to the view will not be able to view other hidden messages.

Command: create view if not exists zomato_filtered as select restaurant_id, restaurant_name, price_range, currency, table_booking from zomato;

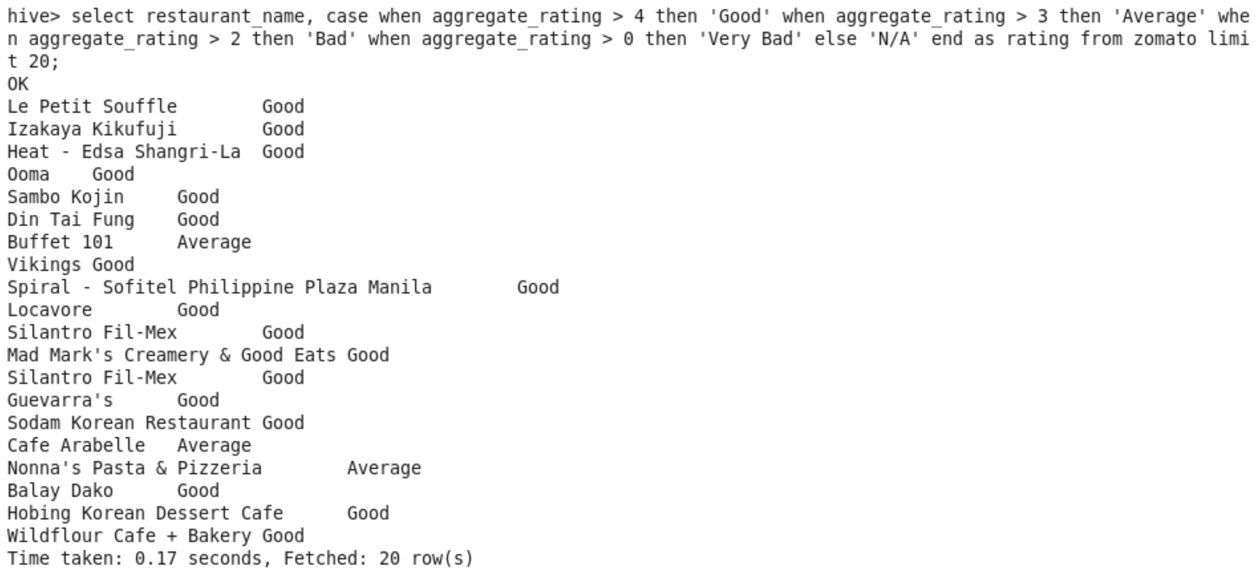
select * from zomato_filtered limit 10;



```
hive> create view if not exists zomato_filtered as select restaurant_id, restaurant_name, price_range, currency, tabl
e_booking from zomato;
OK
Time taken: 0.312 seconds
hive> select * from zomato_filtered limit 10;
OK
6317637 Le Petit Souffle        3       Botswana Pula(P)        true
6304287 Izakaya Kikufuji        3       Botswana Pula(P)        true
6300002 Heat - Edsa Shangri-La  4       Botswana Pula(P)        true
6318506 Ooma    4       Botswana Pula(P)        false
6314302 Sambo Kojin     4       Botswana Pula(P)        true
18189371        Din Tai Fung    3       Botswana Pula(P)        false
6300781 Buffet 101      4       Botswana Pula(P)        true
6301290 Vikings 4       Botswana Pula(P)        true
6300010 Spiral - Sofitel Philippine Plaza Manila        4       Botswana Pula(P)        true
6314987 Locavore        3       Botswana Pula(P)        true
Time taken: 0.134 seconds, Fetched: 10 row(s)
```

Query-6: Categorize the restaurants by their rating as "Good", "Average", "Bad", and "Very Bad".

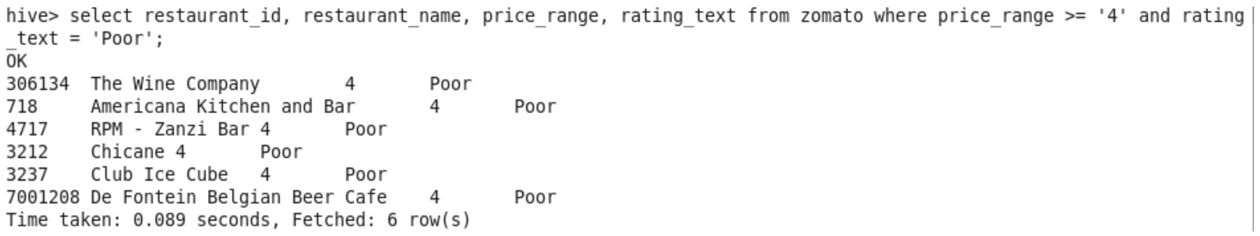
Command: select restaurant_name, case when aggregate_rating > 4 then 'Good' when aggregate_rating > 3 then 'Average' when aggregate_rating > 2 then 'Bad' when aggregate_rating > 0 then 'Very Bad' else 'N/A' end as rating from zomato limit 20;



```
hive> select restaurant_name, case when aggregate_rating > 4 then 'Good' when aggregate_rating > 3 then 'Average' whe
n aggregate_rating > 2 then 'Bad' when aggregate_rating > 0 then 'Very Bad' else 'N/A' end as rating from zomato limi
t 20;
OK
Le Petit Souffle        Good
Izakaya Kikufuji        Good
Heat - Edsa Shangri-La  Good
Ooma    Good
Sambo Kojin     Good
Din Tai Fung    Good
Buffet 101      Average
Vikings Good
Spiral - Sofitel Philippine Plaza Manila        Good
Locavore        Good
Silantro Fil-Mex        Good
Mad Mark's Creamery & Good Eats Good
Silantro Fil-Mex        Good
Guevarra's      Good
Sodam Korean Restaurant Good
Cafe Arabelle   Average
Nonna's Pasta & Pizzeria        Average
Balay Dako      Good
Hobing Korean Dessert Cafe      Good
Wildflour Cafe + Bakery Good
Time taken: 0.17 seconds, Fetched: 20 row(s)
```

Query-7: To print restaurants with highest prices and bad rating.

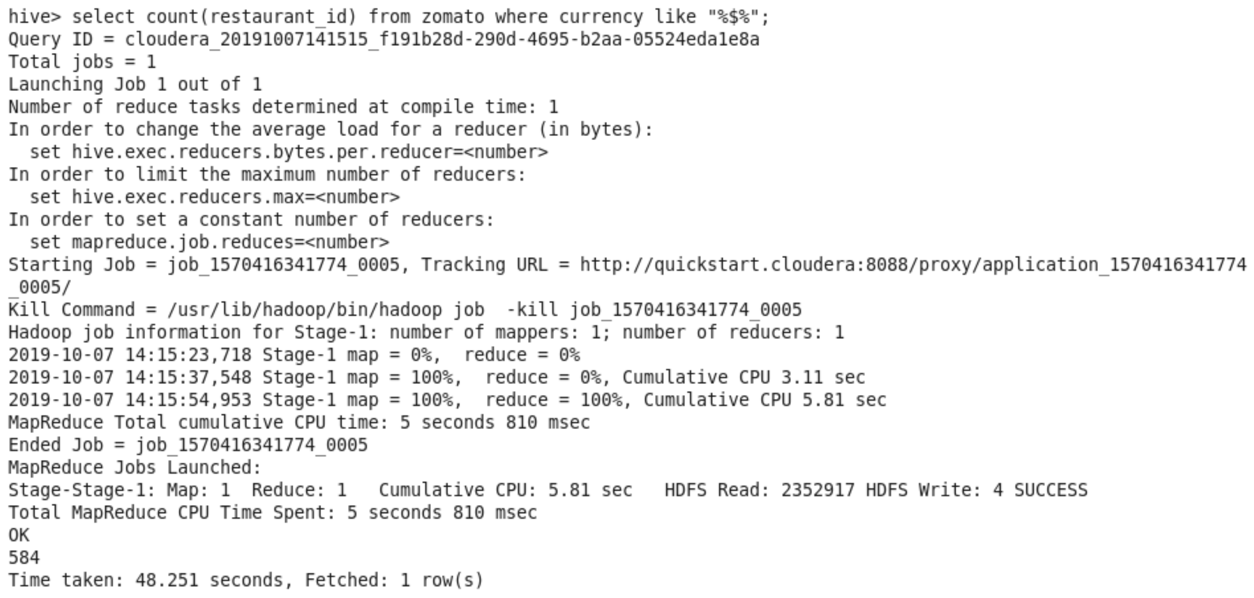
Command: select restaurant_id, restaurant_name, price_range, rating_text from zomato where price_range >= '4' and rating_text = 'Poor';



```
hive> select restaurant_id, restaurant_name, price_range, rating_text from zomato where price_range >= '4' and rating
_text = 'Poor';
OK
306134  The Wine Company        4       Poor
718     Americana Kitchen and Bar       4       Poor
4717    RPM - Zanzi Bar 4       Poor
3212    Chicane 4       Poor
3237    Club Ice Cube   4       Poor
7001208 De Fontein Belgian Beer Cafe    4       Poor
Time taken: 0.089 seconds, Fetched: 6 row(s)
```

Query-8: To count number of restaurants using dollars as their currency.

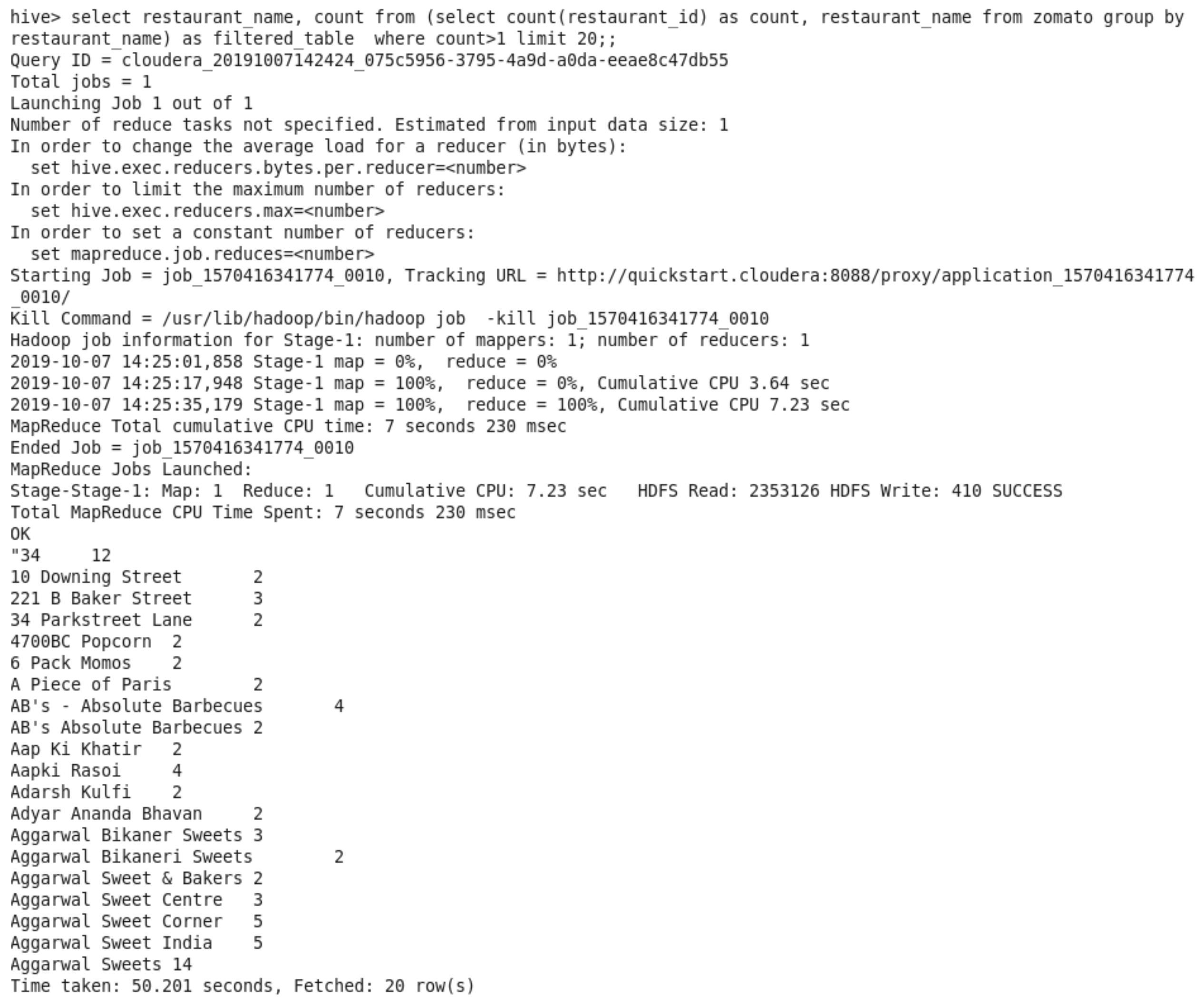
Command: select count(restaurant_id) from zomato where currency like "%$%";



```
hive> select count(restaurant_id) from zomato where currency like "%$%";
Query ID = cloudera_20191007141515_f191b28d-290d-4695-b2aa-05524eda1e8a
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 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>
Starting Job = job_1570416341774_0005, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1570416341774
_0005/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1570416341774_0005
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-10-07 14:15:23,718 Stage-1 map = 0%,  reduce = 0%
2019-10-07 14:15:37,548 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.11 sec
2019-10-07 14:15:54,953 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.81 sec
MapReduce Total cumulative CPU time: 5 seconds 810 msec
Ended Job = job_1570416341774_0005
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.81 sec   HDFS Read: 2352917 HDFS Write: 4 SUCCESS
Total MapReduce CPU Time Spent: 5 seconds 810 msec
OK
584
Time taken: 48.251 seconds, Fetched: 1 row(s)
```

Query-9: To find restaurants with two or more locations.

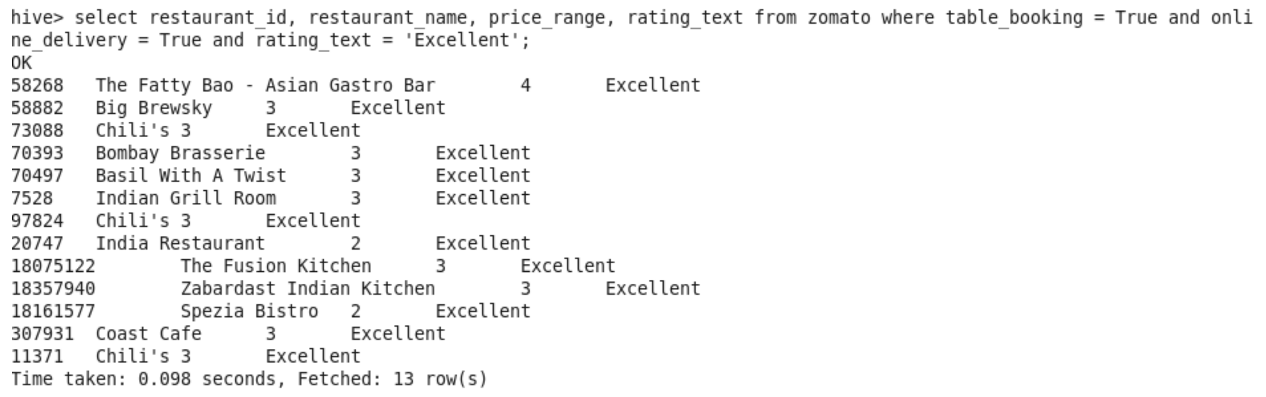
Command: select restaurant_name, count from (select count(restaurant_id) as count, restaurant_name from zomato group by restaurant_name) as filtered_table where count>1 limit 20;



```
hive> select restaurant_name, count from (select count(restaurant_id) as count, restaurant_name from zomato group by
restaurant_name) as filtered_table  where count>1 limit 20;;
Query ID = cloudera_20191007142424_075c5956-3795-4a9d-a0da-eeae8c47db55
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>
Starting Job = job_1570416341774_0010, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1570416341774
_0010/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1570416341774_0010
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2019-10-07 14:25:01,858 Stage-1 map = 0%,  reduce = 0%
2019-10-07 14:25:17,948 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.64 sec
2019-10-07 14:25:35,179 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 7.23 sec
MapReduce Total cumulative CPU time: 7 seconds 230 msec
Ended Job = job_1570416341774_0010
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 7.23 sec   HDFS Read: 2353126 HDFS Write: 410 SUCCESS
Total MapReduce CPU Time Spent: 7 seconds 230 msec
OK
"34     12
10 Downing Street       2
221 B Baker Street      3
34 Parkstreet Lane      2
4700BC Popcorn  2
6 Pack Momos    2
A Piece of Paris        2
AB's - Absolute Barbecues       4
AB's Absolute Barbecues 2
Aap Ki Khatir   2
Aapki Rasoi     4
Adarsh Kulfi    2
Adyar Ananda Bhavan     2
Aggarwal Bikaner Sweets 3
Aggarwal Bikaneri Sweets        2
Aggarwal Sweet & Bakers 2
Aggarwal Sweet Centre   3
Aggarwal Sweet Corner   5
Aggarwal Sweet India    5
Aggarwal Sweets 14
Time taken: 50.201 seconds, Fetched: 20 row(s)
```

Query-10: To print restaurant with best customer satisfaction.

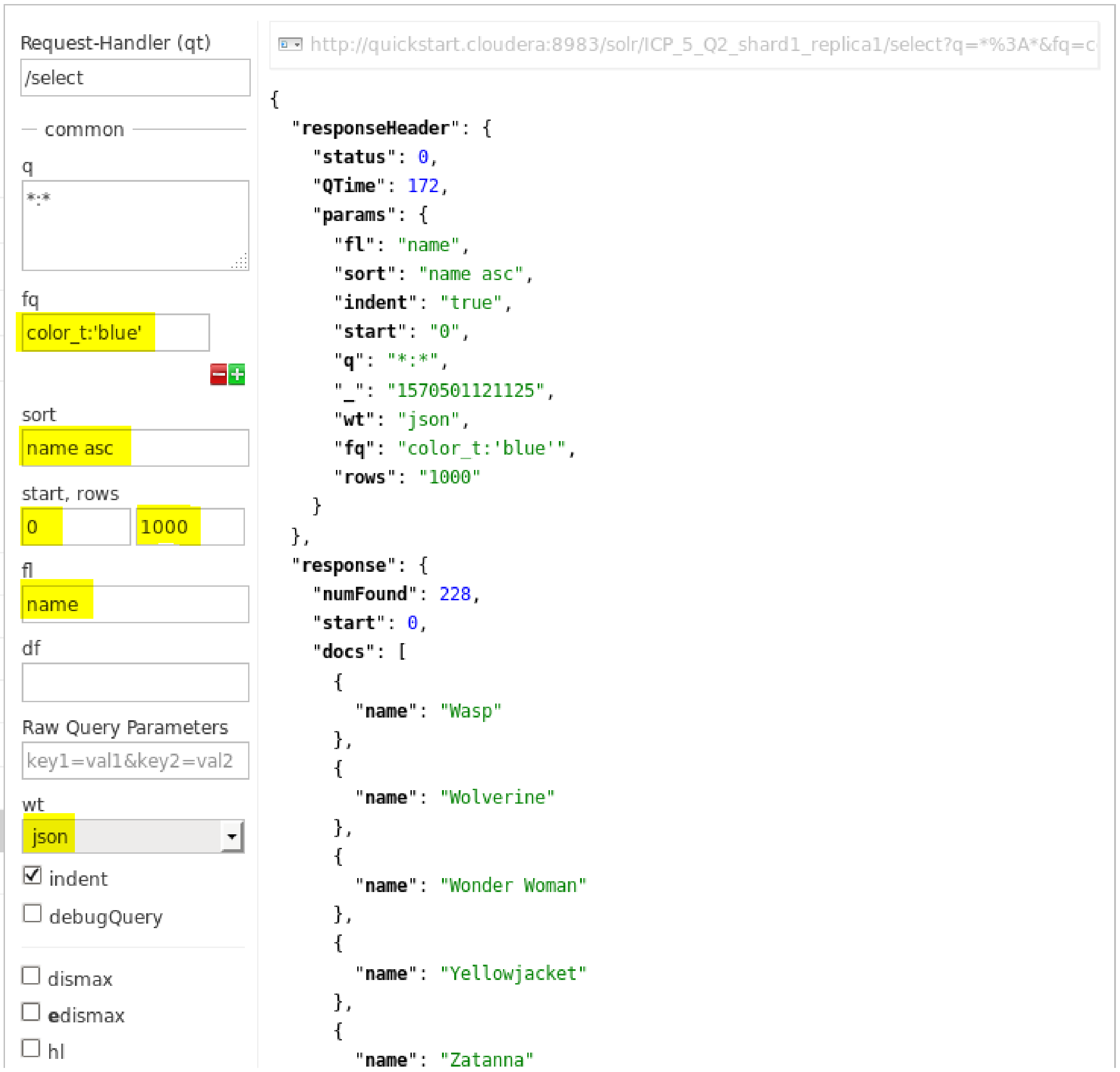
Command: select restaurant_id, restaurant_name, price_range, rating_text from zomato where table_booking = True and online_delivery = True and delivery_now = True and rating_text = 'Excellent';



```
hive> select restaurant_id, restaurant_name, price_range, rating_text from zomato where table_booking = True and onli
ne_delivery = True and rating_text = 'Excellent';
OK
58268   The Fatty Bao - Asian Gastro Bar        4       Excellent
58882   Big Brewsky     3       Excellent
73088   Chili's 3       Excellent
70393   Bombay Brasserie        3       Excellent
70497   Basil With A Twist      3       Excellent
7528    Indian Grill Room       3       Excellent
97824   Chili's 3       Excellent
20747   India Restaurant        2       Excellent
18075122        The Fusion Kitchen      3       Excellent
18357940        Zabardast Indian Kitchen        3       Excellent
18161577        Spezia Bistro   2       Excellent
307931  Coast Cafe      3       Excellent
11371   Chili's 3       Excellent
Time taken: 0.098 seconds, Fetched: 13 row(s)
```

Part-2

Query-1: To print names where color is blue from top 1000 records and print them in asc order of name and in json.



Request-Handler (qt)
/select

common

q
:

fq
color_t:'blue'

sort
name asc

start, rows
0 1000

fl
name

df

Raw Query Parameters
key1=val1&key2=val2

wt
json

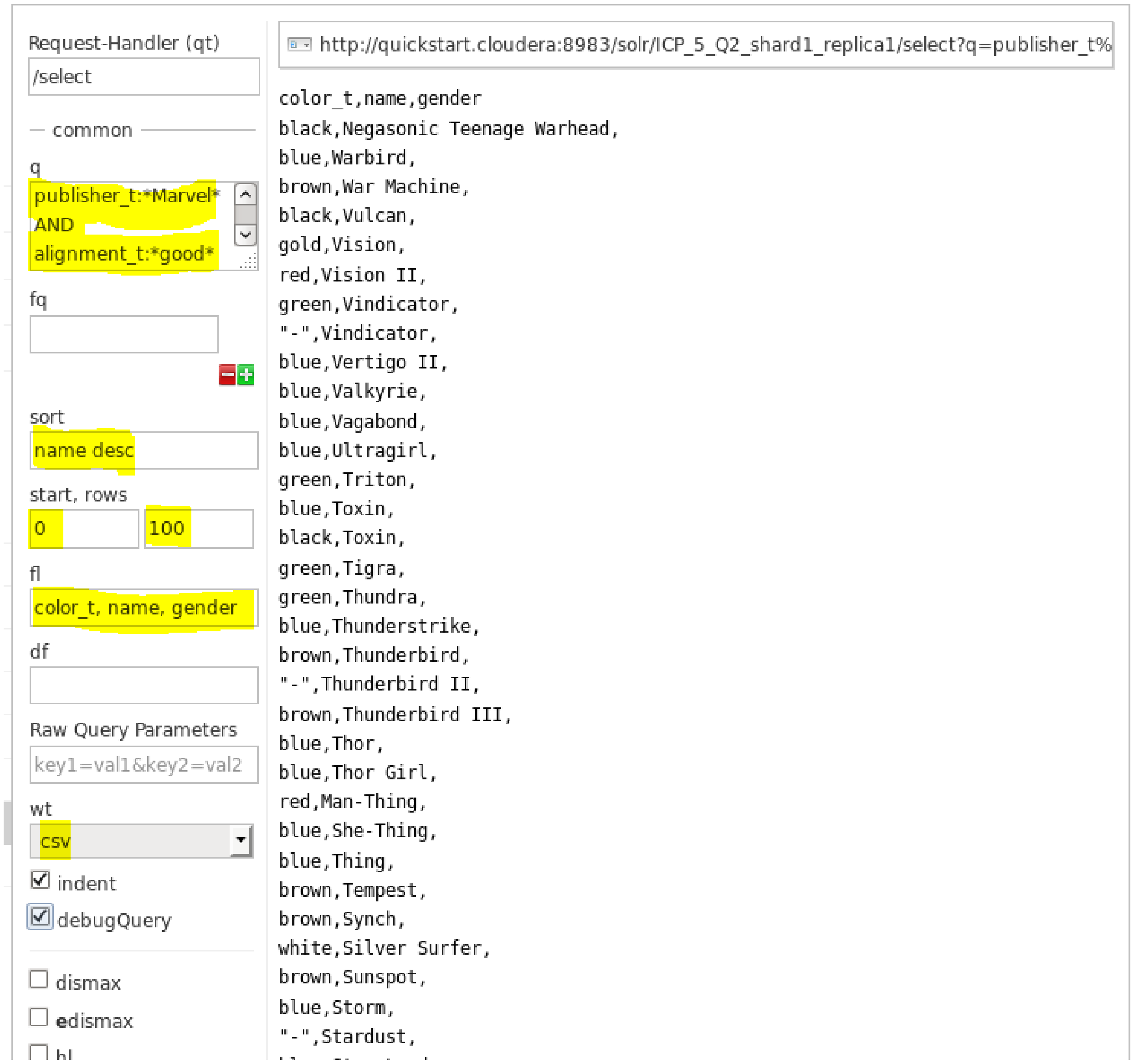
☑ indent
☐ debugQuery

☐ dismax
☐ edismax
☐ hl

http://quickstart.cloudera:8983/solr/ICP_5_Q2_shard1_replica1/select?q=*%3A*&fq=c

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 172,
    "params": {
      "fl": "name",
      "sort": "name asc",
      "indent": "true",
      "start": "0",
      "q": "*:*",
      "_": "1570501121125",
      "wt": "json",
      "fq": "color_t:'blue'",
      "rows": "1000"
    }
  },
  "response": {
    "numFound": 228,
    "start": 0,
    "docs": [
      {
        "name": "Wasp"
      },
      {
        "name": "Wolverine"
      },
      {
        "name": "Wonder Woman"
      },
      {
        "name": "Yellowjacket"
      },
      {
        "name": "Zatanna"
```

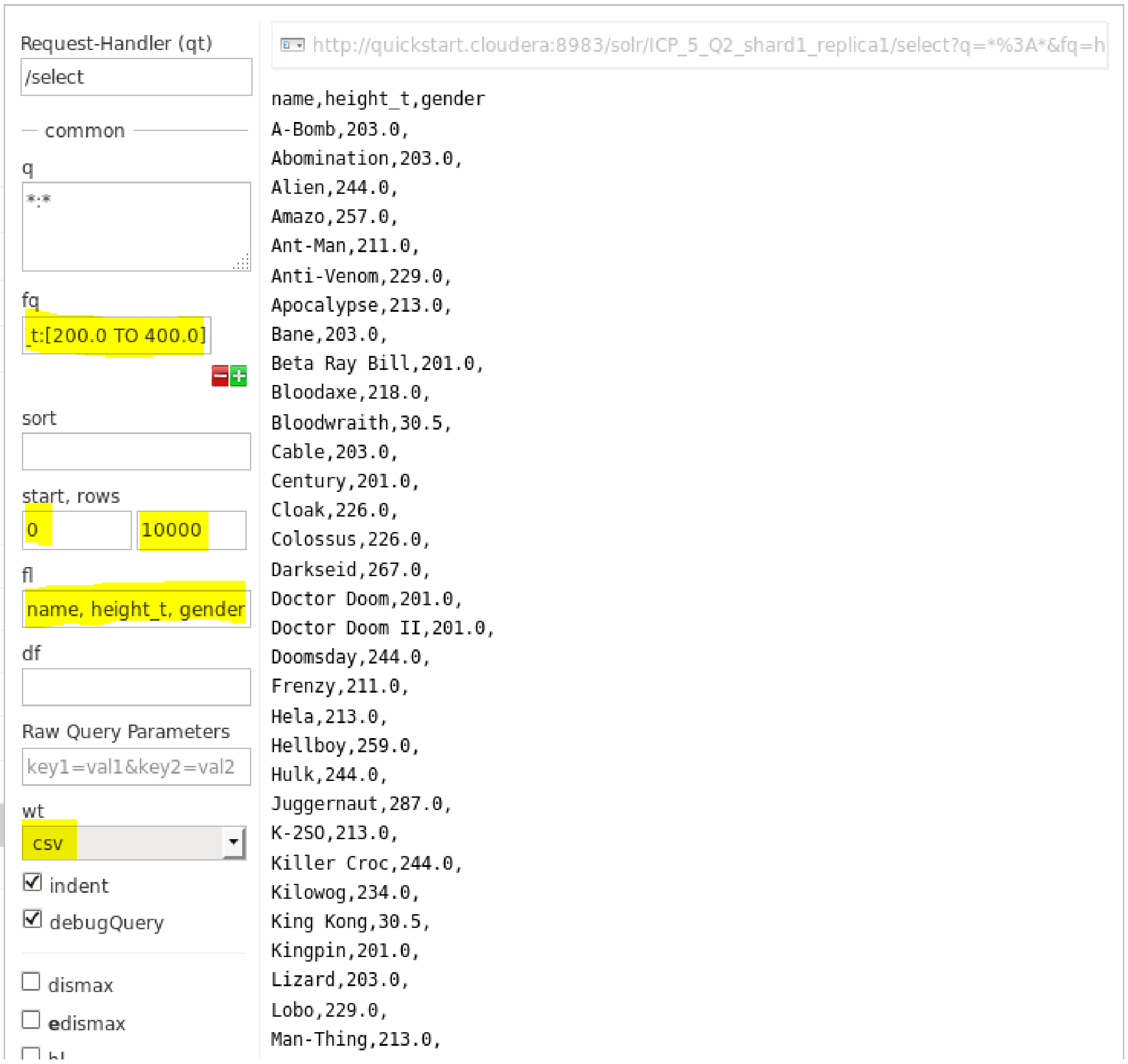
Query-2: Print name, gender and color from the table and print them as csv, in desc order of name and select from top 100 records.


Request-Handler (qt)
/select
common
q
publisher_t:*Marvel* AND alignment_t:*good*
fq
sort
name desc
start, rows
0 100
fl
color_t, name, gender
df
Raw Query Parameters
key1=val1&key2=val2
wt
csv
☑ indent
☑ debugQuery
☐ dismax
☐ edismax
☐ hl

http://quickstart.cloudera:8983/solr/ICP_5_Q2_shard1_replica1/select?q=publisher_t%

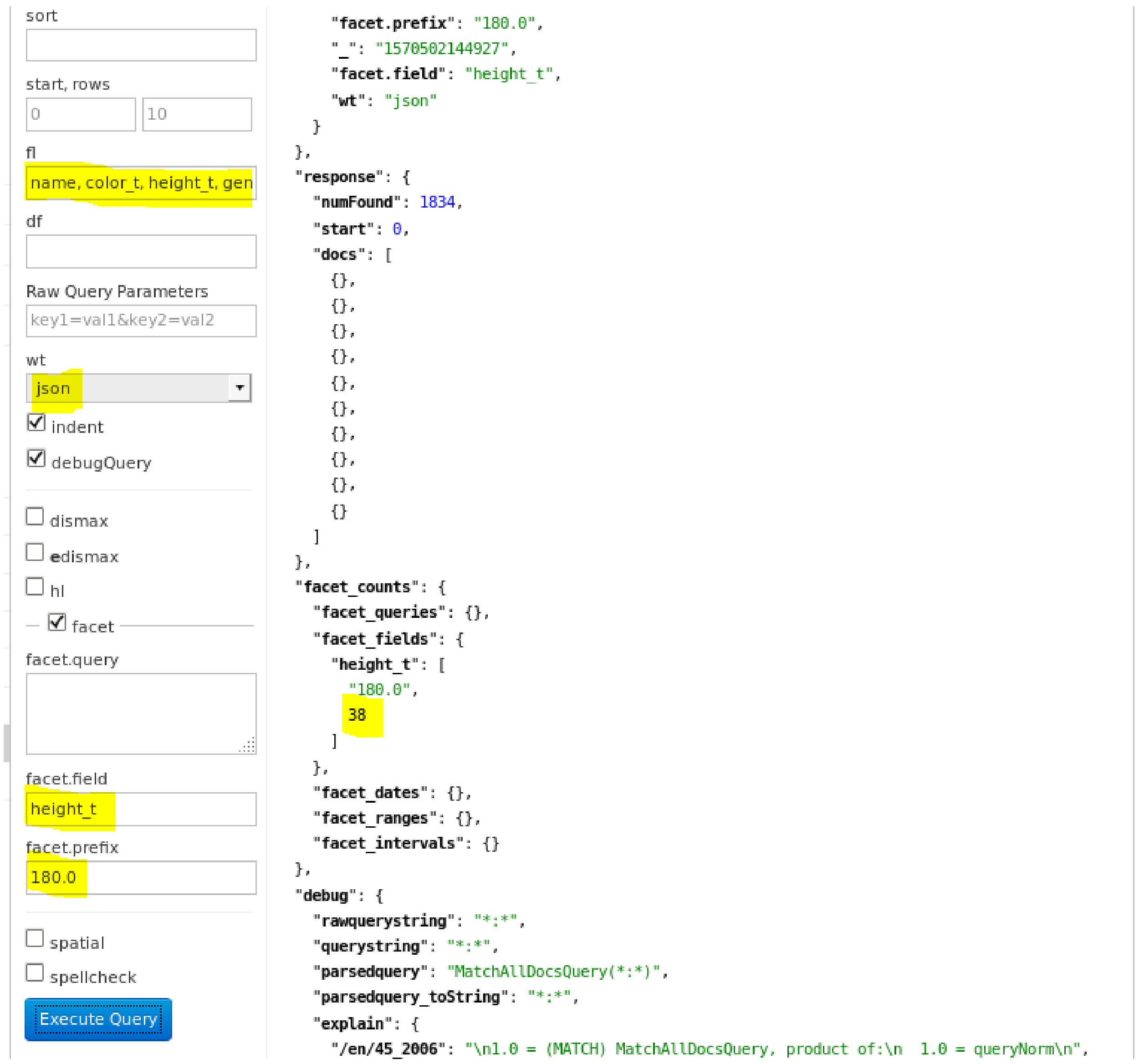
```
color_t,name,gender
black,Negasonic Teenage Warhead,
blue,Warbird,
brown,War Machine,
black,Vulcan,
gold,Vision,
red,Vision II,
green,Vindicator,
"-",Vindicator,
blue,Vertigo II,
blue,Valkyrie,
blue,Vagabond,
blue,Ultragirl,
green,Triton,
blue,Toxin,
black,Toxin,
green,Tigra,
green,Thundra,
blue,Thunderstrike,
brown,Thunderbird,
"-",Thunderbird II,
brown,Thunderbird III,
blue,Thor,
blue,Thor Girl,
red,Man-Thing,
blue,She-Thing,
blue,Thing,
brown,Tempest,
brown,Synch,
white,Silver Surfer,
brown,Sunspot,
blue,Storm,
"-",Stardust,
```

Query-3: Select name, height and gender as csv from top 10000 records where height is between 200 and 400



```
name,height_t,gender
A-Bomb,203.0,
Abomination,203.0,
Alien,244.0,
Amazo,257.0,
Ant-Man,211.0,
Anti-Venom,229.0,
Apocalypse,213.0,
Bane,203.0,
Beta Ray Bill,201.0,
Bloodaxe,218.0,
Bloodwraith,30.5,
Cable,203.0,
Century,201.0,
Cloak,226.0,
Colossus,226.0,
Darkseid,267.0,
Doctor Doom,201.0,
Doctor Doom II,201.0,
Doomsday,244.0,
Frenzy,211.0,
Hela,213.0,
Hellboy,259.0,
Hulk,244.0,
Juggernaut,287.0,
K-2SO,213.0,
Killer Croc,244.0,
Kilowog,234.0,
King Kong,30.5,
Kingpin,201.0,
Lizard,203.0,
Lobo,229.0,
Man-Thing,213.0,
```

Query-4: To print name, color, height and gender as json where height is exactly equal to 180.0.


sort

start, rows
0 10

fl
name, color_t, height_t, gen

df

Raw Query Parameters
key1=val1&key2=val2

wt
json

[✓] indent
[✓] debugQuery

[ ] dismax
[ ] edismax
[ ] hl
[✓] facet

facet.query

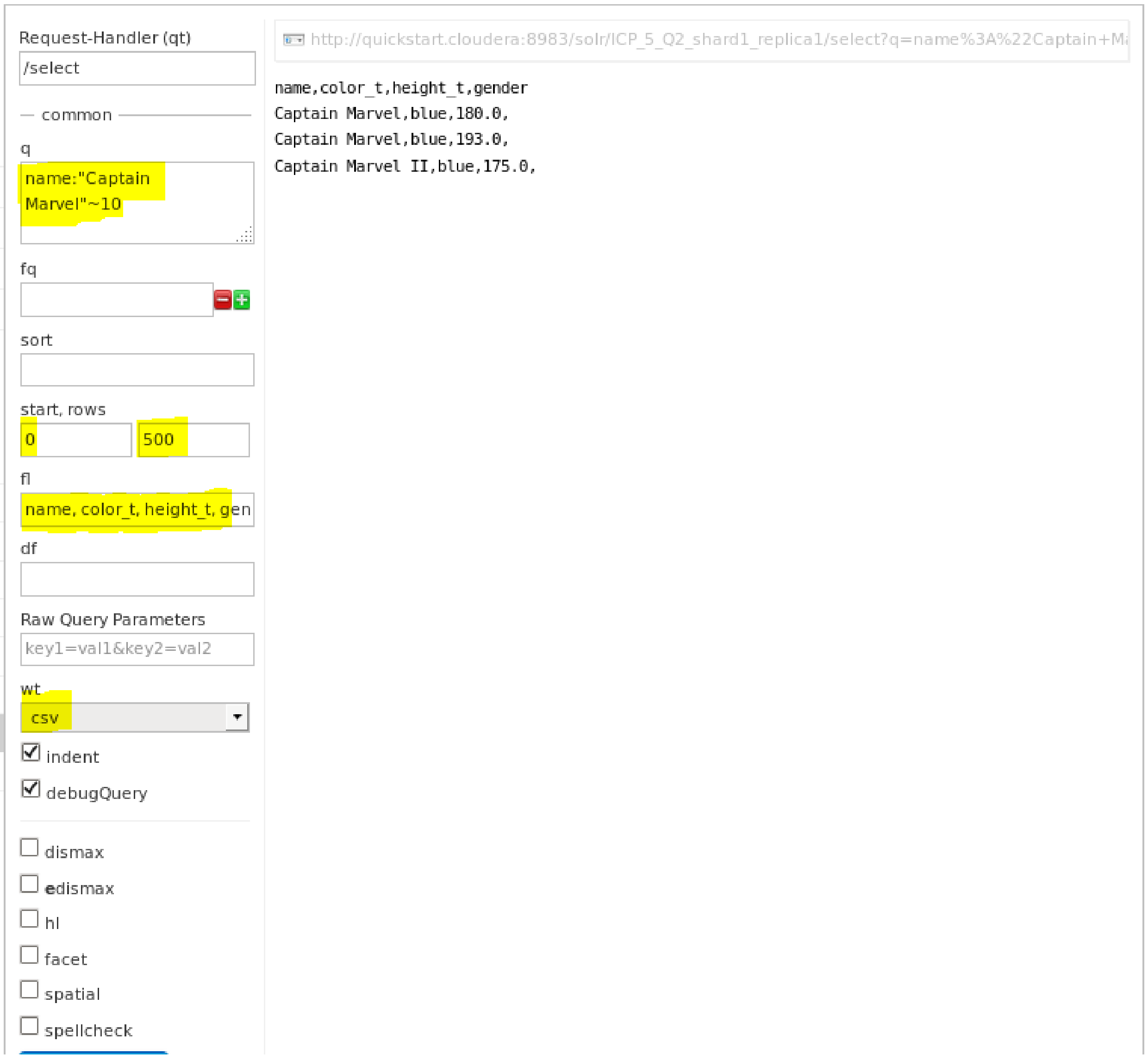
facet.field
height_t

facet.prefix
180.0

[ ] spatial
[ ] spellcheck

```
      "facet.prefix": "180.0",
      "_": "1570502144927",
      "facet.field": "height_t",
      "wt": "json"
    }
  },
  "response": {
    "numFound": 1834,
    "start": 0,
    "docs": [
      {},
      {},
      {},
      {},
      {},
      {},
      {},
      {},
      {},
      {}
    ]
  },
  "facet_counts": {
    "facet_queries": {},
    "facet_fields": {
      "height_t": [
        "180.0",
        38
      ]
    },
    "facet_dates": {},
    "facet_ranges": {},
    "facet_intervals": {}
  },
  "debug": {
    "rawquerystring": "*:*",
    "querystring": "*:*",
    "parsedquery": "MatchAllDocsQuery(*:*)",
    "parsedquery_toString": "*:*",
    "explain": {
      "/en/45_2006": "\n1.0 = (MATCH) MatchAllDocsQuery, product of:\n  1.0 = queryNorm\n",
```

Query-5: To do proximity search on words "Captain" and "Marvel" and print name, color from top 500 records.



Request-Handler (qt)
/select

common

q
name:"Captain Marvel"~10

fq

sort

start, rows
0 500

fl
name, color_t, height_t, gen

df

Raw Query Parameters
key1=val1&key2=val2

wt
csv

[✓] indent

[✓] debugQuery

[ ] dismax

[ ] edismax

[ ] hl

[ ] facet

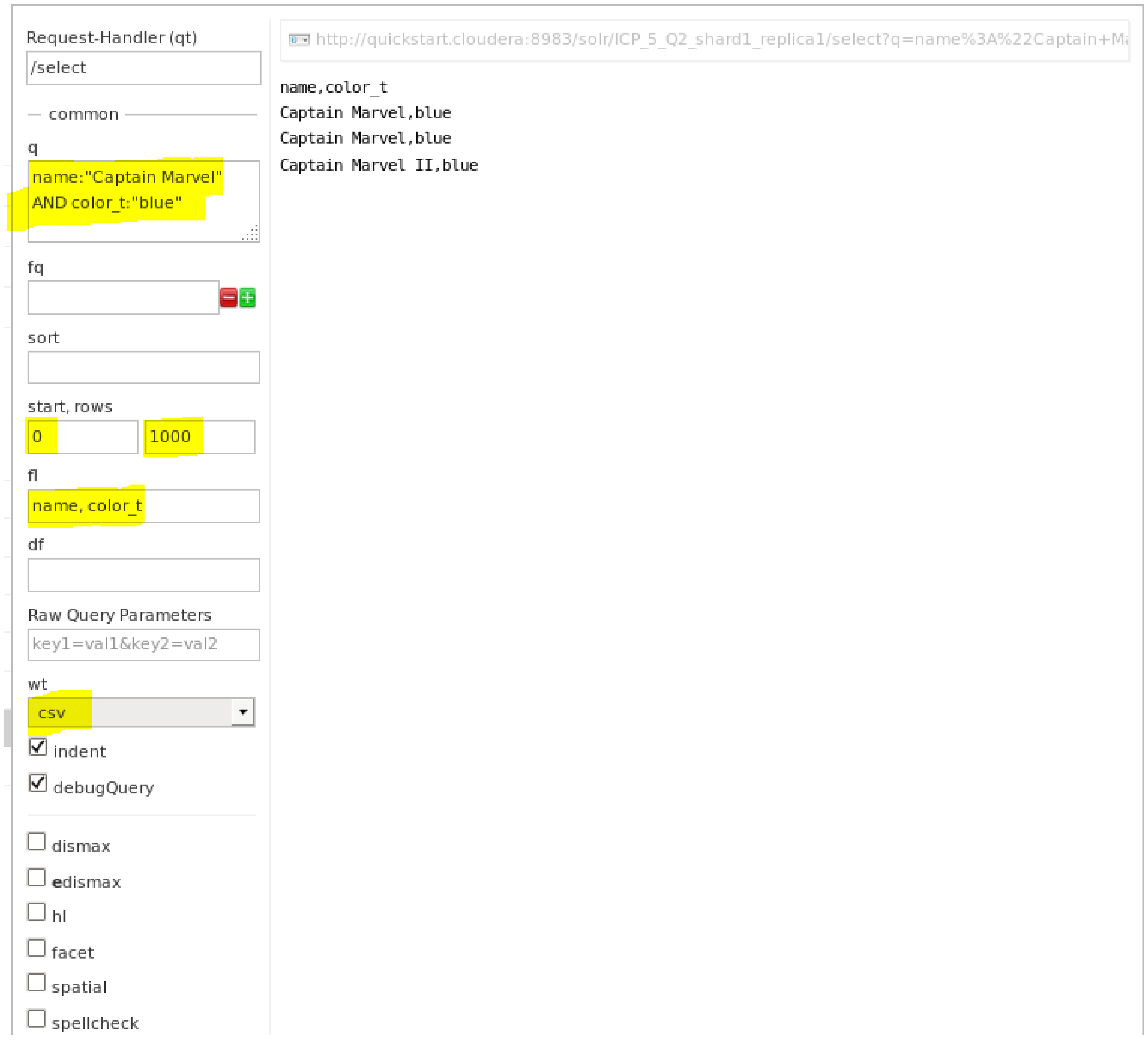
[ ] spatial

[ ] spellcheck

http://quickstart.cloudera:8983/solr/ICP_5_Q2_shard1_replica1/select?q=name%3A%22Captain+Ma

```
name,color_t,height_t,gender
Captain Marvel,blue,180.0,
Captain Marvel,blue,193.0,
Captain Marvel II,blue,175.0,
```

Query-6: To print all the records from 0 to 1000 where name is "Captain Marvel" and color is "blue" (using AND operator). Print as csv.



/select

q: name:"Captain Marvel" AND color_t:"blue"

fq:

sort:

start, rows: 0 1000

fl: name, color_t

df:

Raw Query Parameters: key1=val1&key2=val2

wt: csv

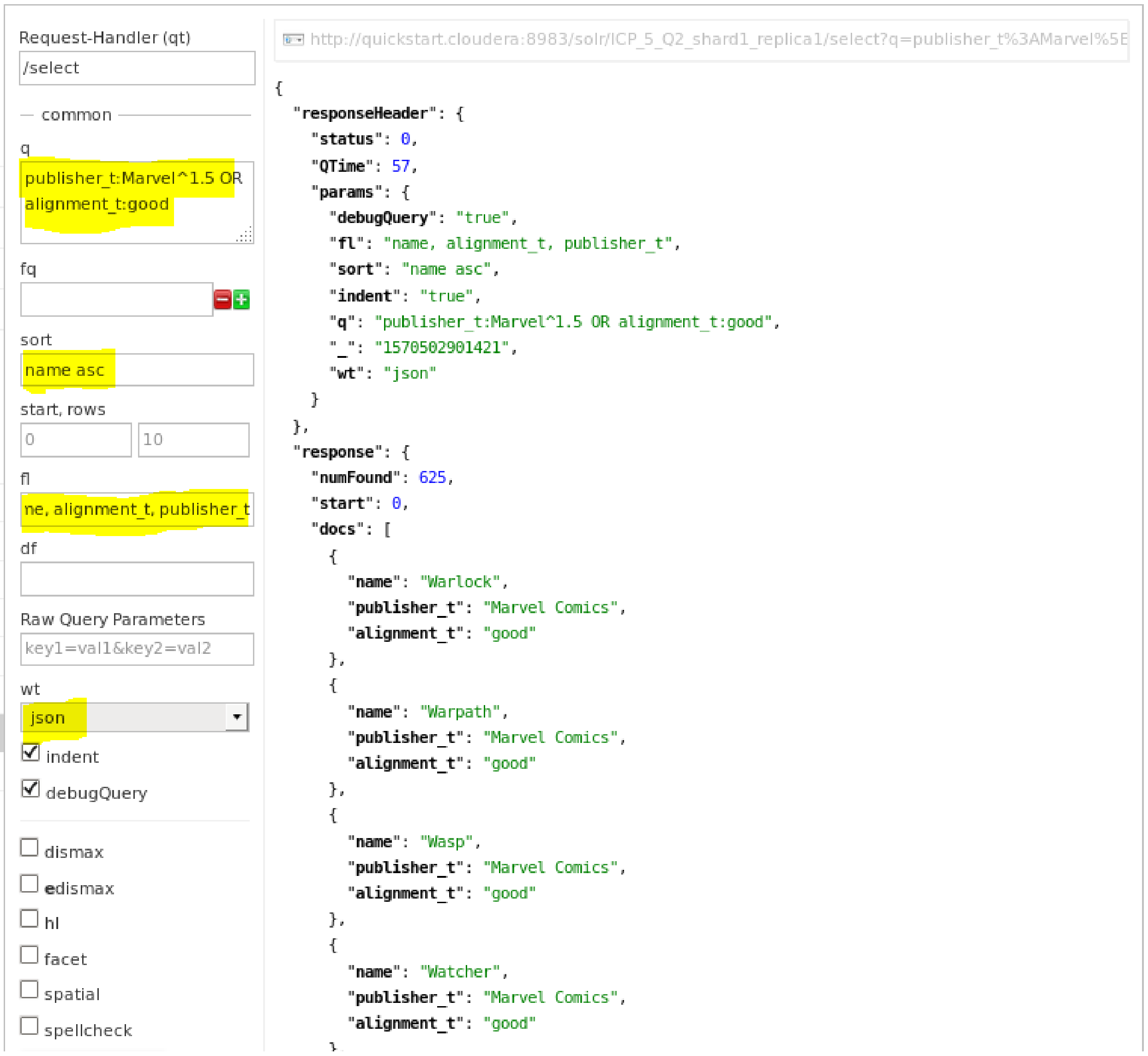
[✓] indent

[✓] debugQuery

http://quickstart.cloudera:8983/solr/ICP_5_Q2_shard1_replica1/select?q=name%3A%22Captain+Ma

```
name,color_t
Captain Marvel,blue
Captain Marvel,blue
Captain Marvel II,blue
```

Query-7: Boost queries: To find publisher as Marvel and alignment is good. Print in asc order of name and in json.



Request-Handler (qt)
/select

common

q
publisher_t:Marvel^1.5 OR alignment_t:good

fq

sort
name asc

start, rows
0 10

fl
ne, alignment_t, publisher_t

df

Raw Query Parameters
key1=val1&key2=val2

wt
json

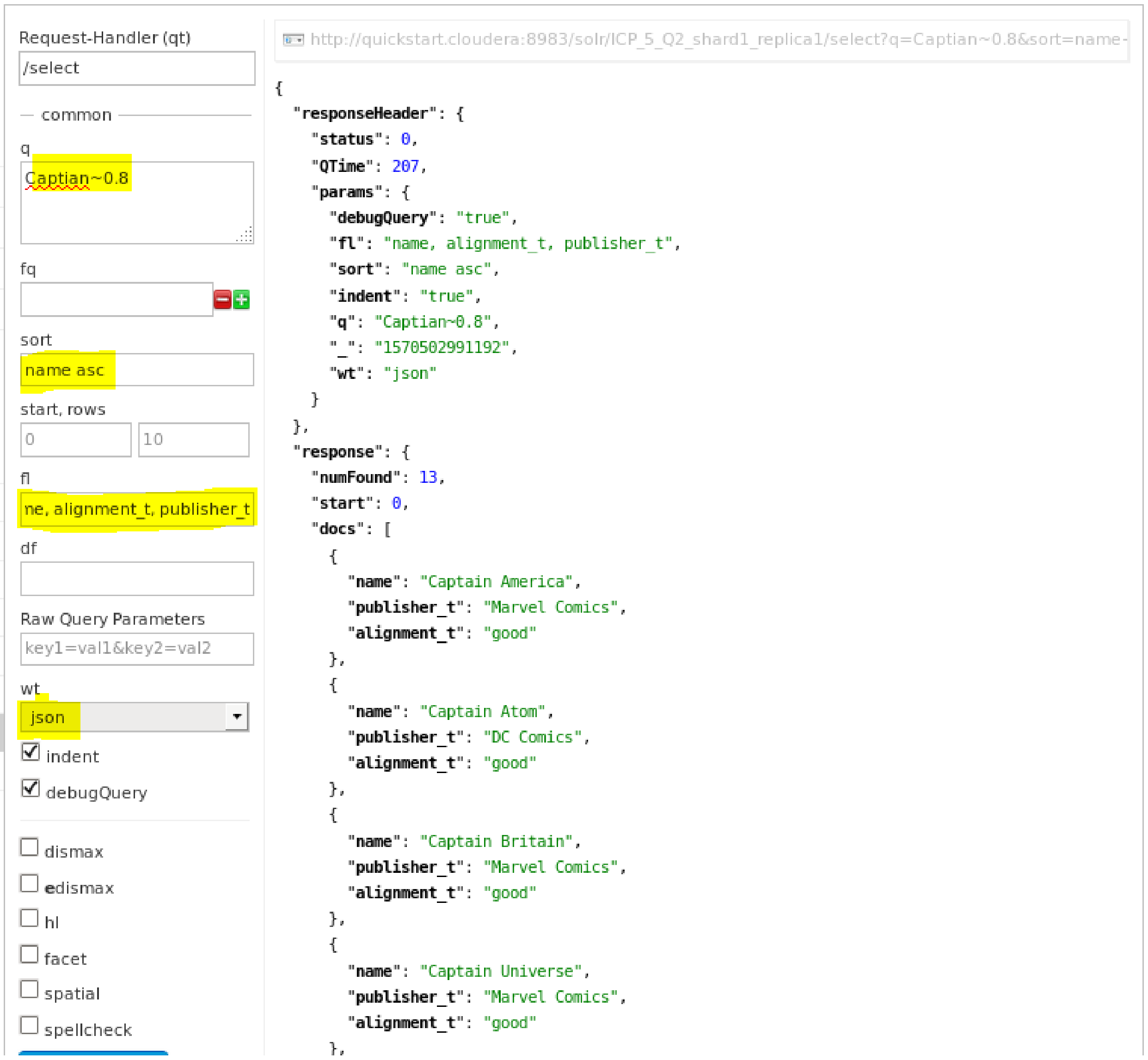
☑ indent
☑ debugQuery

☐ dismax
☐ edismax
☐ hl
☐ facet
☐ spatial
☐ spellcheck

http://quickstart.cloudera:8983/solr/ICP_5_Q2_shard1_replica1/select?q=publisher_t%3AMarvel%5E

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 57,
    "params": {
      "debugQuery": "true",
      "fl": "name, alignment_t, publisher_t",
      "sort": "name asc",
      "indent": "true",
      "q": "publisher_t:Marvel^1.5 OR alignment_t:good",
      "_": "1570502901421",
      "wt": "json"
    }
  },
  "response": {
    "numFound": 625,
    "start": 0,
    "docs": [
      {
        "name": "Warlock",
        "publisher_t": "Marvel Comics",
        "alignment_t": "good"
      },
      {
        "name": "Warpath",
        "publisher_t": "Marvel Comics",
        "alignment_t": "good"
      },
      {
        "name": "Wasp",
        "publisher_t": "Marvel Comics",
        "alignment_t": "good"
      },
      {
        "name": "Watcher",
        "publisher_t": "Marvel Comics",
        "alignment_t": "good"
      }
```

Query-8: To do fuzzy search on the word "Captian" and print in asc of name and in json.



Request-Handler (qt)
/select

common

q
Captian~0.8

fq

sort
name asc

start, rows
0 10

fl
ne, alignment_t, publisher_t

df

Raw Query Parameters
key1=val1&key2=val2

wt
json

[✓] indent

[✓] debugQuery

[ ] dismax

[ ] edismax

[ ] hl

[ ] facet

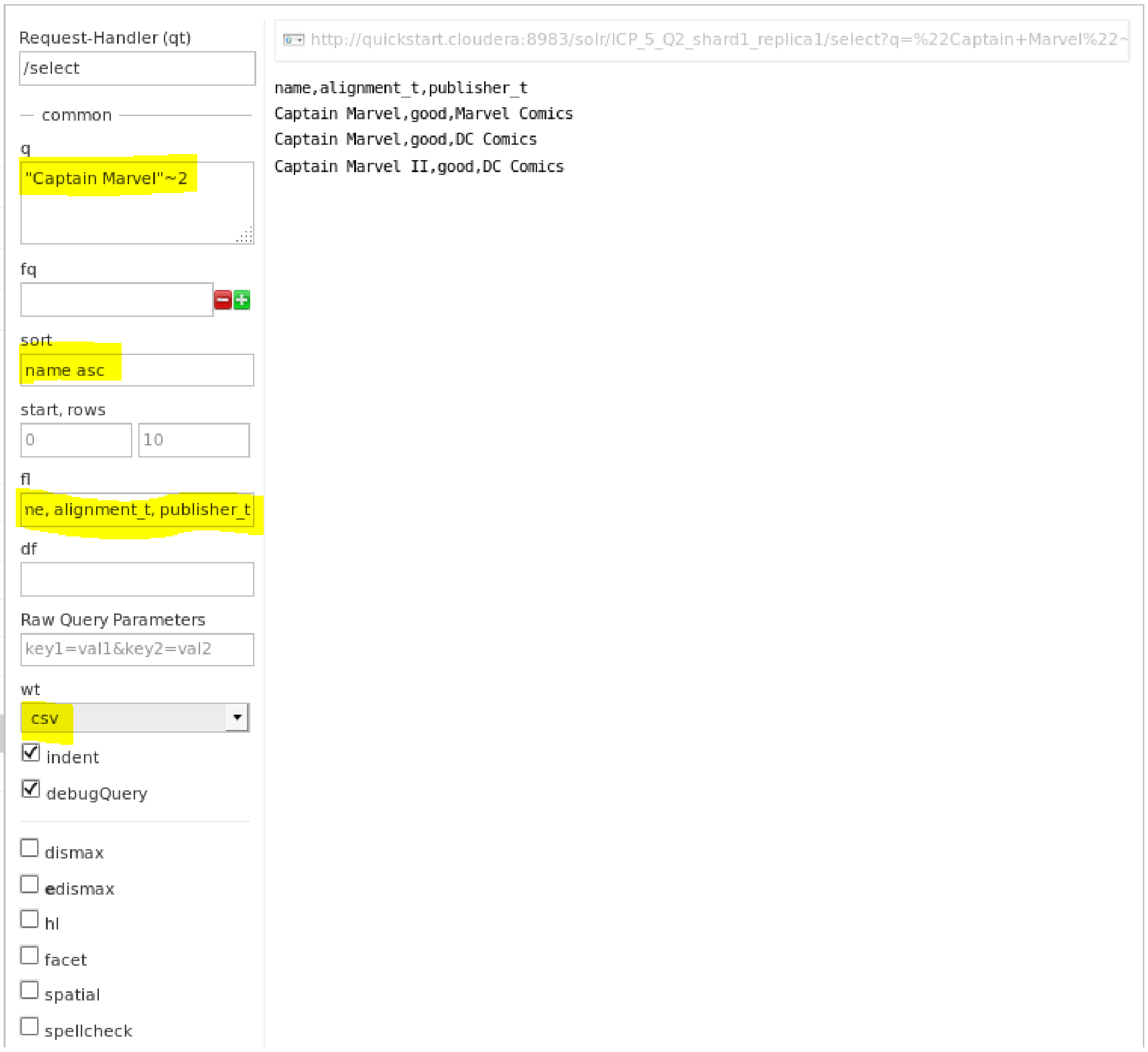
[ ] spatial

[ ] spellcheck

http://quickstart.cloudera:8983/solr/ICP_5_Q2_shard1_replica1/select?q=Captian~0.8&sort=name-

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 207,
    "params": {
      "debugQuery": "true",
      "fl": "name, alignment_t, publisher_t",
      "sort": "name asc",
      "indent": "true",
      "q": "Captian~0.8",
      "_": "1570502991192",
      "wt": "json"
    }
  },
  "response": {
    "numFound": 13,
    "start": 0,
    "docs": [
      {
        "name": "Captain America",
        "publisher_t": "Marvel Comics",
        "alignment_t": "good"
      },
      {
        "name": "Captain Atom",
        "publisher_t": "DC Comics",
        "alignment_t": "good"
      },
      {
        "name": "Captain Britain",
        "publisher_t": "Marvel Comics",
        "alignment_t": "good"
      },
      {
        "name": "Captain Universe",
        "publisher_t": "Marvel Comics",
        "alignment_t": "good"
      },
```

Query-9: To do proximity search on the words "Captain" and "Marvel" with distance as 2. Print in csv format and asc order of name.


Request-Handler (qt)
/select

common

q
"Captain Marvel"~2

fq

sort
name asc

start, rows
0 10

fl
ne, alignment_t, publisher_t

df

Raw Query Parameters
key1=val1&key2=val2

wt
csv

[✓] indent

[✓] debugQuery

[ ] dismax

[ ] edismax

[ ] hl

[ ] facet

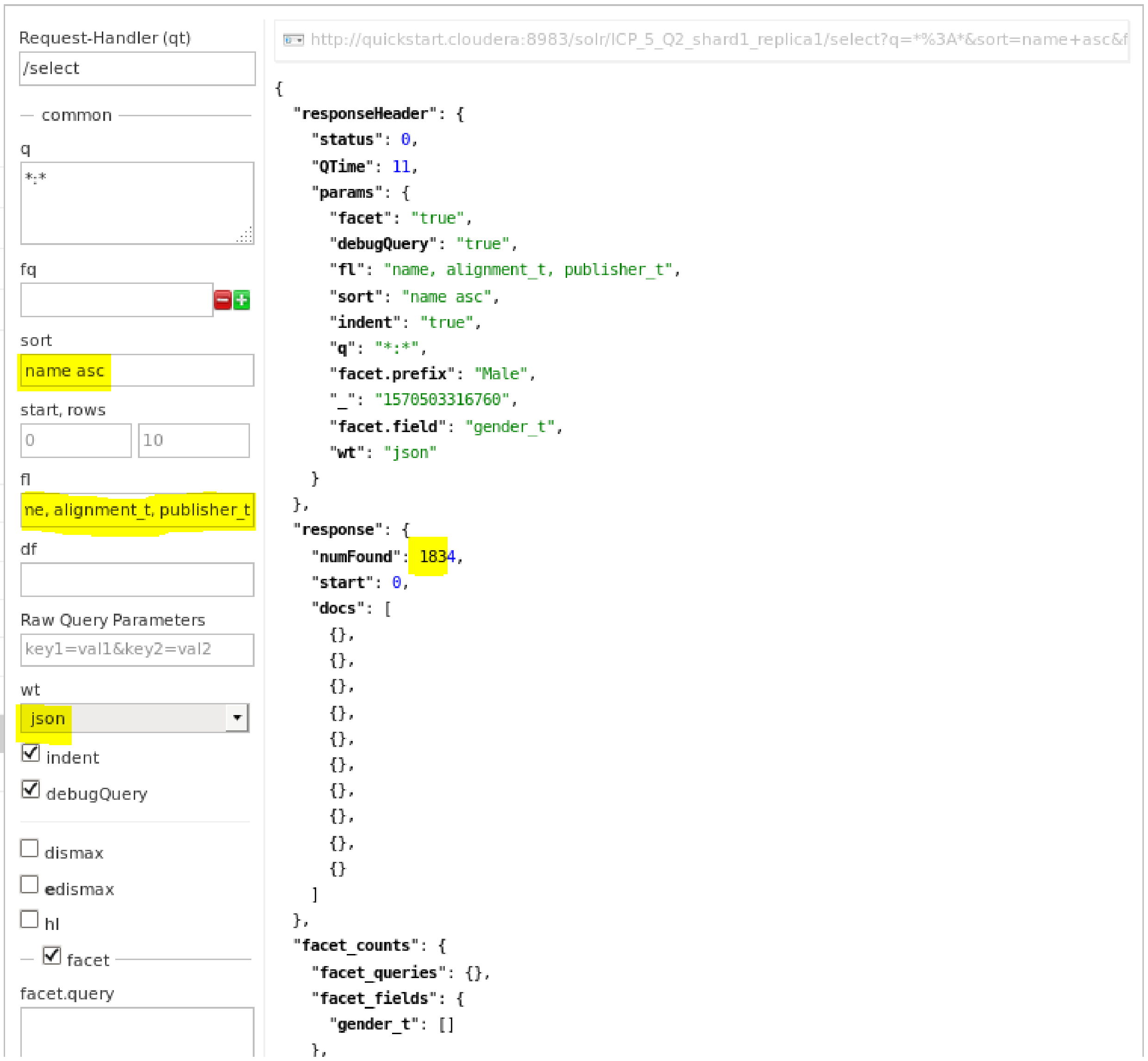
[ ] spatial

[ ] spellcheck

http://quickstart.cloudera:8983/solr/ICP_5_Q2_shard1_replica1/select?q=%22Captain+Marvel%22~

```
name,alignment_t,publisher_t
Captain Marvel,good,Marvel Comics
Captain Marvel,good,DC Comics
Captain Marvel II,good,DC Comics
```

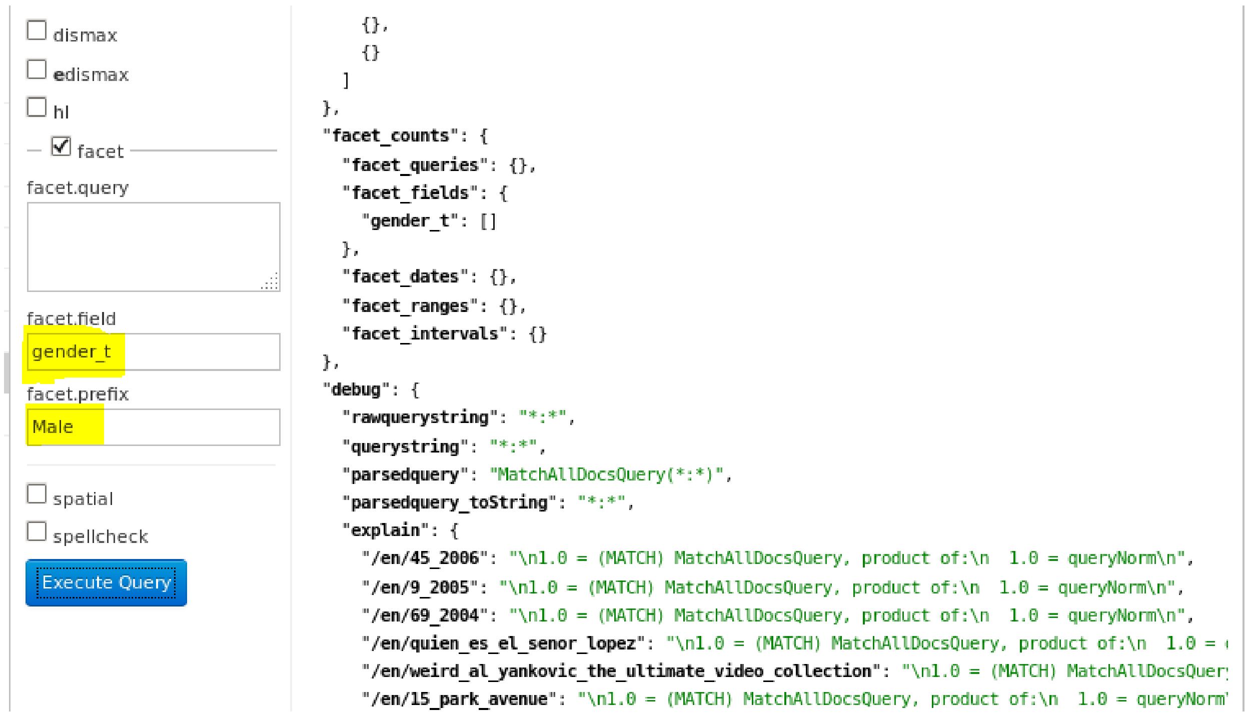
Query-10: To find number of Males.



Request-Handler (qt)
/select
common
q
:
fq
sort
name asc
start, rows
0 10
fl
ne, alignment_t, publisher_t
df
Raw Query Parameters
key1=val1&key2=val2
wt
json
☑ indent
☑ debugQuery
☐ dismax
☐ edismax
☐ hl
☑ facet
facet.query

http://quickstart.cloudera:8983/solr/ICP_5_Q2_shard1_replica1/select?q=*%3A*&sort=name+asc&f

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 11,
    "params": {
      "facet": "true",
      "debugQuery": "true",
      "fl": "name, alignment_t, publisher_t",
      "sort": "name asc",
      "indent": "true",
      "q": "*:*",
      "facet.prefix": "Male",
      "_": "1570503316760",
      "facet.field": "gender_t",
      "wt": "json"
    }
  },
  "response": {
    "numFound": 1834,
    "start": 0,
    "docs": [
      {},
      {},
      {},
      {},
      {},
      {},
      {},
      {},
      {},
      {}
    ]
  },
  "facet_counts": {
    "facet_queries": {},
    "facet_fields": {
      "gender_t": []
    },
```



facet.field: gender_t
facet.prefix: Male

```
      {},
      {}
    ]
  },
  "facet_counts": {
    "facet_queries": {},
    "facet_fields": {
      "gender_t": []
    },
    "facet_dates": {},
    "facet_ranges": {},
    "facet_intervals": {}
  },
  "debug": {
    "rawquerystring": "*:*",
    "querystring": "*:*",
    "parsedquery": "MatchAllDocsQuery(*:*)",
    "parsedquery_toString": "*:*",
    "explain": {
      "/en/45_2006": "\n1.0 = (MATCH) MatchAllDocsQuery, product of:\n  1.0 = queryNorm\n",
      "/en/9_2005": "\n1.0 = (MATCH) MatchAllDocsQuery, product of:\n  1.0 = queryNorm\n",
      "/en/69_2004": "\n1.0 = (MATCH) MatchAllDocsQuery, product of:\n  1.0 = queryNorm\n",
      "/en/quien_es_el_senor_lopez": "\n1.0 = (MATCH) MatchAllDocsQuery, product of:\n  1.0 =
      "/en/weird_al_yankovic_the_ultimate_video_collection": "\n1.0 = (MATCH) MatchAllDocsQuer
      "/en/15_park_avenue": "\n1.0 = (MATCH) MatchAllDocsQuery, product of:\n  1.0 = queryNorm
```

Conclusion

In this Lab Assignment we were able to revise ICP-1 to ICP-7. This lab assignment made us to understand Hive, Solr and MapReduce concepts.