ENGINEERING OF BIG DATA SYSTEMS

AMAZON CUSTOMER REVIEW OF GROCERY PRODUCTS ANALYSIS

Apoorva Mishra
001438598

TABLE OF CONTENTS

Problem Statement

Implement various Big Data Technologies such as Hadoop Map Reduce, HIVE, MongoDB, Apache Pig on Amazon Dataset to analyze various aspects of dataset
Provide visualization Insights using Tableau

Dataset

Dataset: Amazon Customer Reviews on Grocery products
`https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Grocery_v1_00.tsv.gz`

Fields Description:
DATA COLUMNS:
marketplace - 2 letter country code of the marketplace where the review was written.
customer_id - Random identifier that can be used to aggregate reviews written by a single author.
review_id - The unique ID of the review.
product_id - The unique Product ID the review pertains to. In the multilingual dataset the reviews for the same product in different countries can be grouped by the same product_id.
product_parent - Random identifier that can be used to aggregate reviews for the same product.
product_title - Title of the product.
product_category - Broad product category that can be used to group reviews (also used to group the dataset into coherent parts).
star_rating - The 1-5 star rating of the review.
helpful_votes - Number of helpful votes.
total_votes - Number of total votes the review received.
vine - Review was written as part of the Vine program.
verified_purchase - The review is on a verified purchase.
review_headline - The title of the review.
review_body - The review text.
review_date - The date the review was written.

Data Format:
Tab ('\t') separated text file, without quote or escape characters.
First line in each file is header; 1 line corresponds to 1 record.

Objectives

- Find the average product rating reviews for each product
- Find user who has reviewed the product
- Find the year in which the product was reviewed
- Verified /non-verified purchase of the overall products
- Verified Products along with their minimum and maximum ratings
- Find the total number of products in each product category
- Find the daily review count of all products
- Find the total number of products for each rating
- Find Top 5 products for each rating
- Find Top 5 verified purchases
-

Hadoop(HDFS) MapReduce

Case 1: Find the average product rating reviews for each product

```
 package com.finalproject;

import org.apache.hadoop.fs.Path;
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.io.Text;
import java.io.IOException;

// Find the average product rating reviews for each product

public class DriverClass {

    public static void main(String[] args) throws IOException,
ClassNotFoundException, InterruptedException
    {
        try {
            long startTime = System.currentTimeMillis();
            Job job = Job.getInstance();
            job.setJarByClass(DriverClass.class);

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

            job.setMapperClass(MapperClass.class);
            job.setReducerClass(ReducerClass.class);
            job.setCombinerClass(ReducerClass.class);

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

            job.setNumReduceTasks(1);

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

            job.waitForCompletion(true);

        } catch (Exception e) {
            System.out.println("Something went wrong in main class: ");
            e.printStackTrace();
        }
    }
}
```

```java
package com.finalproject;

import java.io.*;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.mapreduce.Mapper;

public class MapperClass extends Mapper<LongWritable, Text, Text,
CountAverageTuple> {

    private CountAverageTuple outCountAverage = new
CountAverageTuple();
    private Text id = new Text();

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

        try {

            String input[] = value.toString().split("\\t");
            String productId = input[3].trim();

            if (!productId.isEmpty()) {
                id.set((productId));
                outCountAverage.setCount(Long.valueOf(1));

outCountAverage.setAverage(Float.valueOf(input[7].trim()));
                context.write(id, outCountAverage);
            }

        } catch (Exception e) {
            System.out.println("Something went wrong in Mapper Task:
");
            e.printStackTrace();
        }

    }
}
```

```java
package com.finalproject;

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

public class ReducerClass extends Reducer<Text, CountAverageTuple,
Text, CountAverageTuple> {

    private CountAverageTuple result = new CountAverageTuple();

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

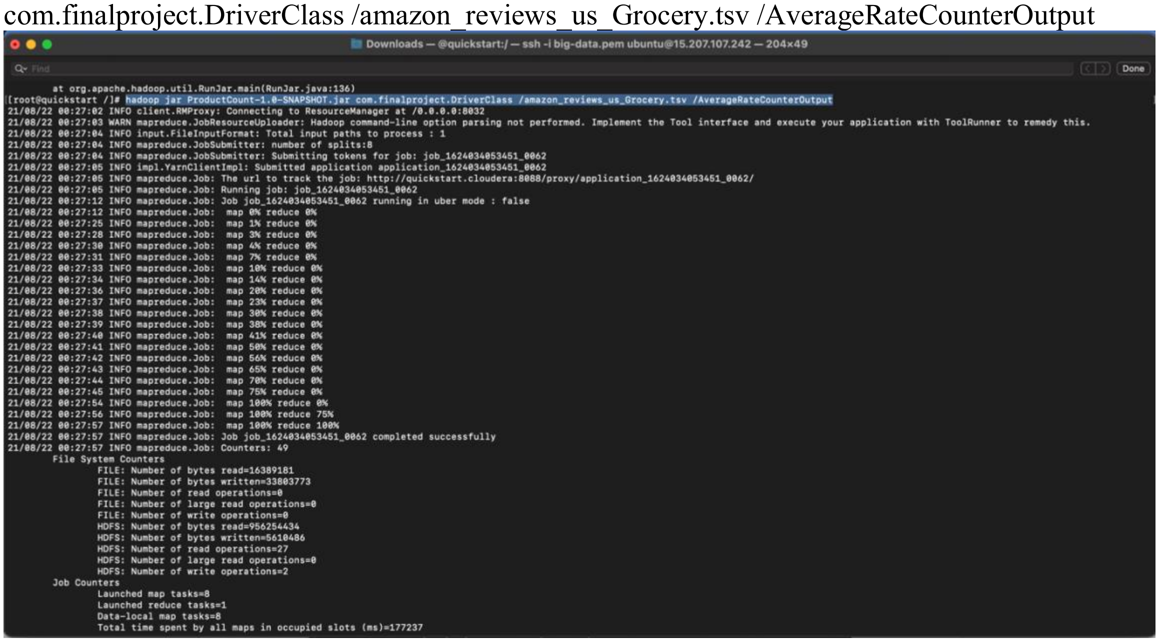
```
        try {
            long count = 0;
            float sum = 0;

            for (CountAverageTuple val: value) {
                count += val.getCount();
                sum += val.getCount() * val.getAverage();
            }

            result.setCount(count);
            result.setAverage(sum/count);
            context.write(key, result);

        } catch (Exception e) {
            System.out.println("Something went wrong in Reducer Task:
");
            e.printStackTrace();
        }
    }

}
```

```
package com.finalproject;

import org.apache.hadoop.io.Writable;

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

public class CountAverageTuple implements Writable {

    private Long count;
    private Float average;

    public CountAverageTuple(){

    }

    public CountAverageTuple(Long count, Float average) {
        this.count = count;
        this.average = average;
    }

    public void write(DataOutput d) throws IOException {
        d.writeLong(count);
        d.writeFloat(average);
    }

    public void readFields(DataInput di) throws IOException {
        count = di.readLong();
        average = di.readFloat();
    }

    public Long getCount() {
```

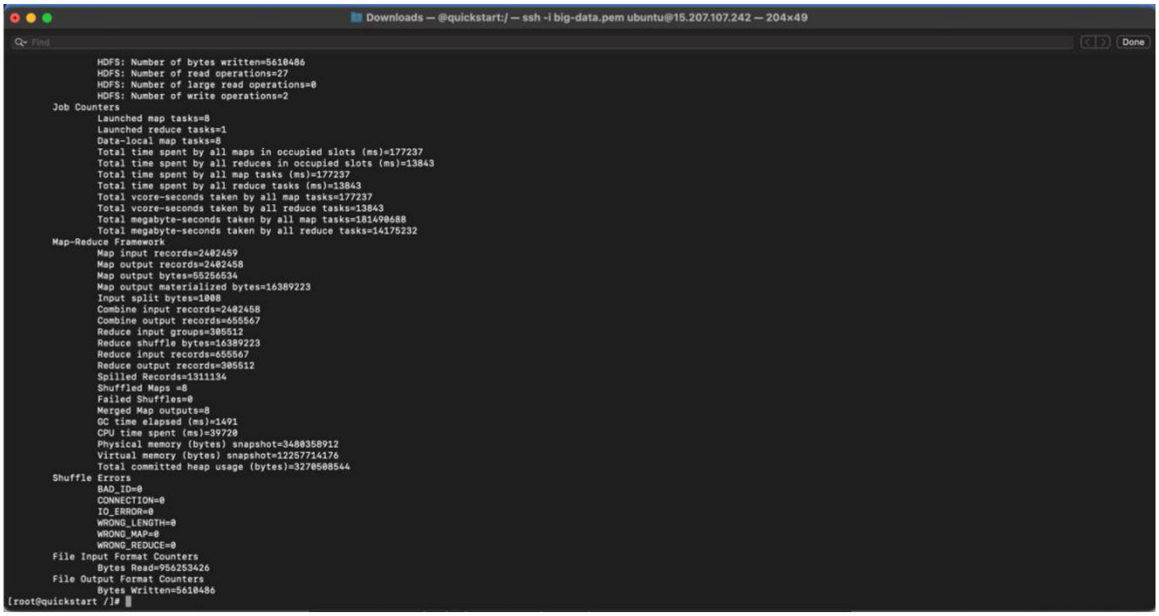
```java
        return count;
    }

    public void setCount(Long count) {
        this.count = count;
    }

    public Float getAverage() {
        return average;
    }

    public void setAverage(Float average) {
        this.average = average;
    }

    @Override
    public String toString() {
        return (new
StringBuilder().append(count).append("\t").append(average).toString());
    }
}
```

Output:

[root@quickstart /]# hadoop jar ProductCount-1.0-SNAPSHOT.jar com.finalproject.DriverClass /amazon_reviews_us_Grocery.tsv /AverageRateCounterOutput



```
Downloads — @quickstart:/ — ssh -i big-data.pem ubuntu@15.207.107.242 — 204x49
        at org.apache.hadoop.util.RunJar.main(RunJar.java:136)
[root@quickstart /]# hadoop jar ProductCount-1.0-SNAPSHOT.jar com.finalproject.DriverClass /amazon_reviews_us_Grocery.tsv /AverageRateCounterOutput
21/08/22 00:27:02 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
21/08/22 00:27:03 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
21/08/22 00:27:04 INFO input.FileInputFormat: Total input paths to process : 1
21/08/22 00:27:04 INFO mapreduce.JobSubmitter: number of splits:8
21/08/22 00:27:04 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1624034053451_0062
21/08/22 00:27:05 INFO impl.YarnClientImpl: Submitted application application_1624034053451_0062
21/08/22 00:27:05 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1624034053451_0062/
21/08/22 00:27:05 INFO mapreduce.Job: Running job: job_1624034053451_0062
21/08/22 00:27:12 INFO mapreduce.Job: Job job_1624034053451_0062 running in uber mode : false
21/08/22 00:27:12 INFO mapreduce.Job:  map 0% reduce 0%
21/08/22 00:27:25 INFO mapreduce.Job:  map 1% reduce 0%
21/08/22 00:27:28 INFO mapreduce.Job:  map 3% reduce 0%
21/08/22 00:27:30 INFO mapreduce.Job:  map 4% reduce 0%
21/08/22 00:27:31 INFO mapreduce.Job:  map 7% reduce 0%
21/08/22 00:27:33 INFO mapreduce.Job:  map 10% reduce 0%
21/08/22 00:27:34 INFO mapreduce.Job:  map 14% reduce 0%
21/08/22 00:27:36 INFO mapreduce.Job:  map 20% reduce 0%
21/08/22 00:27:37 INFO mapreduce.Job:  map 23% reduce 0%
21/08/22 00:27:38 INFO mapreduce.Job:  map 30% reduce 0%
21/08/22 00:27:39 INFO mapreduce.Job:  map 38% reduce 0%
21/08/22 00:27:40 INFO mapreduce.Job:  map 41% reduce 0%
21/08/22 00:27:41 INFO mapreduce.Job:  map 50% reduce 0%
21/08/22 00:27:42 INFO mapreduce.Job:  map 56% reduce 0%
21/08/22 00:27:43 INFO mapreduce.Job:  map 65% reduce 0%
21/08/22 00:27:44 INFO mapreduce.Job:  map 70% reduce 0%
21/08/22 00:27:45 INFO mapreduce.Job:  map 75% reduce 0%
21/08/22 00:27:54 INFO mapreduce.Job:  map 100% reduce 0%
21/08/22 00:27:56 INFO mapreduce.Job:  map 100% reduce 75%
21/08/22 00:27:57 INFO mapreduce.Job:  map 100% reduce 100%
21/08/22 00:27:57 INFO mapreduce.Job: Job job_1624034053451_0062 completed successfully
21/08/22 00:27:57 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=16389181
                FILE: Number of bytes written=33803773
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=956254434
                HDFS: Number of bytes written=5610486
                HDFS: Number of read operations=27
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=8
                Launched reduce tasks=1
                Data-local map tasks=8
                Total time spent by all maps in occupied slots (ms)=177237
```



Case 2: Find user who has reviewed the product

Inverted Index - Inverted index pattern is used to generate an index from a data set to allow for faster searches or data enrichment capabilities. It is often convenient to index large data sets on keywords, so that searches can trace terms back to records that contain specific values. While building an inverted index does require extra processing up front, taking the time to do so can greatly reduce the amount of time it takes to find something.

```
package com.finalproject;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
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.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

import java.io.IOException;

//This pattern is used to find each user who has reviewed the product

public class DriverClass {

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

```java
        Configuration conf = new Configuration();
        FileSystem fs = FileSystem.get(conf);

        try {
            Job invertedIndexJob = Job.getInstance(conf, "Inverted
Index");
            invertedIndexJob.setJarByClass(DriverClass.class);

            invertedIndexJob.setMapperClass(Map.class);
            invertedIndexJob.setReducerClass(Reduce.class);

invertedIndexJob.setInputFormatClass(TextInputFormat.class);

invertedIndexJob.setOutputFormatClass(TextOutputFormat.class);

            invertedIndexJob.setMapOutputKeyClass(Text.class);
            invertedIndexJob.setMapOutputValueClass(Text.class);
            invertedIndexJob.setOutputKeyClass(Text.class);
            invertedIndexJob.setOutputValueClass(Text.class);

            FileInputFormat.addInputPath(invertedIndexJob, new
Path(args[0]));
            FileOutputFormat.setOutputPath(invertedIndexJob, new
Path(args[1]));
            if (fs.exists(new Path(args[1]))) {
                fs.delete(new Path(args[1]), true);
            }

            invertedIndexJob.waitForCompletion(true);

        } catch (Exception e) {
            System.out.println("Something went wrong in main class: ");
            e.printStackTrace();
        }
    }
}
```

```java
package com.finalproject;

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

import java.io.IOException;

public class Map extends Mapper<LongWritable, Text, Text, Text> {

    Text prod_cat = new Text();

    private Text productId = new Text();
    private Text userId = new Text();

    @Override
    protected void map(LongWritable key, Text value, Context context)
```

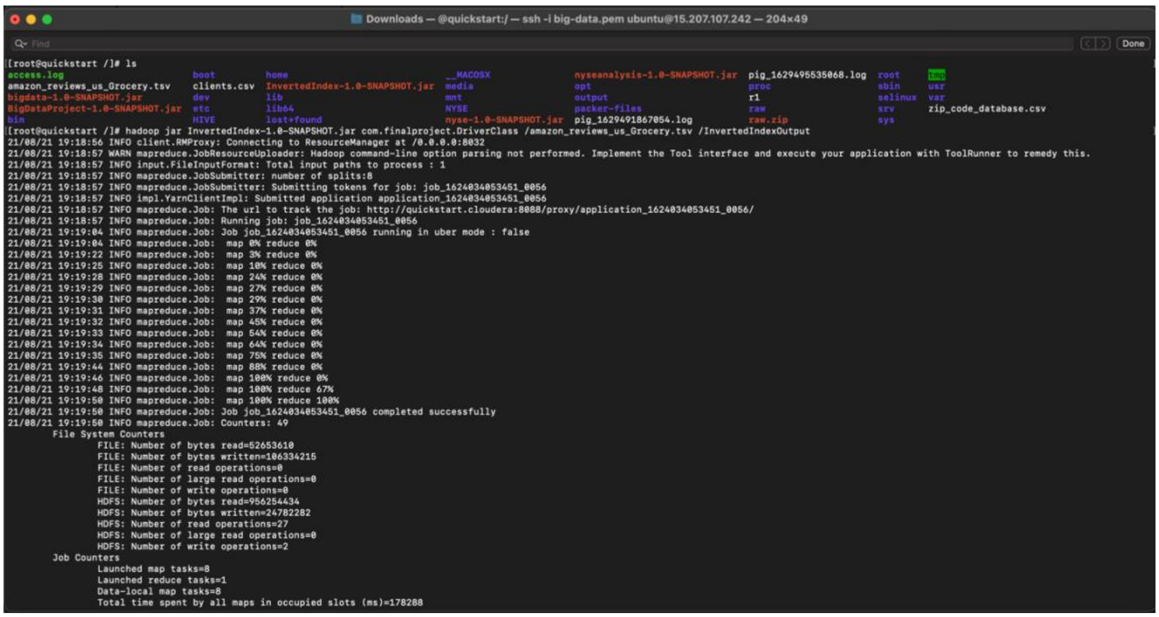
```java
throws IOException, InterruptedException {

        if(key.get()==0){
            return;
        }
        try{
            String[] tokens = value.toString().split("\\t");
            userId.set(tokens[1]);
            productId.set(tokens[3]);
            context.write(productId, userId);

        } catch(Exception  e){
            System.out.println("Something went wrong in Mapper Task:
");
            e.printStackTrace();
        }
    }
}
```

```java
package com.finalproject;

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

public class Reduce extends Reducer<Text, Text, Text, Text> {

    private Text result = new Text();

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

        try {
            StringBuilder sb = new StringBuilder();
            boolean first = true;

            for (Text id : values) {
                if (first) {
                    first = false;
                } else {
                    sb.append(" ");
                }
                sb.append(id.toString());
            }

            result.set(sb.toString());
            context.write(key, result);

        } catch (Exception e) {
            System.out.println("Something went wrong in Reducer Task:
");
            e.printStackTrace();
        }
```

```
	}
}
```

Output:

[root@quickstart /]# hadoop jar InvertedIndex-1.0-SNAPSHOT.jar com.finalproject.DriverClass /amazon_reviews_us_Grocery.tsv /InvertedIndexOutput



```
Downloads — @quickstart:/ — ssh -i big-data.pem ubuntu@15.207.107.242 — 204×49
[root@quickstart /]# ls
access.log                     boot         home                           __MACOSX                   nyseanalysis-1.0-SNAPSHOT.jar  pig_1629495535068.log  root     tmp
amazon_reviews_us_Grocery.tsv  clients.csv  InvertedIndex-1.0-SNAPSHOT.jar  media                      opt                            proc                   sbin     usr
bigdata-1.0-SNAPSHOT.jar       dev          lib                            mnt                        output                         r1                     selinux  var
BigDataProject-1.0-SNAPSHOT.jar etc         lib64                          NYSE                       packer-files                   raw                    srv      zip_code_database.csv
bin                            HIVE         lost+found                     nyse-1.0-SNAPSHOT.jar      pig_1629491867054.log          raw.zip                sys
[root@quickstart /]# hadoop jar InvertedIndex-1.0-SNAPSHOT.jar com.finalproject.DriverClass /amazon_reviews_us_Grocery.tsv /InvertedIndexOutput
21/08/21 19:18:56 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
21/08/21 19:18:57 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
21/08/21 19:18:57 INFO input.FileInputFormat: Total input paths to process : 1
21/08/21 19:18:57 INFO mapreduce.JobSubmitter: number of splits:8
21/08/21 19:18:57 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1624034053451_0056
21/08/21 19:18:57 INFO impl.YarnClientImpl: Submitted application application_1624034053451_0056
21/08/21 19:18:57 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1624034053451_0056/
21/08/21 19:18:57 INFO mapreduce.Job: Running job: job_1624034053451_0056
21/08/21 19:19:04 INFO mapreduce.Job: Job job_1624034053451_0056 running in uber mode : false
21/08/21 19:19:04 INFO mapreduce.Job:  map 0% reduce 0%
21/08/21 19:19:22 INFO mapreduce.Job:  map 3% reduce 0%
21/08/21 19:19:25 INFO mapreduce.Job:  map 10% reduce 0%
21/08/21 19:19:28 INFO mapreduce.Job:  map 24% reduce 0%
21/08/21 19:19:29 INFO mapreduce.Job:  map 27% reduce 0%
21/08/21 19:19:30 INFO mapreduce.Job:  map 29% reduce 0%
21/08/21 19:19:31 INFO mapreduce.Job:  map 37% reduce 0%
21/08/21 19:19:32 INFO mapreduce.Job:  map 45% reduce 0%
21/08/21 19:19:33 INFO mapreduce.Job:  map 54% reduce 0%
21/08/21 19:19:34 INFO mapreduce.Job:  map 64% reduce 0%
21/08/21 19:19:35 INFO mapreduce.Job:  map 75% reduce 0%
21/08/21 19:19:44 INFO mapreduce.Job:  map 88% reduce 0%
21/08/21 19:19:46 INFO mapreduce.Job:  map 100% reduce 0%
21/08/21 19:19:48 INFO mapreduce.Job:  map 100% reduce 67%
21/08/21 19:19:50 INFO mapreduce.Job:  map 100% reduce 100%
21/08/21 19:19:50 INFO mapreduce.Job: Job job_1624034053451_0056 completed successfully
21/08/21 19:19:50 INFO mapreduce.Job: Counters: 49
	File System Counters
		FILE: Number of bytes read=52653610
		FILE: Number of bytes written=106334215
		FILE: Number of read operations=0
		FILE: Number of large read operations=0
		FILE: Number of write operations=0
		HDFS: Number of bytes read=956254434
		HDFS: Number of bytes written=24782282
		HDFS: Number of read operations=27
		HDFS: Number of large read operations=0
		HDFS: Number of write operations=2
	Job Counters
		Launched map tasks=8
		Launched reduce tasks=1
		Data-local map tasks=8
		Total time spent by all maps in occupied slots (ms)=178288
```

Case 3: Find the year in which the product was reviewed

Partitioner - A partitioner works like a condition in processing an input dataset. The partition phase takes place after the Map phase and before the Reduce phase.

The number of partitioners is equal to the number of reducers. That means a partitioner will divide the data according to the number of reducers. Therefore, the data passed from a single partitioner is processed by a single Reducer.

```
package com.finalproject;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
```

```java
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import java.io.IOException;

// Find all the records partitioned by the year in which the product
was reviewed

public class DriverClass {

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

        Configuration conf = new Configuration();
        FileSystem fs = FileSystem.get(conf);
        Job job = Job.getInstance(conf, "Partitioning");

        job.setJarByClass(DriverClass.class);

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

        //Custom Partitioner:
        job.setPartitionerClass(YearPartitionPartitioner.class);

        job.setReducerClass(ReducerClass.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(NullWritable.class);
        job.setNumReduceTasks(14);

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

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

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

public class MapperClass extends Mapper<LongWritable, Text, Text, Text>
{

    private Text inputRec = new Text();
    private Text year = new Text();

    protected void map(LongWritable key, Text value, Mapper.Context
context) throws IOException, InterruptedException{

        if(key.get()==0){
            return;
        }

        String[] line = value.toString().split("\\t");
```

```java
        String[] yearPart = line[14].split("-");
        String yearVal = yearPart[2].trim();

        year.set(yearVal);
        inputRec.set(value);

        context.write(year, inputRec);
    }
}
```

```java
package com.finalproject;

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

public class ReducerClass extends Reducer<Text, Text, Text,
NullWritable> {


    protected void reduce(Text key, Iterable<Text> values,
Reducer.Context context) throws IOException, InterruptedException{
        for(Text t: values){

            context.write(t, NullWritable.get());
        }
    }
}
```

```java
package com.finalproject;

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

public class YearPartitionPartitioner extends Partitioner<Text, Text> {
    @Override
    public int getPartition(Text key, Text value, int numPartitions){
        int n=1;
        if(numPartitions==0){
            return 0;
        }
        else if(key.equals(("99"))){
            return n % numPartitions;
        }
        else if(key.equals(new Text("00"))){
            return 2 % numPartitions;
        }
        else if(key.equals(new Text("01"))){
            return 3 % numPartitions ;
        }
        else if(key.equals(new Text("02"))){
```

```
            return 4 % numPartitions;
        }
        else if(key.equals(new Text("03"))){
            return 5 % numPartitions;
        }
        else if(key.equals(new Text("04"))){
            return 6 % numPartitions;
        }
        else if(key.equals(new Text("05"))){
            return 7 % numPartitions;
        }
        else if(key.equals(new Text("06"))){
            return 8 % numPartitions;
        }
        else if(key.equals(new Text("07"))){
            return 9 % numPartitions;
        }
        else if(key.equals(new Text("08"))){
            return 10 % numPartitions;
        }
        else if (key.equals(new Text("09"))){
            return 11 % numPartitions;
        }
        else if (key.equals(new Text("10"))){
            return 12 % numPartitions;
        }
        else if (key.equals(new Text("11"))){
            return 13 % numPartitions;
        }
        else
        {
            return 14 % numPartitions;
        }
    }
}
```

Output:

[root@quickstart /]# hadoop jar YearPartitioner-1.0-SNAPSHOT.jar com.finalproject.DriverClass /amazon_reviews_us_Grocery.tsv /YearPartitionerOutput

Downloads — @quickstart:/ — ssh -i big-data.pem ubuntu@15.207.107.242 — 204×49

```
exit
ubuntu@big_data:~$ docker cp YearPartitioner-1.0-SNAPSHOT.jar 4db8b228c18c:/YearPartitioner-1.0-SNAPSHOT.jar
ubuntu@big_data:~$ docker exec -it 4db8b228c18c /bin/bash
[root@quickstart /]# hadoop jar YearPartitioner-1.0-SNAPSHOT.jar com.finalproject.DriverClass /amazon_reviews_us_Grocery.tsv /YearPartitionerOutput
21/08/21 21:28:31 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
21/08/21 21:28:31 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
21/08/21 21:28:31 INFO input.FileInputFormat: Total input paths to process : 1
21/08/21 21:28:31 INFO mapreduce.JobSubmitter: number of splits:8
21/08/21 21:28:31 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1624034053451_0060
21/08/21 21:28:32 INFO impl.YarnClientImpl: Submitted application application_1624034053451_0060
21/08/21 21:28:32 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1624034053451_0060/
21/08/21 21:28:32 INFO mapreduce.Job: Running job: job_1624034053451_0060
21/08/21 21:28:38 INFO mapreduce.Job: Job job_1624034053451_0060 running in uber mode : false
21/08/21 21:28:38 INFO mapreduce.Job:  map 0% reduce 0%
21/08/21 21:28:51 INFO mapreduce.Job:  map 1% reduce 0%
21/08/21 21:28:54 INFO mapreduce.Job:  map 2% reduce 0%
21/08/21 21:28:56 INFO mapreduce.Job:  map 3% reduce 0%
21/08/21 21:28:57 INFO mapreduce.Job:  map 5% reduce 0%
21/08/21 21:29:00 INFO mapreduce.Job:  map 9% reduce 0%
21/08/21 21:29:02 INFO mapreduce.Job:  map 10% reduce 0%
21/08/21 21:29:03 INFO mapreduce.Job:  map 16% reduce 0%
21/08/21 21:29:04 INFO mapreduce.Job:  map 17% reduce 0%
21/08/21 21:29:05 INFO mapreduce.Job:  map 20% reduce 0%
21/08/21 21:29:06 INFO mapreduce.Job:  map 24% reduce 0%
21/08/21 21:29:07 INFO mapreduce.Job:  map 28% reduce 0%
21/08/21 21:29:08 INFO mapreduce.Job:  map 33% reduce 0%
21/08/21 21:29:09 INFO mapreduce.Job:  map 36% reduce 0%
21/08/21 21:29:10 INFO mapreduce.Job:  map 39% reduce 0%
21/08/21 21:29:11 INFO mapreduce.Job:  map 44% reduce 0%
21/08/21 21:29:12 INFO mapreduce.Job:  map 46% reduce 0%
21/08/21 21:29:13 INFO mapreduce.Job:  map 50% reduce 0%
21/08/21 21:29:14 INFO mapreduce.Job:  map 52% reduce 0%
21/08/21 21:29:15 INFO mapreduce.Job:  map 53% reduce 0%
21/08/21 21:29:16 INFO mapreduce.Job:  map 55% reduce 0%
21/08/21 21:29:17 INFO mapreduce.Job:  map 56% reduce 0%
21/08/21 21:29:18 INFO mapreduce.Job:  map 58% reduce 0%
21/08/21 21:29:19 INFO mapreduce.Job:  map 59% reduce 0%
21/08/21 21:29:20 INFO mapreduce.Job:  map 63% reduce 0%
21/08/21 21:29:21 INFO mapreduce.Job:  map 71% reduce 0%
21/08/21 21:29:22 INFO mapreduce.Job:  map 75% reduce 0%
21/08/21 21:29:39 INFO mapreduce.Job:  map 88% reduce 0%
21/08/21 21:29:40 INFO mapreduce.Job:  map 89% reduce 2%
21/08/21 21:29:43 INFO mapreduce.Job:  map 96% reduce 4%
21/08/21 21:29:44 INFO mapreduce.Job:  map 96% reduce 6%
21/08/21 21:29:45 INFO mapreduce.Job:  map 100% reduce 8%
21/08/21 21:29:46 INFO mapreduce.Job:  map 100% reduce 18%
21/08/21 21:29:48 INFO mapreduce.Job:  map 100% reduce 23%
21/08/21 21:29:49 INFO mapreduce.Job:  map 100% reduce 25%
21/08/21 21:29:51 INFO mapreduce.Job:  map 100% reduce 32%
```

Downloads — @quickstart:/ — ssh -i big-data.pem ubuntu@15.207.107.242 — 204×49

```
21/08/21 21:30:13 INFO mapreduce.Job:  map 100% reduce 64%
21/08/21 21:30:14 INFO mapreduce.Job:  map 100% reduce 70%
21/08/21 21:30:15 INFO mapreduce.Job:  map 100% reduce 71%
21/08/21 21:30:27 INFO mapreduce.Job:  map 100% reduce 79%
21/08/21 21:30:29 INFO mapreduce.Job:  map 100% reduce 86%
21/08/21 21:30:30 INFO mapreduce.Job:  map 100% reduce 100%
21/08/21 21:30:30 INFO mapreduce.Job: Job job_1624034053451_0060 completed successfully
21/08/21 21:30:30 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=1935936028
                FILE: Number of bytes written=2914248077
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=956254434
                HDFS: Number of bytes written=963431936
                HDFS: Number of read operations=66
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=28
        Job Counters
                Launched map tasks=8
                Launched reduce tasks=14
                Data-local map tasks=8
                Total time spent by all maps in occupied slots (ms)=273518
                Total time spent by all reduces in occupied slots (ms)=307312
                Total time spent by all map tasks (ms)=273518
                Total time spent by all reduce tasks (ms)=307312
                Total vcore-seconds taken by all map tasks=273518
                Total vcore-seconds taken by all reduce tasks=307312
                Total megabyte-seconds taken by all map tasks=280082432
                Total megabyte-seconds taken by all reduce tasks=314687488
        Map-Reduce Framework
                Map input records=2402459
                Map output records=2402458
                Map output bytes=967422778
                Map output materialized bytes=976236591
                Input split bytes=1008
                Combine input records=0
                Combine output records=0
                Reduce input groups=31
                Reduce shuffle bytes=976236591
                Reduce input records=2402458
                Reduce output records=2402458
                Spilled Records=7176810
                Shuffled Maps =112
                Failed Shuffles=0
                Merged Map outputs=112
                GC time elapsed (ms)=4737
                CPU time spent (ms)=96350
```

MongoDB MapReduce

Case: To calculate the total number of verified and non - verified purchases of all products.

Steps:

Import data to mongo

```
mongoimport --db reviewdata --collection reviewcoll --type tsv --headerline --file
'/Users/apoorvamishra/Downloads/amazon_reviews_us_Grocery.tsv'
```

Map Function

```
map1 = function() {
emit(this.verified_purchase, this.product_id);
}
```

Reduce Function

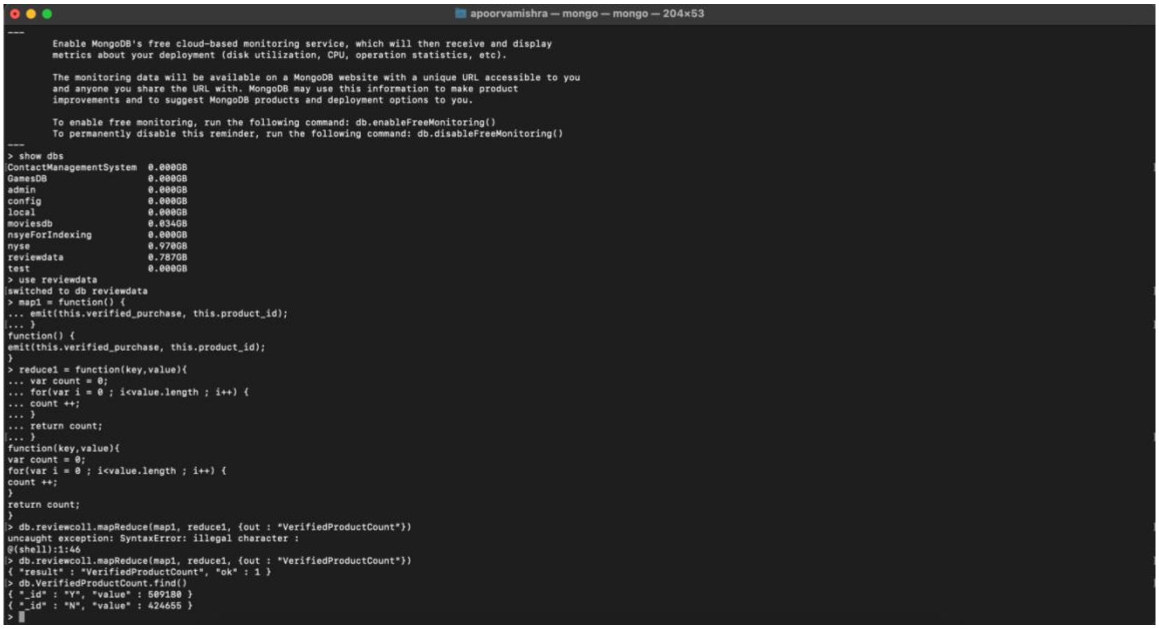
```
reduce1 = function(key,value){
var count = 0;
for(var i = 0 ; i<value.length ; i++) {
count ++;
}
return count;
}
```

MapReduce in MongoDB

```
db.reviewcoll.mapReduce(map1, reduce1, {out : “VerifiedProductCount”})
```

Output:

```
db.VerifiedProductCount.find()
```



```
apoorvamishra — mongo — mongo — 204×53
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> show dbs
ContactManagementSystem  0.000GB
GamesDB                  0.000GB
admin                    0.000GB
config                   0.000GB
local                    0.000GB
moviesdb                 0.034GB
nsyeForIndexing          0.000GB
nyse                     0.970GB
reviewdata               0.787GB
test                     0.000GB
> use reviewdata
switched to db reviewdata
> map1 = function() {
... emit(this.verified_purchase, this.product_id);
... }
function() {
emit(this.verified_purchase, this.product_id);
}
> reduce1 = function(key,value){
... var count = 0;
... for(var i = 0 ; i<value.length ; i++) {
... count ++;
... }
... return count;
... }
function(key,value){
var count = 0;
for(var i = 0 ; i<value.length ; i++) {
count ++;
}
return count;
}
> db.reviewcoll.mapReduce(map1, reduce1, {out : "VerifiedProductCount"})
uncaught exception: SyntaxError: illegal character :
@(shell):1:46
> db.reviewcoll.mapReduce(map1, reduce1, {out : "VerifiedProductCount"})
{ "result" : "VerifiedProductCount", "ok" : 1 }
> db.VerifiedProductCount.find()
{ "_id" : "Y", "value" : 509180 }
{ "_id" : "N", "value" : 424655 }
>
```

HIVE

Apache Hive is a data warehouse system built on top of Hadoop and is used for analyzing structured and semi-structured data. It provides a mechanism to project structure onto the data and perform queries written in HQL (Hive Query Language) that are similar to SQL statements. Internally, these queries or HQL gets converted to map reduce jobs by the Hive compiler.

Case: Create Table and upload customer review data from a tsv file onto HIVE.

Create Table customerreviewdata in HIVE

```
CREATE TABLE IF NOT EXISTS customerreviewdata (marketplace String,
customer_id String, review_id String, product_id String, product_parent String,
product_title String, product_category String, star_rating String, helpful_votes
String, total_votes String, vine String, verified_purchase String, review_headline
String, review_body String, review_date String) ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE tblproperties("skip.header.line.count" = "1");
```

Load data from local into table

```
Load data local inpath '/amazon_reviews_us_Grocery.tsv' into table reviewdata;
```

Query: Find the highest and lowest rated verified products.

```
INSERT OVERWRITE LOCAL DIRECTORY 'GroceryHiveout.tsv' ROW FORMAT
DELIMITED FIELDS TERMINATED BY ',' SELECT product_id, Max(star_rating), Min
(star_rating), SUM(helpful_votes) from reviewdata where verified_purchase = 'Y'
GROUP BY product_id ;
```

Output:

```
Downloads — @quickstart:/ — ssh -i big-data.pem ubuntu@15.207.107.242 — 204x49
ubuntu@big_data:~$ docker exec -it 4db8b228c18c /bin/bash
[root@quickstart /]# hive
2021-08-22 04:33:29,188 WARN  [main] mapreduce.TableMapReduceUtil: The hbase-prefix-tree module jar containing PrefixTreeCodec is not present.  Continuing without it.

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 IF NOT EXISTS customerreviewdata (marketplace String, customer_id String, review_id String, product_id String, product_parent String, product_title String, product_category String, star
_rating String, helpful_votes String, total_votes String, vine String, verified_purchase String, review_headline String, review_body String, review_date String) ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY '\t'
    > LINES TERMINATED BY '\n'
    > STORED AS TEXTFILE tblproperties("skip.header.line.count" = "1");
OK
Time taken: 0.798 seconds
hive> Load data local inpath '/amazon_reviews_us_Grocery.tsv' into table reviewdata;
Loading data to table default.reviewdata
Table default.reviewdata stats: [numFiles=2, totalSize=1912449508]
OK
Time taken: 11.49 seconds
hive> INSERT OVERWRITE LOCAL DIRECTORY 'GroceryHiveout.tsv' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' SELECT product_id, product_title, Max(star_rating), Min (star_rating), SUM(helpful_votes) from rev
iewdata where verified_purchase = 'Y'  GROUP BY product_id ;
FAILED: SemanticException [Error 10025]: Line 1:119 Expression not in GROUP BY key 'product_title'
hive> INSERT OVERWRITE LOCAL DIRECTORY 'GroceryHiveout.tsv' ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' SELECT product_id, Max(star_rating), Min (star_rating), SUM(helpful_votes) from reviewdata where v
erified_purchase = 'Y'  GROUP BY product_id ;
Query ID = root_20210822043333_919e89e2-6651-46b4-8b7a-2a24675d4fc1
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 8
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_1624034053451_0063, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1624034053451_0063/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1624034053451_0063
Hadoop job information for Stage-1: number of mappers: 8; number of reducers: 8
2021-08-22 04:35:17,802 Stage-1 map = 0%,  reduce = 0%
2021-08-22 04:35:44,863 Stage-1 map = 4%,  reduce = 0%, Cumulative CPU 11.86 sec
2021-08-22 04:35:49,283 Stage-1 map = 9%,  reduce = 0%, Cumulative CPU 23.61 sec
2021-08-22 04:35:51,521 Stage-1 map = 17%,  reduce = 0%, Cumulative CPU 28.53 sec
2021-08-22 04:35:53,713 Stage-1 map = 29%,  reduce = 0%, Cumulative CPU 33.16 sec
2021-08-22 04:35:54,834 Stage-1 map = 33%,  reduce = 0%, Cumulative CPU 34.07 sec
2021-08-22 04:35:55,946 Stage-1 map = 46%,  reduce = 0%, Cumulative CPU 36.8 sec
2021-08-22 04:35:58,141 Stage-1 map = 50%,  reduce = 0%, Cumulative CPU 40.13 sec
2021-08-22 04:35:59,218 Stage-1 map = 58%,  reduce = 0%, Cumulative CPU 42.27 sec
2021-08-22 04:36:00,376 Stage-1 map = 67%,  reduce = 0%, Cumulative CPU 43.79 sec
2021-08-22 04:36:01,476 Stage-1 map = 71%,  reduce = 0%, Cumulative CPU 45.18 sec
2021-08-22 04:36:03,731 Stage-1 map = 75%,  reduce = 0%, Cumulative CPU 45.52 sec
2021-08-22 04:36:19,546 Stage-1 map = 88%,  reduce = 0%, Cumulative CPU 53.31 sec
```

Query: Find the total number of products

```
Downloads — @quickstart:/ — ssh -i big-data.pem ubuntu@15.207.107.242 — 204x49
OK
Time taken: 0.061 seconds
hive> select count (*) from customerreviewdata;
Query ID = root_20210822043333_919e89e2-6651-46b4-8b7a-2a24675d4fc1
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_1624034053451_0065, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1624034053451_0065/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1624034053451_0065
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-08-22 04:46:49,051 Stage-1 map = 0%,  reduce = 0%
2021-08-22 04:46:55,285 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.4 sec
2021-08-22 04:47:01,545 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.84 sec
MapReduce Total cumulative CPU time: 2 seconds 840 msec
Ended Job = job_1624034053451_0065
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.84 sec   HDFS Read: 8069 HDFS Write: 2 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 840 msec
OK
0
Time taken: 20.377 seconds, Fetched: 1 row(s)
```

Query: Find the total number of products in each product category

```
hive> select product_category, SUM(product_id) from customerreviewdata GROUP BY product_category;
Query ID = root_20210822043333_919e89e2-6651-46b4-8b7a-2a24675d4fc1
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_1624034053451_0066, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1624034053451_0066/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1624034053451_0066
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2021-08-22 04:49:06,738 Stage-1 map = 0%,  reduce = 0%
2021-08-22 04:49:12,983 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.39 sec
2021-08-22 04:49:19,195 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.61 sec
MapReduce Total cumulative CPU time: 2 seconds 610 msec
Ended Job = job_1624034053451_0066
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 2.61 sec   HDFS Read: 8674 HDFS Write: 0 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 610 msec
OK
Time taken: 20.206 seconds
hive>
```

Apache PIG

Apache Pig is a platform, used to analyze large data sets representing them as data flows. It is designed to provide an abstraction over MapReduce, reducing the complexities of writing a MapReduce program. We can perform data manipulation operations very easily in Hadoop using Apache Pig.

Case: Find the daily review count of all products

data = LOAD ‘/amazon_reviews_us_Grocery.tsv’ AS (marketplace, customer_id, review_id, product_id, product_parent, product_title, product_category, star_rating, helpful_votes, total_votes, vine, verified_purchase, review_headline, review_body, review_date);

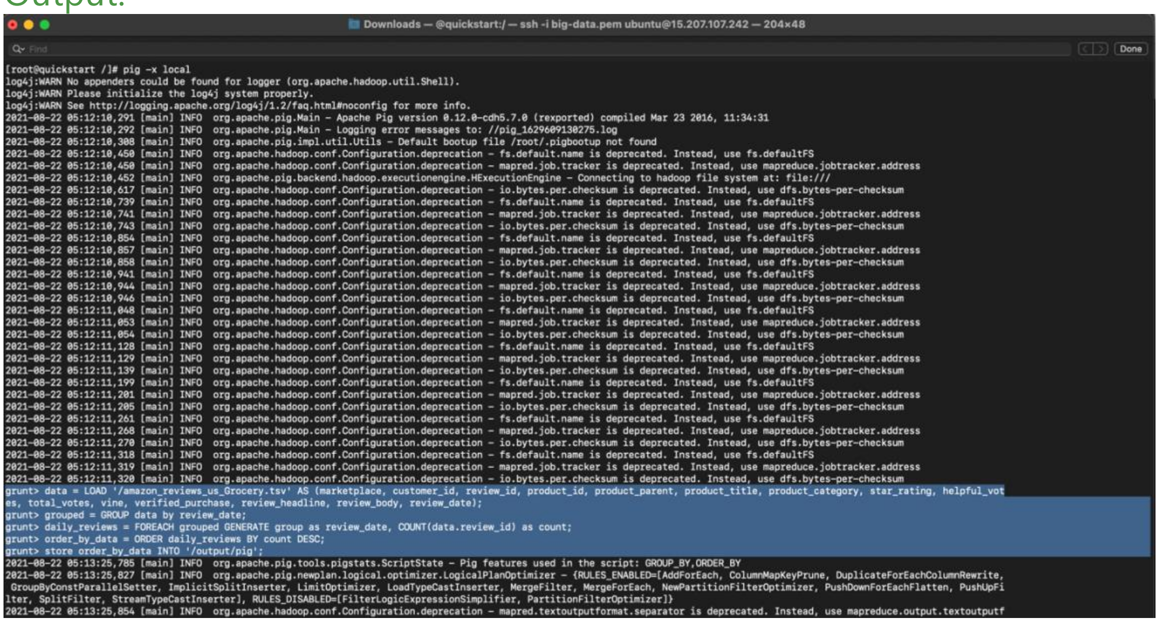
grouped = GROUP data by review_date;

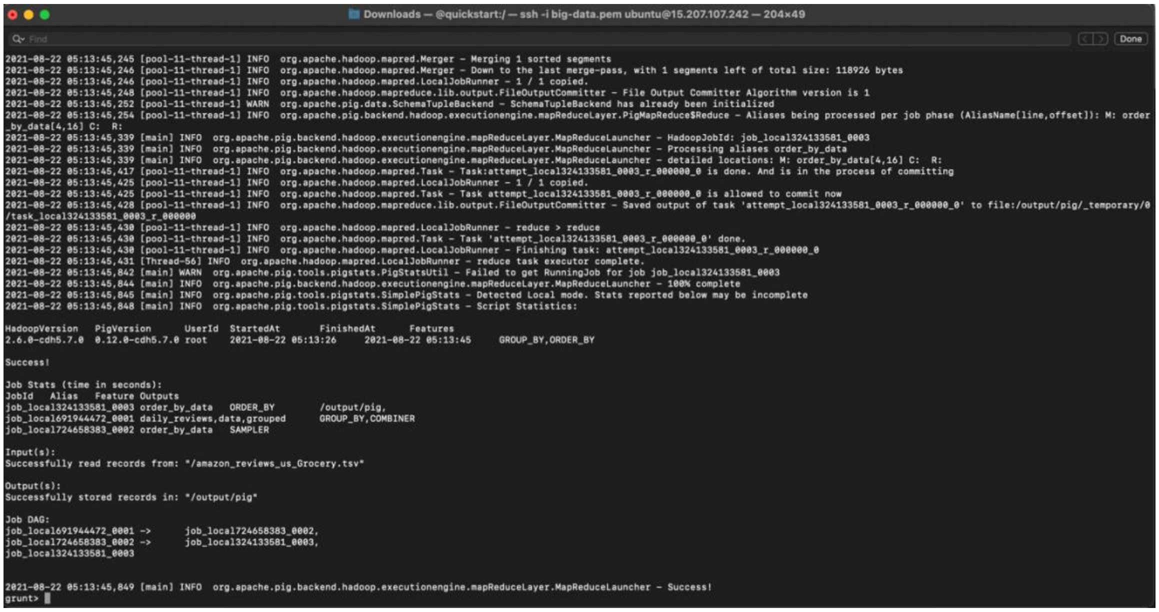
daily_reviews = FOREACH grouped GENERATE group as review_date, COUNT(data.review_id) as count;

order_by_data = ORDER daily_reviews BY count DESC;

store order_by_data INTO '/output/pig';

Output:





Case: Find the total number of products for each rating

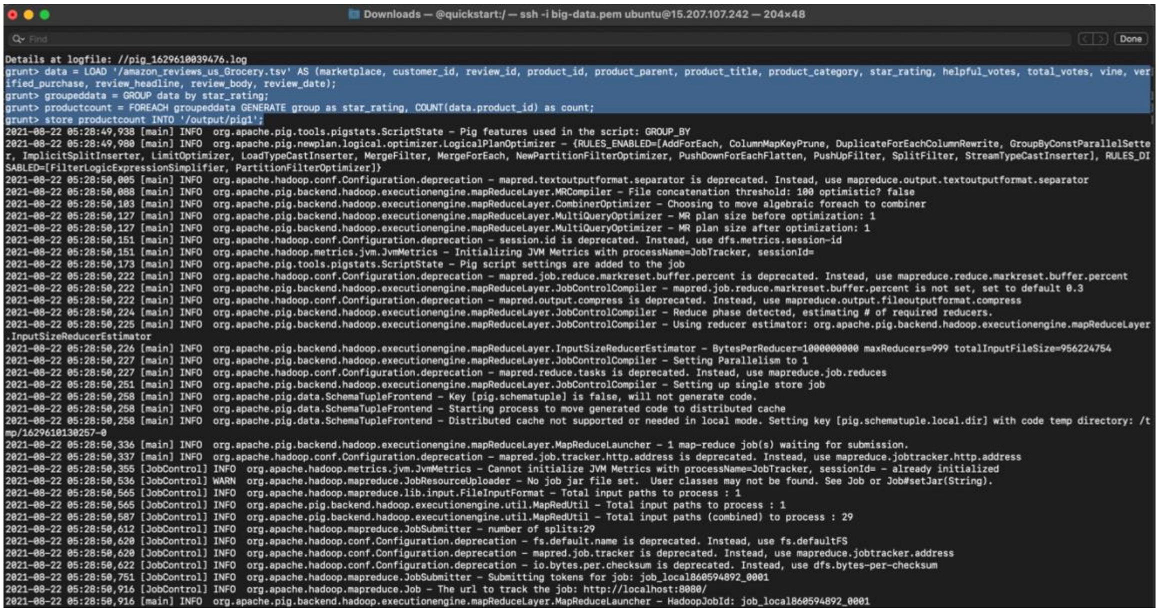
data = LOAD ‘/amazon_reviews_us_Grocery.tsv’ AS (marketplace, customer_id, review_id, product_id, product_parent, product_title, product_category, star_rating, helpful_votes, total_votes, vine, verified_purchase, review_headline, review_body, review_date);

groupeddata = GROUP data by star_rating;

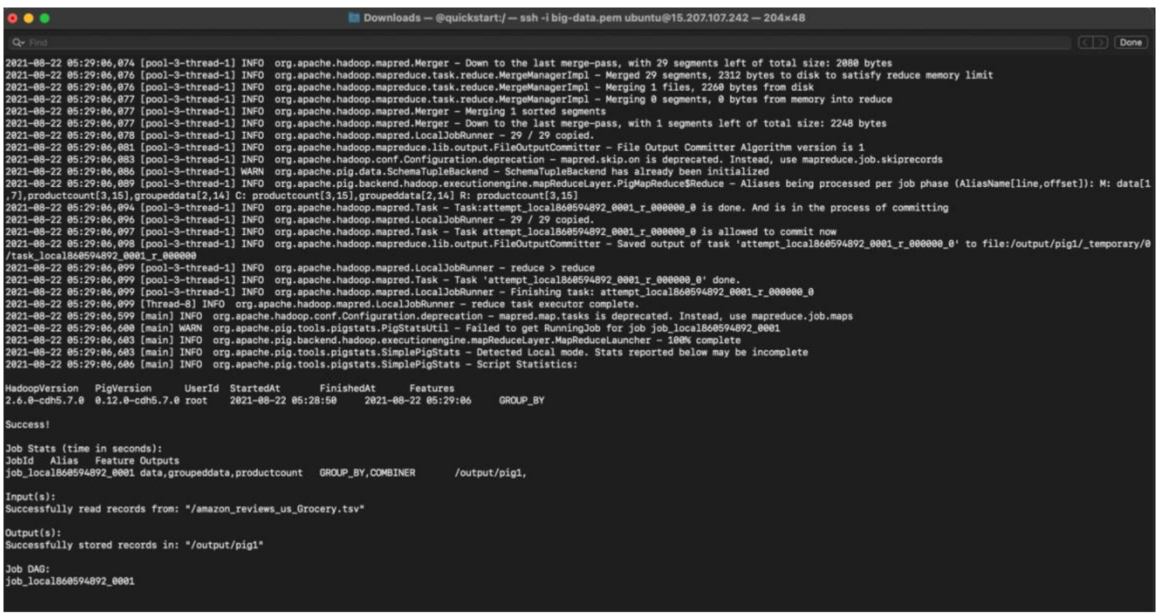
productcount = FOREACH groupeddata GENERATE group as star_rating, COUNT(data.product_id) as count;

store productcount INTO '/output/pig1';

Output:



```
Downloads — @quickstart:/ — ssh -i big-data.pem ubuntu@15.207.107.242 — 204×48
Details at logfile: //pig_1629610039476.log
grunt> data = LOAD '/amazon_reviews_us_Grocery.tsv' AS (marketplace, customer_id, review_id, product_id, product_parent, product_title, product_category, star_rating, helpful_votes, total_votes, vine, verified_purchase, review_headline, review_body, review_date);
grunt> groupeddata = GROUP data by star_rating;
grunt> productcount = FOREACH groupeddata GENERATE group as star_rating, COUNT(data.product_id) as count;
grunt> store productcount INTO '/output/pig1';
2021-08-22 05:28:49,938 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP_BY
2021-08-22 05:28:49,980 [main] INFO  org.apache.pig.newplan.logical.optimizer.LogicalPlanOptimizer - {RULES_ENABLED=[AddForEach, ColumnMapKeyPrune, DuplicateForEachColumnRewrite, GroupByConstParallelSetter, ImplicitSplitInserter, LimitOptimizer, LoadTypeCastInserter, MergeFilter, MergeForEach, NewPartitionFilterOptimizer, PushDownForEachFlatten, PushUpFilter, SplitFilter, StreamTypeCastInserter], RULES_DISABLED=[FilterLogicExpressionSimplifier, PartitionFilterOptimizer]}
2021-08-22 05:28:50,005 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.textoutputformat.separator is deprecated. Instead, use mapreduce.output.textoutputformat.separator
2021-08-22 05:28:50,088 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MRCompiler - File concatenation threshold: 100 optimistic? false
2021-08-22 05:28:50,103 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.CombinerOptimizer - Choosing to move algebraic foreach to combiner
2021-08-22 05:28:50,127 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size before optimization: 1
2021-08-22 05:28:50,127 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MultiQueryOptimizer - MR plan size after optimization: 1
2021-08-22 05:28:50,151 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - session.id is deprecated. Instead, use dfs.metrics.session-id
2021-08-22 05:28:50,151 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Initializing JVM Metrics with processName=JobTracker, sessionId=
2021-08-22 05:28:50,173 [main] INFO  org.apache.pig.tools.pigstats.ScriptState - Pig script settings are added to the job
2021-08-22 05:28:50,222 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.reduce.markreset.buffer.percent is deprecated. Instead, use mapreduce.reduce.markreset.buffer.percent
2021-08-22 05:28:50,222 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - mapred.job.reduce.markreset.buffer.percent is not set, set to default 0.3
2021-08-22 05:28:50,222 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.output.compress is deprecated. Instead, use mapreduce.output.fileoutputformat.compress
2021-08-22 05:28:50,224 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Reduce phase detected, estimating # of required reducers.
2021-08-22 05:28:50,225 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Using reducer estimator: org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.InputSizeReducerEstimator
2021-08-22 05:28:50,226 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.InputSizeReducerEstimator - BytesPerReducer=1000000000 maxReducers=999 totalInputFileSize=956224754
2021-08-22 05:28:50,227 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Setting Parallelism to 1
2021-08-22 05:28:50,227 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.reduce.tasks is deprecated. Instead, use mapreduce.job.reduces
2021-08-22 05:28:50,251 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.JobControlCompiler - Setting up single store job
2021-08-22 05:28:50,258 [main] INFO  org.apache.pig.data.SchemaTupleFrontend - Key [pig.schematuple] is false, will not generate code.
2021-08-22 05:28:50,258 [main] INFO  org.apache.pig.data.SchemaTupleFrontend - Starting process to move generated code to distributed cache
2021-08-22 05:28:50,258 [main] INFO  org.apache.pig.data.SchemaTupleFrontend - Distributed cache not supported or needed in local mode. Setting key [pig.schematuple.local.dir] with code temp directory: /tmp/1629610130257-0
2021-08-22 05:28:50,336 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 1 map-reduce job(s) waiting for submission.
2021-08-22 05:28:50,337 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker.http.address is deprecated. Instead, use mapreduce.jobtracker.http.address
2021-08-22 05:28:50,355 [JobControl] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM Metrics with processName=JobTracker, sessionId= - already initialized
2021-08-22 05:28:50,536 [JobControl] WARN  org.apache.hadoop.mapreduce.JobResourceUploader - No job jar file set.  User classes may not be found. See Job or Job#setJar(String).
2021-08-22 05:28:50,565 [JobControl] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input paths to process : 1
2021-08-22 05:28:50,565 [JobControl] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
2021-08-22 05:28:50,587 [JobControl] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths (combined) to process : 29
2021-08-22 05:28:50,612 [JobControl] INFO  org.apache.hadoop.mapreduce.JobSubmitter - number of splits:29
2021-08-22 05:28:50,620 [JobControl] INFO  org.apache.hadoop.conf.Configuration.deprecation - fs.default.name is deprecated. Instead, use fs.defaultFS
2021-08-22 05:28:50,620 [JobControl] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tracker is deprecated. Instead, use mapreduce.jobtracker.address
2021-08-22 05:28:50,622 [JobControl] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2021-08-22 05:28:50,751 [JobControl] INFO  org.apache.hadoop.mapreduce.JobSubmitter - Submitting tokens for job: job_local860594892_0001
2021-08-22 05:28:50,916 [JobControl] INFO  org.apache.hadoop.mapreduce.Job - The url to track the job: http://localhost:8080/
2021-08-22 05:28:50,916 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - HadoopJobId: job_local860594892_0001
```



```
Downloads — @quickstart:/ — ssh -i big-data.pem ubuntu@15.207.107.242 — 204×48
2021-08-22 05:29:06,074 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.Merger - Down to the last merge-pass, with 29 segments left of total size: 2080 bytes
2021-08-22 05:29:06,076 [pool-3-thread-1] INFO  org.apache.hadoop.mapreduce.task.reduce.MergeManagerImpl - Merged 29 segments, 2312 bytes to disk to satisfy reduce memory limit
2021-08-22 05:29:06,076 [pool-3-thread-1] INFO  org.apache.hadoop.mapreduce.task.reduce.MergeManagerImpl - Merging 1 files, 2260 bytes from disk
2021-08-22 05:29:06,077 [pool-3-thread-1] INFO  org.apache.hadoop.mapreduce.task.reduce.MergeManagerImpl - Merging 0 segments, 0 bytes from memory into reduce
2021-08-22 05:29:06,077 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.Merger - Merging 1 sorted segments
2021-08-22 05:29:06,077 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.Merger - Down to the last merge-pass, with 1 segments left of total size: 2248 bytes
2021-08-22 05:29:06,078 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.LocalJobRunner - 29 / 29 copied.
2021-08-22 05:29:06,081 [pool-3-thread-1] INFO  org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter - File Output Committer Algorithm version is 1
2021-08-22 05:29:06,083 [pool-3-thread-1] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.skip.on is deprecated. Instead, use mapreduce.job.skiprecords
2021-08-22 05:29:06,086 [pool-3-thread-1] WARN  org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2021-08-22 05:29:06,089 [pool-3-thread-1] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.PigMapReduce$Reduce - Aliases being processed per job phase (AliasName[line,offset]): M: data[1,7],productcount[3,15],groupeddata[2,14] C: productcount[3,15],groupeddata[2,14] R: productcount[3,15]
2021-08-22 05:29:06,094 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.Task - Task:attempt_local860594892_0001_r_000000_0 is done. And is in the process of committing
2021-08-22 05:29:06,096 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.LocalJobRunner - 29 / 29 copied.
2021-08-22 05:29:06,097 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.Task - Task attempt_local860594892_0001_r_000000_0 is allowed to commit now
2021-08-22 05:29:06,098 [pool-3-thread-1] INFO  org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter - Saved output of task 'attempt_local860594892_0001_r_000000_0' to file:/output/pig1/_temporary/0/task_local860594892_0001_r_000000
2021-08-22 05:29:06,099 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.LocalJobRunner - reduce > reduce
2021-08-22 05:29:06,099 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.Task - Task 'attempt_local860594892_0001_r_000000_0' done.
2021-08-22 05:29:06,099 [pool-3-thread-1] INFO  org.apache.hadoop.mapred.LocalJobRunner - Finishing task: attempt_local860594892_0001_r_000000_0
2021-08-22 05:29:06,099 [Thread-8] INFO  org.apache.hadoop.mapred.LocalJobRunner - reduce task executor complete.
2021-08-22 05:29:06,599 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.map.tasks is deprecated. Instead, use mapreduce.job.maps
2021-08-22 05:29:06,600 [main] WARN  org.apache.pig.tools.pigstats.PigStatsUtil - Failed to get RunningJob for job job_local860594892_0001
2021-08-22 05:29:06,603 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - 100% complete
2021-08-22 05:29:06,603 [main] INFO  org.apache.pig.tools.pigstats.SimplePigStats - Detected Local mode. Stats reported below may be incomplete
2021-08-22 05:29:06,606 [main] INFO  org.apache.pig.tools.pigstats.SimplePigStats - Script Statistics:

HadoopVersion   PigVersion      UserId  StartedAt       FinishedAt      Features
2.6.0-cdh5.7.0  0.12.0-cdh5.7.0 root    2021-08-22 05:28:50     2021-08-22 05:29:06     GROUP_BY

Success!

Job Stats (time in seconds):
JobId   Alias   Feature Outputs
job_local860594892_0001 data,groupeddata,productcount   GROUP_BY,COMBINER       /output/pig1,

Input(s):
Successfully read records from: "/amazon_reviews_us_Grocery.tsv"

Output(s):
Successfully stored records in: "/output/pig1"

Job DAG:
job_local860594892_0001
```

Visualization (Tableau)

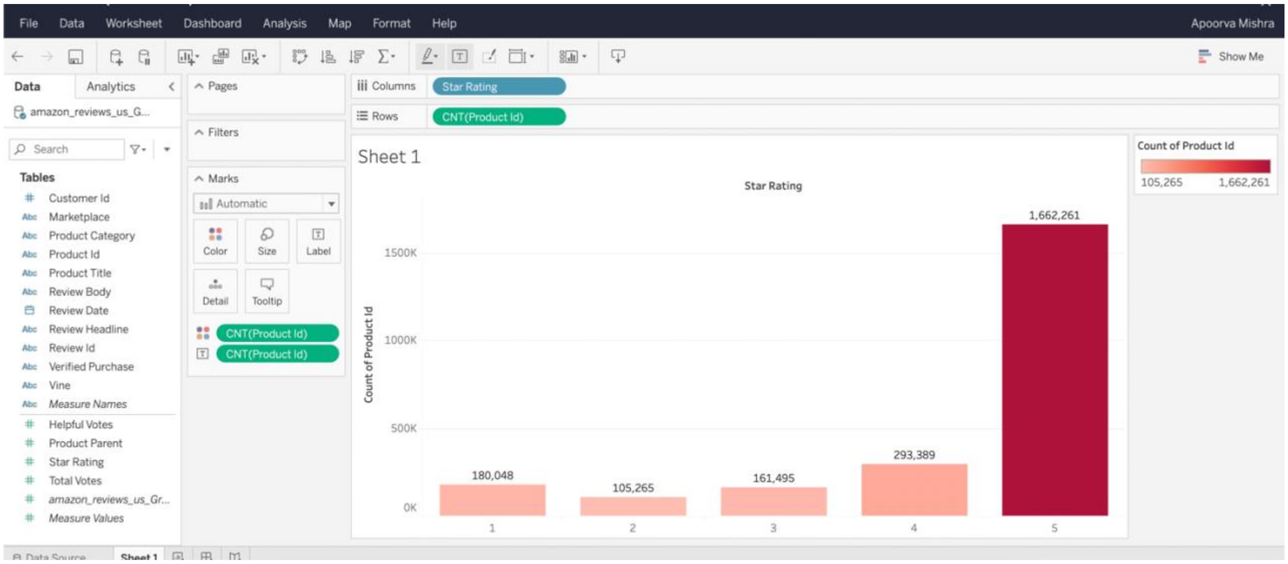


Fig: Total number of products of each rating

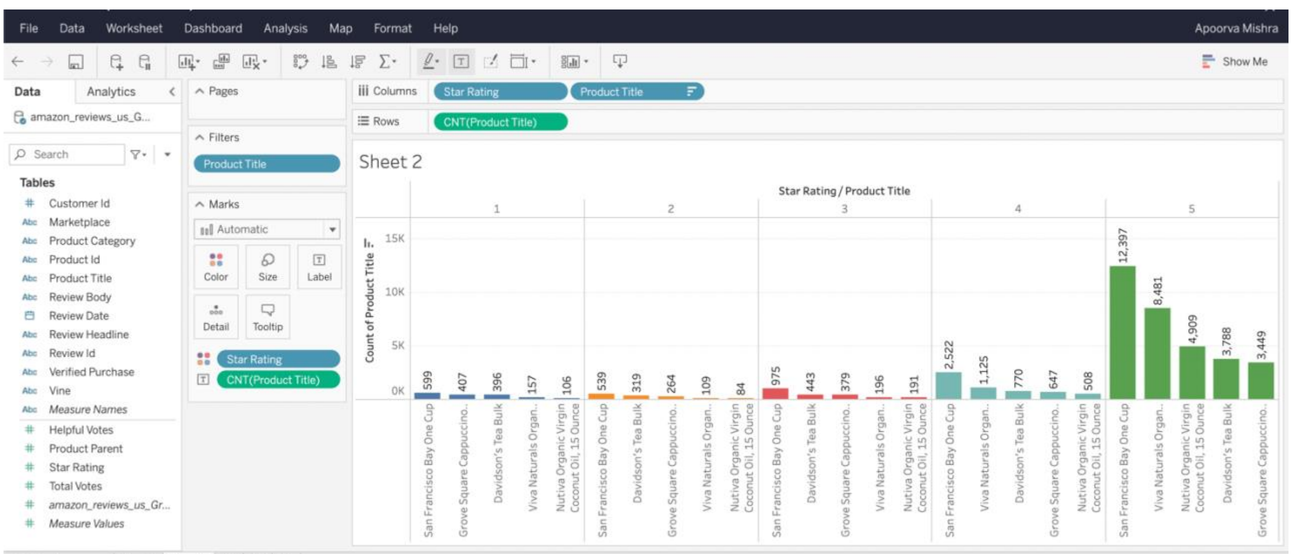


Fig: Top 5 products of each rating

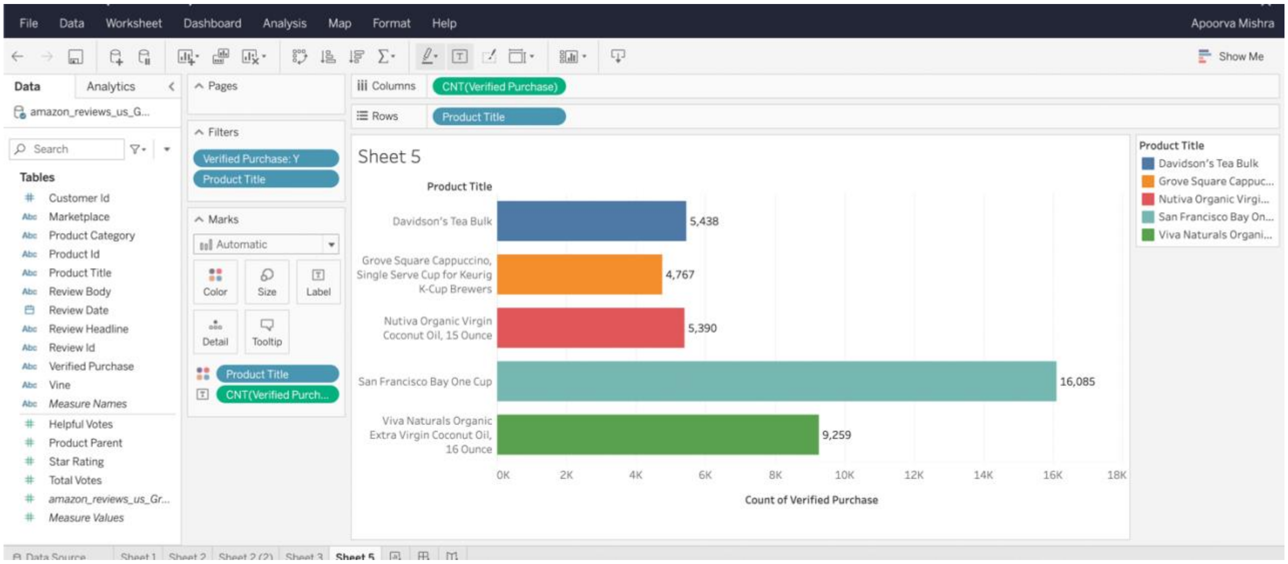


Fig: Top 5 verified purchased products

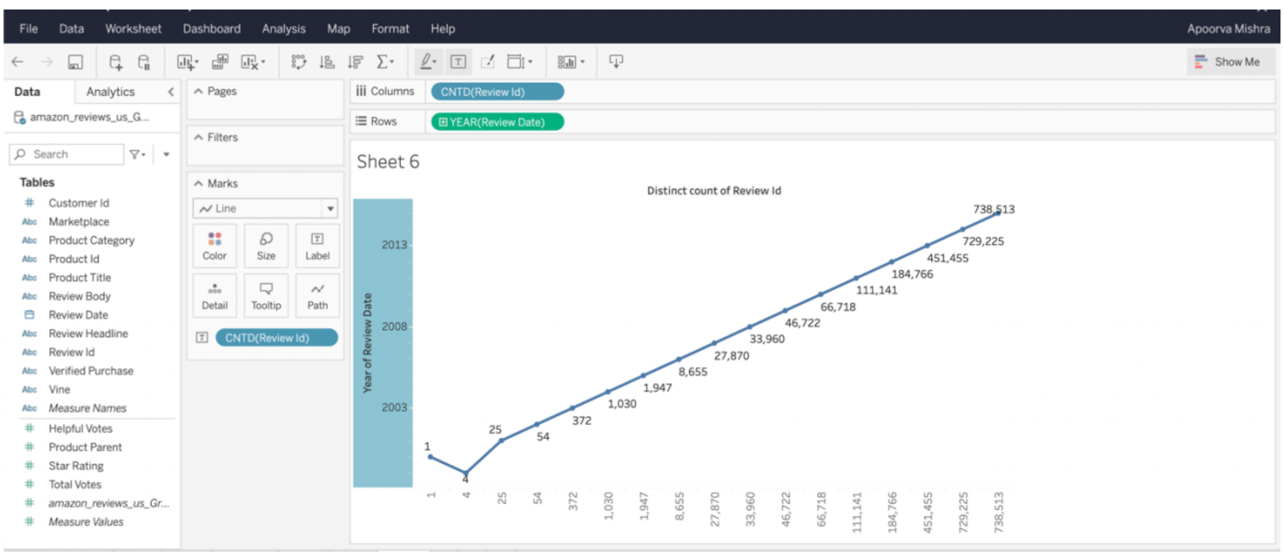


Fig: Total reviews by year