INFO 7250 Final Project Report
Title: Yelp Dataset Analysis

Author: Jixiao Yang

1. Project Summary

The Yelp dataset is a subset of our businesses, reviews, and user data for use in personal, educational, and academic purposes. Available as JSON files, use it to teach students about databases, to learn NLP, or for sample production data while you learn how to make mobile apps.
The Yelp dataset contains about 6,685,900 reviews, 192,609 businesses, 200,000 pictures, 10 metropolitan areas, 1,223,094 tips by 1,637,138 users, over 1.2 million business attributes like hours, parking, availability, and ambience, and aggregated check-ins over time for each of the 192,609 businesses. Such big dataset is very suitable for big data analysis.
In this project, I use some technologies to do MapReduce jobs to analyze yelp dataset and compare the running time of them.

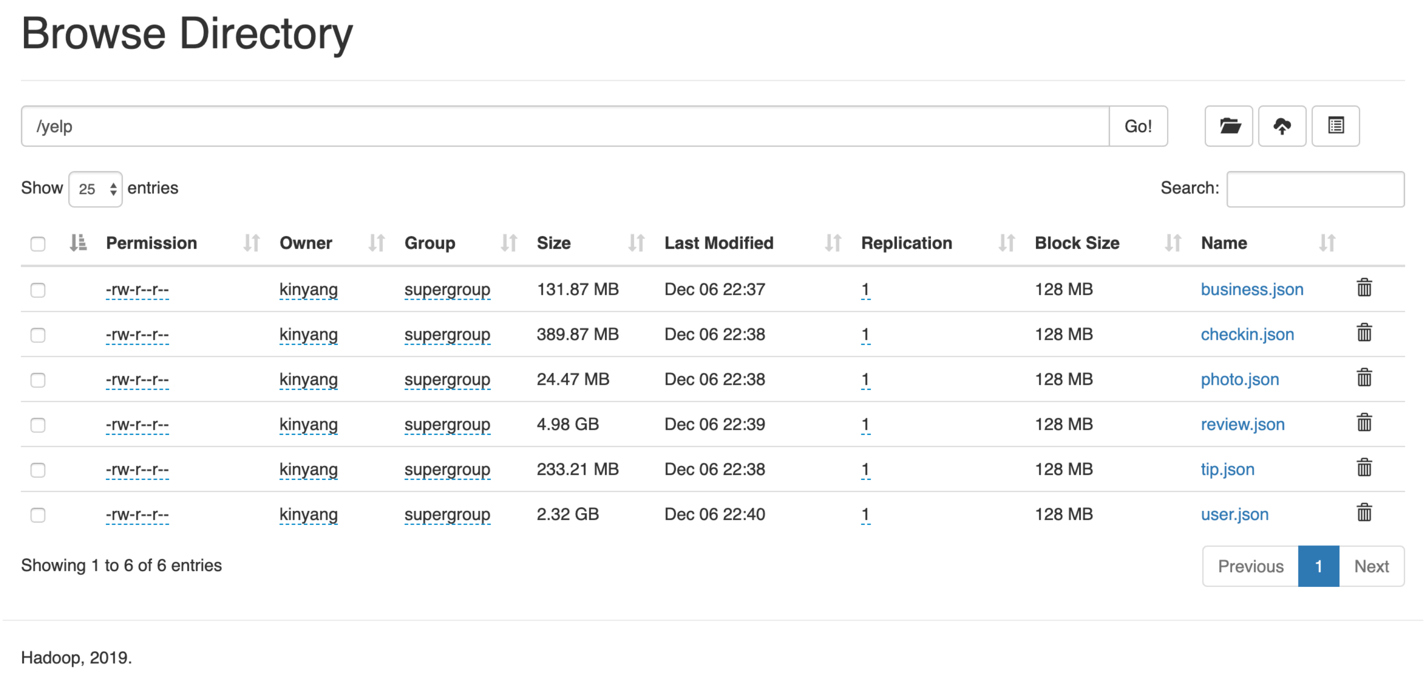
2. Sources and Technologies

Dataset: Yelp Dataset (https://www.yelp.com/dataset)
Technologies: Apache Hadoop 3.2.1, Apache HBase 2.2.2, Apache Hive 3.1.2

3. Apache Hadoop Analysis

(1) Data Import

Here is the whole yelp dataset without picture data.



Browse Directory

/yelp

Show 25 entries

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	kinyang	supergroup	131.87 MB	Dec 06 22:37	1	128 MB	business.json
-rw-r--r--	kinyang	supergroup	389.87 MB	Dec 06 22:38	1	128 MB	checkin.json
-rw-r--r--	kinyang	supergroup	24.47 MB	Dec 06 22:38	1	128 MB	photo.json
-rw-r--r--	kinyang	supergroup	4.98 GB	Dec 06 22:39	1	128 MB	review.json
-rw-r--r--	kinyang	supergroup	233.21 MB	Dec 06 22:38	1	128 MB	tip.json
-rw-r--r--	kinyang	supergroup	2.32 GB	Dec 06 22:40	1	128 MB	user.json

Showing 1 to 6 of 6 entries

Hadoop, 2019.

Sample Data in *business.json*

```
{
    // string, 22 character unique string business id
    "business_id": "tnhfDv5Il8EaGSXZGiuQGg",

    // string, the business's name
    "name": "Garaje",

    // string, the full address of the business
    "address": "475 3rd St",

    // string, the city
    "city": "San Francisco",

    // string, 2 character state code, if applicable
    "state": "CA",
    // string, the postal code
    "postal code": "94107",

    // float, latitude
    "latitude": 37.7817529521,

    // float, longitude
    "longitude": -122.39612197,

    // float, star rating, rounded to half-stars
    "stars": 4.5,

    // integer, number of reviews
    "review_count": 1198,

    // integer, 0 or 1 for closed or open, respectively
    "is_open": 1,

    // object, business attributes to values. note: some attribute values might be objects
    "attributes": {
        "RestaurantsTakeOut": true,
        "BusinessParking": {
            "garage": false,
            "street": true,
            "validated": false,
            "lot": false,
            "valet": false
        },
    },

    // an array of strings of business categories
    "categories": [
        "Mexican",
        "Burgers",
        "Gastropubs"
    ],

    // an object of key day to value hours, hours are using a 24hr clock
    "hours": {
        "Monday": "10:00-21:00",
        "Tuesday": "10:00-21:00",
        "Friday": "10:00-21:00",
        "Wednesday": "10:00-21:00",
        "Thursday": "10:00-21:00",
        "Sunday": "11:00-18:00",
        "Saturday": "10:00-21:00"
    }
}
```

Sample Data in *review.json*

```
{
    // string, 22 character unique review id
    "review_id": "zdSx_SD6obEhz9VrW9uAWA",

    // string, 22 character unique user id, maps to the user in user.json
    "user_id": "Ha3iJu77CxlrFm-vQRs_8g",

    // string, 22 character business id, maps to business in business.json
    "business_id": "tnhfDv5Il8EaGSXZGiuQGg",

    // integer, star rating
    "stars": 4,

    // string, date formatted YYYY-MM-DD
    "date": "2016-03-09",

    // string, the review itself
    "text": "Great place to hang out after work: the prices are decent, and the ambience is fun. It's a
bit loud, but very lively. The staff is friendly, and the food is good. They have a good selection of
drinks.",

    // integer, number of useful votes received
    "useful": 0,

    // integer, number of funny votes received
    "funny": 0,

    // integer, number of cool votes received
    "cool": 0
}
```

(2) MapReduce – Numbers of Businesses in Each City and State

In this analysis, the key is state and city of business. Therefore, we can generate a class to store these two values as a key to MapReduce.

Terminal Command:

```
nohup hadoop jar Hadoop/target/Hadoop-1.0-SNAPSHOT.jar Business.NumersInEachCityAndState.MapReduc
e /yelp/business.json /fpresults/hadoop/NumberOfBusinessesEachCityAndState > shell_logs/hadoop_nu
mber_of_businesses_each_city.log &
```

Command Log:

```
2019-12-14 00:56:13,216 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-12-14 00:56:13,632 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not
performed. Implement the Tool interface and execute your application with ToolRunner to remedy
this.
2019-12-14 00:56:13,646 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path:
/tmp/hadoop-yarn/staging/kinyang/.staging/job_1576235822595_0022
2019-12-14 00:56:13,737 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:56:13,850 INFO input.FileInputFormat: Total input files to process : 1
2019-12-14 00:56:13,879 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:56:13,895 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:56:13,900 INFO mapreduce.JobSubmitter: number of splits:1
2019-12-14 00:56:14,043 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:56:14,469 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1576235822595_0022
2019-12-14 00:56:14,469 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-12-14 00:56:14,630 INFO conf.Configuration: resource-types.xml not found
2019-12-14 00:56:14,630 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-12-14 00:56:14,728 INFO impl.YarnClientImpl: Submitted application
application_1576235822595_0022
2019-12-14 00:56:14,799 INFO mapreduce.Job: The url to track the job:
http://localhost:8088/proxy/application_1576235822595_0022/
2019-12-14 00:56:14,800 INFO mapreduce.Job: Running job: job_1576235822595_0022
2019-12-14 00:56:20,908 INFO mapreduce.Job: Job job_1576235822595_0022 running in uber mode : false
2019-12-14 00:56:20,909 INFO mapreduce.Job:  map 0% reduce 0%
2019-12-14 00:56:29,007 INFO mapreduce.Job:  map 100% reduce 0%
2019-12-14 00:56:33,029 INFO mapreduce.Job:  map 100% reduce 100%
2019-12-14 00:56:34,046 INFO mapreduce.Job: Job job_1576235822595_0022 completed successfully
2019-12-14 00:56:34,125 INFO mapreduce.Job: Counters: 50
    File System Counters
        FILE: Number of bytes read=3918462
        FILE: Number of bytes written=8289927
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=138279854
        HDFS: Number of bytes written=21730
        HDFS: Number of read operations=8
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Launched map tasks=1
        Launched reduce tasks=1
        Data-local map tasks=1
        Total time spent by all maps in occupied slots (ms)=5748
        Total time spent by all reduces in occupied slots (ms)=2402
        Total time spent by all map tasks (ms)=5748
        Total time spent by all reduce tasks (ms)=2402
        Total vcore-milliseconds taken by all map tasks=5748
        Total vcore-milliseconds taken by all reduce tasks=2402
        Total megabyte-milliseconds taken by all map tasks=5885952
        Total megabyte-milliseconds taken by all reduce tasks=2459648
    Map-Reduce Framework
        Map input records=192609
        Map output records=192609
        Map output bytes=3533238
        Map output materialized bytes=3918462
        Input split bytes=105
        Combine input records=0
        Combine output records=0
        Reduce input groups=1258
        Reduce shuffle bytes=3918462
        Reduce input records=192609
        Reduce output records=1258
        Spilled Records=385218
        Shuffled Maps =1
        Failed Shuffles=0
        Merged Map outputs=1
        GC time elapsed (ms)=250
        CPU time spent (ms)=0
        Physical memory (bytes) snapshot=0
        Virtual memory (bytes) snapshot=0
        Total committed heap usage (bytes)=727711744
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=138279749
    File Output Format Counters
        Bytes Written=21730
Hadoop Number of Businesses in each city and state MapReduce Time: 20980 ms
```

MapReduce Result (Part):

```
AB	Airdrie	168
AB	Alberta	1
AB	Balzac	11
AB	Beltline	1
AB	CALGARY	1
AB	Calgary	7735
AB	Chestermere	31
AB	De Winton	1
AB	Division No. 6	3
AB	Downtown	1
AB	East Calgary	1
AB	Edgemont	1
AB	Edmonton	2
AB	Evergreen	1
AB	Highland Park	1
AB	Medicine Hat	1
AB	Midnapore	1
AB	Montreal	1
AB	North York	1
AB	Northeast Calgary	1
AB	Northwest Calgary	3
AB	Rockey View	1
AB	Rocky View	20
AB	Rocky View County	9
AB	Rocky View No. 44	4
AB	Rockyview	2
AB	Rockyview County	1
AB	SW Calgary	1
AB	Sage Hill	1
AB	Sainte-Adèle	1
AB	Southeast Calgary	3
AB	Toronto	1
AB	calgary	1
```

(3) MapReduce – Top 5 Rated Business in Each State

This analysis is referenced from the ***Top 10 List Algorithm***. Since we need to calculate the top 5 in each state, I choose to use a ***HashMap*** to calculate the result. The key is state name and the value is a ***TreeSet*** which stores the information of a business. A ***WritableComparable*** class is generated to store the information of business, which can be sorted by rating.

Terminal Command:

```
nohup hadoop jar Hadoop/target/Hadoop-1.0-SNAPSHOT.jar Business.Top5RatedBusinessInEachState.MapReduce /yelp/business.json /fpresults/hadoop/Top5RatedBusinessInEachState > shell_logs/hadoop_top_5_rated_business_in_each_state.log &
```

Command Log:

```
2019-12-13 04:31:37,457 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-12-13 04:31:37,932 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
2019-12-13 04:31:37,948 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/kinyang/.staging/job_1576224793999_0010
2019-12-13 04:31:38,050 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:31:38,168 INFO input.FileInputFormat: Total input files to process : 1
2019-12-13 04:31:38,200 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:31:38,216 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:31:38,221 INFO mapreduce.JobSubmitter: number of splits:1
2019-12-13 04:31:38,407 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:31:38,429 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1576224793999_0010
```

```
2019-12-13 04:31:38,429 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-12-13 04:31:38,597 INFO conf.Configuration: resource-types.xml not found
2019-12-13 04:31:38,597 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-12-13 04:31:38,672 INFO impl.YarnClientImpl: Submitted application
application_1576224793999_0010
2019-12-13 04:31:38,749 INFO mapreduce.Job: The url to track the job:
http://localhost:8088/proxy/application_1576224793999_0010/
2019-12-13 04:31:38,750 INFO mapreduce.Job: Running job: job_1576224793999_0010
2019-12-13 04:31:45,852 INFO mapreduce.Job: Job job_1576224793999_0010 running in uber mode : false
2019-12-13 04:31:45,855 INFO mapreduce.Job:  map 0% reduce 0%
2019-12-13 04:31:52,945 INFO mapreduce.Job:  map 100% reduce 0%
2019-12-13 04:31:57,981 INFO mapreduce.Job:  map 100% reduce 100%
2019-12-13 04:31:57,989 INFO mapreduce.Job: Job job_1576224793999_0010 completed successfully
2019-12-13 04:31:58,073 INFO mapreduce.Job: Counters: 50
    File System Counters
        FILE: Number of bytes read=6781
        FILE: Number of bytes written=467335
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=138279854
        HDFS: Number of bytes written=10944
        HDFS: Number of read operations=8
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Launched map tasks=1
        Launched reduce tasks=1
        Data-local map tasks=1
        Total time spent by all maps in occupied slots (ms)=5388
        Total time spent by all reduces in occupied slots (ms)=2215
        Total time spent by all map tasks (ms)=5388
        Total time spent by all reduce tasks (ms)=2215
        Total vcore-milliseconds taken by all map tasks=5388
        Total vcore-milliseconds taken by all reduce tasks=2215
        Total megabyte-milliseconds taken by all map tasks=5517312
        Total megabyte-milliseconds taken by all reduce tasks=2268160
    Map-Reduce Framework
        Map input records=192609
        Map output records=110
        Map output bytes=6555
        Map output materialized bytes=6781
        Input split bytes=105
        Combine input records=0
        Combine output records=0
        Reduce input groups=36
        Reduce shuffle bytes=6781
        Reduce input records=110
        Reduce output records=110
        Spilled Records=220
        Shuffled Maps =1
        Failed Shuffles=0
        Merged Map outputs=1
        GC time elapsed (ms)=113
        CPU time spent (ms)=0
        Physical memory (bytes) snapshot=0
        Virtual memory (bytes) snapshot=0
        Total committed heap usage (bytes)=607125504
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=138279749
    File Output Format Counters
        Bytes Written=10944
Hadoop Top 5 rated businesses in each state MapReduce Time: 20690 ms
```
MapReduce Result (First 10 Records):
```
{businessId: '2HmjQlkl4dzZTAbSTpTbog', businessName: '20/20 Master Home Inspections', state: 'AB', rate: 
5.0}
{businessId: 'TXTCF0pCOB9IIoJme6ISuA', businessName: '24/7 Electric', state: 'AB', rate: 5.0}
{businessId: 'eV8qKYa60WVNYidUXEzR4A', businessName: '41'st Auto', state: 'AB', rate: 5.0}
{businessId: 'EqyI7DwA9i5AQgDM-EKzSA', businessName: '4Cats Arts Studio', state: 'AB', rate: 5.0}
{businessId: '9WszpXOZSTRq73YPV7Z6mw', businessName: '9-1-PLUMB', state: 'AB', rate: 5.0}
{businessId: 'W839iVmIb3dKrU_-eyOhQA', businessName: 'Mathnasium of Queen Creek', state: 'AK', rate: 4.0}
{businessId: 'WwwYZakdSQM9174gdZdUIA', businessName: 'The Wing Warehouse', state: 'AK', rate: 1.5}
{businessId: 'sSlMkHBYFOMYbrYG5Jg0Bw', businessName: 'Mr Pretzels', state: 'AL', rate: 3.5}
{businessId: '6NAWNCgdLHeMh3wHRgu6vw', businessName: 'Chevron', state: 'AL', rate: 3.0}
{businessId: 'sbtxQN1-pxyfNr_aVYew9Q', businessName: 'Tiffani Tonkinson, DDS', state: 'AL', rate: 5.0}
```
(4) MapReduce – Percentage of Ratings

In this analysis, the key is the rate of each review. We need to count the number of each rate and the total review number to calculate the percentage of each rating.

Terminal Command:
```
nohup hadoop jar Hadoop/target/Hadoop-1.0-SNAPSHOT.jar Review.PercentageOfRatings.MapReduce /yelp/
review.json /fpresults/hadoop/PercentageOfRatings > shell_logs/hadoop_percentage_of_ratings.log &
```
Command Log:
```
2019-12-13 23:54:42,041 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-12-13 23:54:42,669 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not 
performed. Implement the Tool interface and execute your application with ToolRunner to remedy 
this.
2019-12-13 23:54:42,694 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: 
/tmp/hadoop-yarn/staging/kinyang/.staging/job_1576235822595_0018
2019-12-13 23:54:42,814 INFO sasl.SaslDataTransferClient: SASL encryption trust check: 
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 23:54:43,010 INFO input.FileInputFormat: Total input files to process : 1
2019-12-13 23:54:43,063 INFO sasl.SaslDataTransferClient: SASL encryption trust check: 
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 23:54:43,079 INFO sasl.SaslDataTransferClient: SASL encryption trust check: 
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 23:54:43,086 INFO mapreduce.JobSubmitter: number of splits:40
2019-12-13 23:54:43,255 INFO sasl.SaslDataTransferClient: SASL encryption trust check: 
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 23:54:43,688 INFO mapreduce.JobSubmitter: Submitting tokens for job: 
job_1576235822595_0018
2019-12-13 23:54:43,688 INFO mapreduce.JobSubmitter: Executing with tokens: []
```

```
2019-12-13 23:54:43,938 INFO conf.Configuration: resource-types.xml not found
2019-12-13 23:54:43,938 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-12-13 23:54:44,045 INFO impl.YarnClientImpl: Submitted application
application_1576235822595_0018
2019-12-13 23:54:44,154 INFO mapreduce.Job: The url to track the job:
http://localhost:8088/proxy/application_1576235822595_0018/
2019-12-13 23:54:44,155 INFO mapreduce.Job: Running job: job_1576235822595_0018
2019-12-13 23:54:51,326 INFO mapreduce.Job: Job job_1576235822595_0018 running in uber mode : false
2019-12-13 23:54:51,328 INFO mapreduce.Job:  map 0% reduce 0%
2019-12-13 23:55:14,633 INFO mapreduce.Job:  map 15% reduce 0%
2019-12-13 23:55:32,816 INFO mapreduce.Job:  map 17% reduce 0%
2019-12-13 23:55:33,835 INFO mapreduce.Job:  map 25% reduce 0%
2019-12-13 23:55:34,844 INFO mapreduce.Job:  map 28% reduce 0%
2019-12-13 23:55:35,850 INFO mapreduce.Job:  map 30% reduce 0%
2019-12-13 23:55:51,940 INFO mapreduce.Job:  map 40% reduce 0%
2019-12-13 23:55:52,947 INFO mapreduce.Job:  map 43% reduce 0%
2019-12-13 23:55:56,967 INFO mapreduce.Job:  map 43% reduce 14%
2019-12-13 23:56:07,021 INFO mapreduce.Job:  map 47% reduce 14%
2019-12-13 23:56:08,030 INFO mapreduce.Job:  map 55% reduce 14%
2019-12-13 23:56:09,036 INFO mapreduce.Job:  map 55% reduce 18%
2019-12-13 23:56:22,099 INFO mapreduce.Job:  map 57% reduce 18%
2019-12-13 23:56:23,110 INFO mapreduce.Job:  map 65% reduce 18%
2019-12-13 23:56:24,117 INFO mapreduce.Job:  map 68% reduce 18%
2019-12-13 23:56:27,133 INFO mapreduce.Job:  map 68% reduce 23%
2019-12-13 23:56:37,190 INFO mapreduce.Job:  map 75% reduce 23%
2019-12-13 23:56:38,198 INFO mapreduce.Job:  map 80% reduce 23%
2019-12-13 23:56:39,205 INFO mapreduce.Job:  map 80% reduce 26%
2019-12-13 23:56:45,234 INFO mapreduce.Job:  map 80% reduce 27%
2019-12-13 23:56:54,277 INFO mapreduce.Job:  map 85% reduce 27%
2019-12-13 23:56:55,281 INFO mapreduce.Job:  map 90% reduce 27%
2019-12-13 23:56:56,288 INFO mapreduce.Job:  map 93% reduce 27%
2019-12-13 23:56:57,291 INFO mapreduce.Job:  map 93% reduce 31%
2019-12-13 23:57:05,332 INFO mapreduce.Job:  map 100% reduce 31%
2019-12-13 23:57:09,358 INFO mapreduce.Job:  map 100% reduce 100%
2019-12-13 23:57:09,364 INFO mapreduce.Job: Job job_1576235822595_0018 completed successfully
2019-12-13 23:57:09,517 INFO mapreduce.Job: Counters: 51
    File System Counters
        FILE: Number of bytes read=93602606
        FILE: Number of bytes written=196505369
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=5347639502
        HDFS: Number of bytes written=113
        HDFS: Number of read operations=125
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Killed map tasks=1
        Launched map tasks=40
        Launched reduce tasks=1
        Data-local map tasks=40
```

```
		Total time spent by all maps in occupied slots (ms)=649287
		Total time spent by all reduces in occupied slots (ms)=92838
		Total time spent by all map tasks (ms)=649287
		Total time spent by all reduce tasks (ms)=92838
		Total vcore-milliseconds taken by all map tasks=649287
		Total vcore-milliseconds taken by all reduce tasks=92838
		Total megabyte-milliseconds taken by all map tasks=664869888
		Total megabyte-milliseconds taken by all reduce tasks=95066112
	Map-Reduce Framework
		Map input records=6685900
		Map output records=6685900
		Map output bytes=80230800
		Map output materialized bytes=93602840
		Input split bytes=4120
		Combine input records=0
		Combine output records=0
		Reduce input groups=5
		Reduce shuffle bytes=93602840
		Reduce input records=6685900
		Reduce output records=5
		Spilled Records=13371800
		Shuffled Maps =40
		Failed Shuffles=0
		Merged Map outputs=40
		GC time elapsed (ms)=7655
		CPU time spent (ms)=0
		Physical memory (bytes) snapshot=0
		Virtual memory (bytes) snapshot=0
		Total committed heap usage (bytes)=17534287872
	Shuffle Errors
		BAD_ID=0
		CONNECTION=0
		IO_ERROR=0
		WRONG_LENGTH=0
		WRONG_MAP=0
		WRONG_REDUCE=0
	File Input Format Counters
		Bytes Read=5347635382
	File Output Format Counters
		Bytes Written=113
Hadoop Percentage of Ratings MapReduce Time: 147569 ms
```

MapReduce Result:

```
1.0	14.989141327270824
2.0	8.11250542185794
3.0	11.057299690393215
4.0	21.971387546927115
5.0	43.869666013550905
```

(5) MapReduce – Moving Average Rating of Each Business – 10 Reviews Each

This analysis is referenced from the ***Moving Average Algorithm***. We calculate the moving average rating of each businesses in every 10 reviews, sorted by review datetime. We can store the business_id and timestamp in a ***CompositeKey*** class as the key of the Mapper and store the timestamp and rating in a ***BusinessRatingData*** class as the value of the Mapper. Using the ***Secondary Sort Algorithm*** and ***Partitioner***, we can let the keys have same business_id but different timestamp can be partitioned into same Reducer, sorted by timestamp.

In the Reducer class, we do the ***Moving Average Algorithm***. We can use ***Queue*** to calculate the moving average value. When we iterate each value in a Reducer, we add the rating value into the queue. If the queue size is more than 10, we pop the front value of the queue. Then we calculate a moving average value as the result.
Terminal Command:

```
nohup hadoop jar Hadoop/target/Hadoop-1.0-SNAPSHOT.jar Review.MovingAverageRatingOfEachBusiness.Ma
pReduce /yelp/review.json /fpresults/hadoop/MovingAverageRatingOfEachBusiness > shell_logs/hadoop_
moving_average_rating_of_each_business.log &
```

Command Log:

```
2019-12-13 06:19:29,980 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-12-13 06:19:30,485 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not
performed. Implement the Tool interface and execute your application with ToolRunner to remedy
this.
2019-12-13 06:19:30,518 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path:
/tmp/hadoop-yarn/staging/kinyang/.staging/job_1576235822595_0001
2019-12-13 06:19:30,681 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 06:19:30,832 INFO input.FileInputFormat: Total input files to process : 1
2019-12-13 06:19:30,886 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 06:19:30,901 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 06:19:30,909 INFO mapreduce.JobSubmitter: number of splits:40
2019-12-13 06:19:31,098 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 06:19:31,130 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1576235822595_0001
2019-12-13 06:19:31,130 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-12-13 06:19:31,325 INFO conf.Configuration: resource-types.xml not found
2019-12-13 06:19:31,326 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-12-13 06:19:31,788 INFO impl.YarnClientImpl: Submitted application
application_1576235822595_0001
2019-12-13 06:19:31,823 INFO mapreduce.Job: The url to track the job:
http://localhost:8088/proxy/application_1576235822595_0001/
2019-12-13 06:19:31,824 INFO mapreduce.Job: Running job: job_1576235822595_0001
2019-12-13 06:19:38,925 INFO mapreduce.Job: Job job_1576235822595_0001 running in uber mode : false
2019-12-13 06:19:38,928 INFO mapreduce.Job:  map 0% reduce 0%
2019-12-13 06:20:01,092 INFO mapreduce.Job:  map 6% reduce 0%
2019-12-13 06:20:07,169 INFO mapreduce.Job:  map 11% reduce 0%
2019-12-13 06:20:08,176 INFO mapreduce.Job:  map 15% reduce 0%
2019-12-13 06:20:28,294 INFO mapreduce.Job:  map 18% reduce 0%
2019-12-13 06:20:29,307 INFO mapreduce.Job:  map 25% reduce 0%
2019-12-13 06:20:30,313 INFO mapreduce.Job:  map 26% reduce 0%
2019-12-13 06:20:31,319 INFO mapreduce.Job:  map 27% reduce 0%
2019-12-13 06:20:32,328 INFO mapreduce.Job:  map 30% reduce 0%
2019-12-13 06:20:51,460 INFO mapreduce.Job:  map 32% reduce 0%
2019-12-13 06:20:52,464 INFO mapreduce.Job:  map 40% reduce 0%
2019-12-13 06:20:53,470 INFO mapreduce.Job:  map 43% reduce 0%
2019-12-13 06:20:54,477 INFO mapreduce.Job:  map 45% reduce 0%
2019-12-13 06:21:13,582 INFO mapreduce.Job:  map 55% reduce 15%
2019-12-13 06:21:14,588 INFO mapreduce.Job:  map 57% reduce 15%
2019-12-13 06:21:19,615 INFO mapreduce.Job:  map 57% reduce 19%
2019-12-13 06:21:31,684 INFO mapreduce.Job:  map 63% reduce 19%
2019-12-13 06:21:32,692 INFO mapreduce.Job:  map 69% reduce 19%
2019-12-13 06:21:33,698 INFO mapreduce.Job:  map 70% reduce 19%
```

```
2019-12-13 06:21:37,726 INFO mapreduce.Job:  map 70% reduce 23%
2019-12-13 06:21:51,819 INFO mapreduce.Job:  map 72% reduce 23%
2019-12-13 06:21:52,826 INFO mapreduce.Job:  map 76% reduce 23%
2019-12-13 06:21:53,831 INFO mapreduce.Job:  map 80% reduce 23%
2019-12-13 06:21:54,837 INFO mapreduce.Job:  map 82% reduce 23%
2019-12-13 06:21:55,857 INFO mapreduce.Job:  map 82% reduce 28%
2019-12-13 06:22:12,965 INFO mapreduce.Job:  map 90% reduce 28%
2019-12-13 06:22:13,985 INFO mapreduce.Job:  map 95% reduce 28%
2019-12-13 06:22:19,014 INFO mapreduce.Job:  map 95% reduce 32%
2019-12-13 06:22:22,025 INFO mapreduce.Job:  map 98% reduce 32%
2019-12-13 06:22:23,029 INFO mapreduce.Job:  map 100% reduce 32%
2019-12-13 06:22:25,048 INFO mapreduce.Job:  map 100% reduce 45%
2019-12-13 06:22:31,082 INFO mapreduce.Job:  map 100% reduce 67%
2019-12-13 06:22:37,117 INFO mapreduce.Job:  map 100% reduce 76%
2019-12-13 06:22:43,148 INFO mapreduce.Job:  map 100% reduce 85%
2019-12-13 06:22:49,190 INFO mapreduce.Job:  map 100% reduce 96%
2019-12-13 06:22:52,212 INFO mapreduce.Job:  map 100% reduce 100%
2019-12-13 06:22:52,220 INFO mapreduce.Job: Job job_1576235822595_0001 completed successfully
2019-12-13 06:22:52,471 INFO mapreduce.Job: Counters: 51
    File System Counters
        FILE: Number of bytes read=334295006
        FILE: Number of bytes written=677907389
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=5347639502
        HDFS: Number of bytes written=320923200
        HDFS: Number of read operations=125
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Killed map tasks=1
        Launched map tasks=40
        Launched reduce tasks=1
        Data-local map tasks=40
        Total time spent by all maps in occupied slots (ms)=825072
        Total time spent by all reduces in occupied slots (ms)=118291
        Total time spent by all map tasks (ms)=825072
        Total time spent by all reduce tasks (ms)=118291
        Total vcore-milliseconds taken by all map tasks=825072
        Total vcore-milliseconds taken by all reduce tasks=118291
        Total megabyte-milliseconds taken by all map tasks=844873728
        Total megabyte-milliseconds taken by all reduce tasks=121129984
    Map-Reduce Framework
        Map input records=6685900
        Map output records=6685900
        Map output bytes=320923200
        Map output materialized bytes=334295240
        Input split bytes=4120
        Combine input records=0
        Combine output records=0
        Reduce input groups=192606
        Reduce shuffle bytes=334295240
        Reduce input records=6685900
        Reduce output records=6685900
        Spilled Records=13371800
        Shuffled Maps =40
        Failed Shuffles=0
        Merged Map outputs=40
        GC time elapsed (ms)=9685
        CPU time spent (ms)=0
        Physical memory (bytes) snapshot=0
        Virtual memory (bytes) snapshot=0
        Total committed heap usage (bytes)=18078498816
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=5347635382
    File Output Format Counters
        Bytes Written=320923200
Hadoop Moving Average Rating of Each Business MapReduce Time: 202580 ms
```

MapReduce Result (One Business):

```
 --1UhMGODdWsrMastO9DZw      2016-04-21 18:21:32   4.00
 --1UhMGODdWsrMastO9DZw      2016-06-04 20:57:29   2.50
 --1UhMGODdWsrMastO9DZw      2016-06-04 21:09:10   2.33
 --1UhMGODdWsrMastO9DZw      2016-06-13 19:01:34   2.50
 --1UhMGODdWsrMastO9DZw      2016-06-18 23:39:03   3.00
 --1UhMGODdWsrMastO9DZw      2016-07-16 21:44:33   3.33
 --1UhMGODdWsrMastO9DZw      2016-07-25 23:31:01   3.43
 --1UhMGODdWsrMastO9DZw      2016-08-16 00:27:50   3.50
 --1UhMGODdWsrMastO9DZw      2016-09-07 15:18:25   3.67
 --1UhMGODdWsrMastO9DZw      2016-11-14 20:09:54   3.70
 --1UhMGODdWsrMastO9DZw      2016-12-06 20:04:46   3.80
 --1UhMGODdWsrMastO9DZw      2017-04-03 21:04:06   4.20
 --1UhMGODdWsrMastO9DZw      2017-05-01 02:36:44   4.50
 --1UhMGODdWsrMastO9DZw      2017-05-09 00:32:24   4.60
 --1UhMGODdWsrMastO9DZw      2017-05-09 18:57:16   4.50
 --1UhMGODdWsrMastO9DZw      2017-05-10 17:34:06   4.50
 --1UhMGODdWsrMastO9DZw      2017-06-17 20:06:11   4.60
 --1UhMGODdWsrMastO9DZw      2017-07-26 17:27:49   4.70
 --1UhMGODdWsrMastO9DZw      2017-08-09 18:12:06   4.60
 --1UhMGODdWsrMastO9DZw      2017-08-27 03:06:39   4.60
 --1UhMGODdWsrMastO9DZw      2017-12-12 19:08:18   4.60
 --1UhMGODdWsrMastO9DZw      2017-12-23 00:49:01   4.60
 --1UhMGODdWsrMastO9DZw      2018-01-11 19:55:31   4.60
 --1UhMGODdWsrMastO9DZw      2018-02-25 17:47:12   4.70
 --1UhMGODdWsrMastO9DZw      2018-04-22 17:42:09   4.50
 --1UhMGODdWsrMastO9DZw      2018-05-06 04:22:48   4.10
```

4. Apache HBase Analysis

(1) Insert Data

We can insert the data into HBase using Java API. And check the data table in HBase.
Table Structure:

```
 hbase(main):033:0> describe 'business'
 Table business is ENABLED
```

```
business

COLUMN FAMILIES DESCRIPTION

{NAME => 'info', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'fals
e', KEEP_DELETED_CELLS => 'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE
', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW', CACHE_I
NDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOC
KS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}


{NAME => 'position', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => '
false', KEEP_DELETED_CELLS => 'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => '
NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW', CAC
HE_INDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_
BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}


2 row(s)

QUOTAS

0 row(s)
Took 0.2437 seconds
hbase(main):034:0> describe 'review'
Table review is ENABLED

review

COLUMN FAMILIES DESCRIPTION

{NAME => 'info', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'fals
e', KEEP_DELETED_CELLS => 'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE
', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW', CACHE_I
NDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOC
KS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}


{NAME => 'vote', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'fals
e', KEEP_DELETED_CELLS => 'FALSE', CACHE_DATA_ON_WRITE => 'false', DATA_BLOCK_ENCODING => 'NONE
', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW', CACHE_I
NDEX_ON_WRITE => 'false', IN_MEMORY => 'false', CACHE_BLOOMS_ON_WRITE => 'false', PREFETCH_BLOC
KS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}


2 row(s)

QUOTAS

0 row(s)
Took 0.0980 seconds
```

(2) MapReduce – Numbers of Businesses in Each City and State

The MapReduce Algorithm method is the same as the one in Hadoop.

Terminal Command:

```
nohup hadoop jar HBase/target/HBase-1.0-SNAPSHOT.jar Business.NumbersInEachCityAndState.MapReduce
 > shell_logs/hbase_number_of_businesses_each_city.log &
```

Command Log:

```
2019-12-14 00:46:05,869 INFO zookeeper.ZooKeeper: Client environment:zookeeper.version=3.4.13-
2d71af4dbe22557fda74f9a9b4309b15a7487f03, built on 06/29/2018 00:39 GMT
2019-12-14 00:46:05,870 INFO zookeeper.ZooKeeper: Client environment:host.name=localhost
2019-12-14 00:46:05,870 INFO zookeeper.ZooKeeper: Client environment:java.version=1.8.0_181
2019-12-14 00:46:05,870 INFO zookeeper.ZooKeeper: Client environment:java.vendor=Oracle Corporation
2019-12-14 00:46:05,870 INFO zookeeper.ZooKeeper: Client
environment:java.home=/Library/Java/JavaVirtualMachines/jdk1.8.0_181.jdk/Contents/Home/jre
2019-12-14 00:46:05,871 INFO zookeeper.ZooKeeper: Client
environment:java.library.path=/usr/local/Cellar/hadoop/3.2.1/lib/native
```

```
2019-12-14 00:46:05,871 INFO zookeeper.ZooKeeper: Client
environment:java.io.tmpdir=/var/folders/87/2sdw4zvx7s51ll9p6tw4clfh0000gn/T/
2019-12-14 00:46:05,871 INFO zookeeper.ZooKeeper: Client environment:java.compiler=<NA>
2019-12-14 00:46:05,871 INFO zookeeper.ZooKeeper: Client environment:os.name=Mac OS X
2019-12-14 00:46:05,871 INFO zookeeper.ZooKeeper: Client environment:os.arch=x86_64
2019-12-14 00:46:05,871 INFO zookeeper.ZooKeeper: Client environment:os.version=10.15.1
2019-12-14 00:46:05,871 INFO zookeeper.ZooKeeper: Client environment:user.name=kinyang
2019-12-14 00:46:05,871 INFO zookeeper.ZooKeeper: Client environment:user.home=/Users/kinyang
2019-12-14 00:46:05,871 INFO zookeeper.ZooKeeper: Client
environment:user.dir=/Users/kinyang/GitHub/INFO_7250/Final Project
2019-12-14 00:46:05,873 INFO zookeeper.ZooKeeper: Initiating client connection,
connectString=localhost:2181 sessionTimeout=90000
watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$12/1567887502@2e1d500
2019-12-14 00:46:05,890 INFO zookeeper.ClientCnxn: Opening socket connection to server
localhost/0:0:0:0:0:0:0:1:2181. Will not attempt to authenticate using SASL (unknown error)
2019-12-14 00:46:05,906 INFO zookeeper.ClientCnxn: Socket connection established to
localhost/0:0:0:0:0:0:0:1:2181, initiating session
2019-12-14 00:46:05,914 INFO zookeeper.ClientCnxn: Session establishment complete on server
localhost/0:0:0:0:0:0:0:1:2181, sessionid = 0x1000d8642be003c, negotiated timeout = 40000
2019-12-14 00:46:06,599 WARN client.ConnectionImplementation: Table numbers_in_each_city_and_state
does not exist
2019-12-14 00:46:08,024 INFO client.HBaseAdmin: Operation: CREATE, Table Name:
default:numbers_in_each_city_and_state, procId: 29 completed
2019-12-14 00:46:08,025 INFO client.ConnectionImplementation: Closing master protocol:
MasterService
2019-12-14 00:46:08,029 INFO zookeeper.ZooKeeper: Session: 0x1000d8642be003c closed
2019-12-14 00:46:08,029 INFO zookeeper.ClientCnxn: EventThread shut down for session:
0x1000d8642be003c
2019-12-14 00:46:08,178 INFO Configuration.deprecation: io.bytes.per.checksum is deprecated.
Instead, use dfs.bytes-per-checksum
2019-12-14 00:46:08,266 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-12-14 00:46:08,794 INFO Configuration.deprecation: io.bytes.per.checksum is deprecated.
Instead, use dfs.bytes-per-checksum
2019-12-14 00:46:08,795 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-
publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
2019-12-14 00:46:08,799 INFO zookeeper.ZooKeeper: Initiating client connection,
connectString=localhost:2181 sessionTimeout=90000
watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$12/1567887502@2e1d500
2019-12-14 00:46:08,800 INFO zookeeper.ClientCnxn: Opening socket connection to server
localhost/127.0.0.1:2181. Will not attempt to authenticate using SASL (unknown error)
2019-12-14 00:46:08,800 INFO zookeeper.ClientCnxn: Socket connection established to
localhost/127.0.0.1:2181, initiating session
2019-12-14 00:46:08,804 INFO zookeeper.ClientCnxn: Session establishment complete on server
localhost/127.0.0.1:2181, sessionid = 0x1000d8642be003d, negotiated timeout = 40000
2019-12-14 00:46:09,112 INFO zookeeper.ZooKeeper: Session: 0x1000d8642be003d closed
2019-12-14 00:46:09,112 INFO zookeeper.ClientCnxn: EventThread shut down for session:
0x1000d8642be003d
2019-12-14 00:46:09,236 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not
performed. Implement the Tool interface and execute your application with ToolRunner to remedy
this.
2019-12-14 00:46:09,252 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path:
/tmp/hadoop-yarn/staging/kinyang/.staging/job_1576235822595_0021
```

```
2019-12-14 00:46:09,348 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:09,446 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:09,465 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:09,888 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:09,906 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:09,921 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:09,938 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:09,956 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:09,969 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:10,005 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:10,097 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:10,120 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:10,135 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:10,557 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:10,575 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:10,597 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:11,018 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:11,034 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:11,050 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:11,068 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:11,097 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:11,534 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:11,549 INFO zookeeper.ZooKeeper: Initiating client connection,
connectString=localhost:2181 sessionTimeout=90000
watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$12/1567887502@2e1d500
2019-12-14 00:46:11,550 INFO zookeeper.ClientCnxn: Opening socket connection to server
localhost/0:0:0:0:0:0:0:1:2181. Will not attempt to authenticate using SASL (unknown error)
2019-12-14 00:46:11,550 INFO zookeeper.ClientCnxn: Socket connection established to
localhost/0:0:0:0:0:0:0:1:2181, initiating session
2019-12-14 00:46:11,553 INFO zookeeper.ClientCnxn: Session establishment complete on server
localhost/0:0:0:0:0:0:0:1:2181, sessionid = 0x1000d8642be003e, negotiated timeout = 40000
```

```
2019-12-14 00:46:11,554 INFO mapreduce.RegionSizeCalculator: Calculating region sizes for table
"business".
2019-12-14 00:46:11,676 INFO zookeeper.ZooKeeper: Session: 0x1000d8642be003e closed
2019-12-14 00:46:11,676 INFO zookeeper.ClientCnxn: EventThread shut down for session:
0x1000d8642be003e
2019-12-14 00:46:11,691 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:11,704 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:12,114 INFO mapreduce.JobSubmitter: number of splits:1
2019-12-14 00:46:12,182 INFO Configuration.deprecation: io.bytes.per.checksum is deprecated.
Instead, use dfs.bytes-per-checksum
2019-12-14 00:46:12,183 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-
publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
2019-12-14 00:46:12,256 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-14 00:46:12,272 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1576235822595_0021
2019-12-14 00:46:12,272 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-12-14 00:46:12,438 INFO conf.Configuration: resource-types.xml not found
2019-12-14 00:46:12,438 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-12-14 00:46:12,501 INFO impl.YarnClientImpl: Submitted application
application_1576235822595_0021
2019-12-14 00:46:12,540 INFO mapreduce.Job: The url to track the job:
http://localhost:8088/proxy/application_1576235822595_0021/
2019-12-14 00:46:12,541 INFO mapreduce.Job: Running job: job_1576235822595_0021
2019-12-14 00:46:19,651 INFO mapreduce.Job: Job job_1576235822595_0021 running in uber mode : false
2019-12-14 00:46:19,652 INFO mapreduce.Job:  map 0% reduce 0%
2019-12-14 00:46:27,732 INFO mapreduce.Job:  map 100% reduce 0%
2019-12-14 00:46:34,803 INFO mapreduce.Job:  map 100% reduce 100%
2019-12-14 00:46:34,812 INFO mapreduce.Job: Job job_1576235822595_0021 completed successfully
2019-12-14 00:46:34,889 INFO mapreduce.Job: Counters: 63
    File System Counters
        FILE: Number of bytes read=3533244
        FILE: Number of bytes written=7592831
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=142
        HDFS: Number of bytes written=0
        HDFS: Number of read operations=1
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=0
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Launched map tasks=1
        Launched reduce tasks=1
        Data-local map tasks=1
        Total time spent by all maps in occupied slots (ms)=5928
        Total time spent by all reduces in occupied slots (ms)=3590
        Total time spent by all map tasks (ms)=5928
        Total time spent by all reduce tasks (ms)=3590
        Total vcore-milliseconds taken by all map tasks=5928
        Total vcore-milliseconds taken by all reduce tasks=3590
        Total megabyte-milliseconds taken by all map tasks=6070272
        Total megabyte-milliseconds taken by all reduce tasks=3676160
    Map-Reduce Framework
        Map input records=192609
        Map output records=192609
        Map output bytes=3148020
        Map output materialized bytes=3533244
        Input split bytes=142
        Combine input records=0
        Combine output records=0
        Reduce input groups=1258
        Reduce shuffle bytes=3533244
        Reduce input records=192609
        Reduce output records=1258
        Spilled Records=385218
        Shuffled Maps =1
        Failed Shuffles=0
        Merged Map outputs=1
        GC time elapsed (ms)=214
        CPU time spent (ms)=0
        Physical memory (bytes) snapshot=0
        Virtual memory (bytes) snapshot=0
        Total committed heap usage (bytes)=774897664
    HBaseCounters
        BYTES_IN_REMOTE_RESULTS=0
        BYTES_IN_RESULTS=131501399
        MILLIS_BETWEEN_NEXTS=2728
        NOT_SERVING_REGION_EXCEPTION=0
        NUM_SCANNER_RESTARTS=0
        NUM_SCAN_RESULTS_STALE=0
        REGIONS_SCANNED=1
        REMOTE_RPC_CALLS=0
        REMOTE_RPC_RETRIES=0
        ROWS_FILTERED=0
        ROWS_SCANNED=192609
        RPC_CALLS=387
        RPC_RETRIES=0
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=0
    File Output Format Counters
        Bytes Written=0
HBase Numbers In Each City And State MapReduce Time: 26692 ms
```

MapReduce Table Result:

```
hbase(main):035:0> scan 'numbers_in_each_city_and_state'
 ROW                COLUMN+CELL
 \x00\x00\x00\x01 column=position:city, timestamp=1576302392858, value=Airdrie
 \x00\x00\x00\x01 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x01 column=results:count, timestamp=1576302392858, value=168
 \x00\x00\x00\x02 column=position:city, timestamp=1576302392858, value=Alberta
 \x00\x00\x00\x02 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x02 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x03 column=position:city, timestamp=1576302392858, value=Balzac
 \x00\x00\x00\x03 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x03 column=results:count, timestamp=1576302392858, value=11
 \x00\x00\x00\x04 column=position:city, timestamp=1576302392858, value=Beltline
 \x00\x00\x00\x04 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x04 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x05 column=position:city, timestamp=1576302392858, value=CALGARY
 \x00\x00\x00\x05 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x05 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x06 column=position:city, timestamp=1576302392858, value=Calgary
 \x00\x00\x00\x06 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x06 column=results:count, timestamp=1576302392858, value=7735
 \x00\x00\x00\x07 column=position:city, timestamp=1576302392858, value=Chestermere
 \x00\x00\x00\x07 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x07 column=results:count, timestamp=1576302392858, value=31
 \x00\x00\x00\x08 column=position:city, timestamp=1576302392858, value=De Winton
 \x00\x00\x00\x08 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x08 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x09 column=position:city, timestamp=1576302392858, value=Division No. 6
 \x00\x00\x00\x09 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x09 column=results:count, timestamp=1576302392858, value=3
 \x00\x00\x00\x0A column=position:city, timestamp=1576302392858, value=Downtown
 \x00\x00\x00\x0A column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x0A column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x0B column=position:city, timestamp=1576302392858, value=East Calgary
 \x00\x00\x00\x0B column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x0B column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x0C column=position:city, timestamp=1576302392858, value=Edgemont
 \x00\x00\x00\x0C column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x0C column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x0D column=position:city, timestamp=1576302392858, value=Edmonton
 \x00\x00\x00\x0D column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x0D column=results:count, timestamp=1576302392858, value=2
 \x00\x00\x00\x0E column=position:city, timestamp=1576302392858, value=Evergreen
 \x00\x00\x00\x0E column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x0E column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x0F column=position:city, timestamp=1576302392858, value=Highland Park
 \x00\x00\x00\x0F column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x0F column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x10 column=position:city, timestamp=1576302392858, value=Medicine Hat
 \x00\x00\x00\x10 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x10 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x11 column=position:city, timestamp=1576302392858, value=Midnapore
 \x00\x00\x00\x11 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x11 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x12 column=position:city, timestamp=1576302392858, value=Montreal
 \x00\x00\x00\x12 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x12 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x13 column=position:city, timestamp=1576302392858, value=North York
 \x00\x00\x00\x13 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x13 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x14 column=position:city, timestamp=1576302392858, value=Northeast Calgary
 \x00\x00\x00\x14 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x14 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x15 column=position:city, timestamp=1576302392858, value=Northwest Calgary
 \x00\x00\x00\x15 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x15 column=results:count, timestamp=1576302392858, value=3
 \x00\x00\x00\x16 column=position:city, timestamp=1576302392858, value=Rockey View
 \x00\x00\x00\x16 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x16 column=results:count, timestamp=1576302392858, value=1
 \x00\x00\x00\x17 column=position:city, timestamp=1576302392858, value=Rocky View
 \x00\x00\x00\x17 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x17 column=results:count, timestamp=1576302392858, value=20
 \x00\x00\x00\x18 column=position:city, timestamp=1576302392858, value=Rocky View County
 \x00\x00\x00\x18 column=position:state, timestamp=1576302392858, value=AB
 \x00\x00\x00\x18 column=results:count, timestamp=1576302392858, value=9
 \x00\x00\x00\x19 column=position:city, timestamp=1576302392858, value=Rocky View No. 44
```

```
  \x00\x00\x00\x19 column=position:state, timestamp=1576302392858, value=AB
  \x00\x00\x00\x19 column=results:count, timestamp=1576302392858, value=4
  \x00\x00\x00\x1A column=position:city, timestamp=1576302392858, value=Rockyview
  \x00\x00\x00\x1A column=position:state, timestamp=1576302392858, value=AB
  \x00\x00\x00\x1A column=results:count, timestamp=1576302392858, value=2
  \x00\x00\x00\x1B column=position:city, timestamp=1576302392858, value=Rockyview County
  \x00\x00\x00\x1B column=position:state, timestamp=1576302392858, value=AB
  \x00\x00\x00\x1B column=results:count, timestamp=1576302392858, value=1
  \x00\x00\x00\x1C column=position:city, timestamp=1576302392858, value=SW Calgary
  \x00\x00\x00\x1C column=position:state, timestamp=1576302392858, value=AB
  \x00\x00\x00\x1C column=results:count, timestamp=1576302392858, value=1
  \x00\x00\x00\x1D column=position:city, timestamp=1576302392858, value=Sage Hill
  \x00\x00\x00\x1D column=position:state, timestamp=1576302392858, value=AB
  \x00\x00\x00\x1D column=results:count, timestamp=1576302392858, value=1
  \x00\x00\x00\x1E column=position:city, timestamp=1576302392858, value=Sainte-Ad\xC3\xA8le
  \x00\x00\x00\x1E column=position:state, timestamp=1576302392858, value=AB
  \x00\x00\x00\x1E column=results:count, timestamp=1576302392858, value=1
  \x00\x00\x00\x1F column=position:city, timestamp=1576302392858, value=Southeast Calgary
  \x00\x00\x00\x1F column=position:state, timestamp=1576302392858, value=AB
  \x00\x00\x00\x1F column=results:count, timestamp=1576302392858, value=3
  \x00\x00\x00     column=position:city, timestamp=1576302392858, value=Toronto
  \x00\x00\x00     column=position:state, timestamp=1576302392858, value=AB
  \x00\x00\x00     column=results:count, timestamp=1576302392858, value=1
  \x00\x00\x00!    column=position:city, timestamp=1576302392858, value=calgary
  \x00\x00\x00!    column=position:state, timestamp=1576302392858, value=AB
  \x00\x00\x00!    column=results:count, timestamp=1576302392858, value=1
 (rest of the rows omitted)
 1258 row(s)
 Took 1.3093 seconds
```

(3) MapReduce – Percentage of Ratings

The MapReduce Algorithm method is the same as the one in Hadoop.

Terminal Command:

```
 nohup hadoop jar HBase/target/HBase-1.0-SNAPSHOT.jar Review.PercentageOfRatings.MapReduce > shell
 _logs/hbase_percentage_of_ratings.log &
```

Command Log:

```
2019-12-13 04:34:52,806 INFO zookeeper.ZooKeeper: Client environment:zookeeper.version=3.4.13-
2d71af4dbe22557fda74f9a9b4309b15a7487f03, built on 06/29/2018 00:39 GMT
2019-12-13 04:34:52,806 INFO zookeeper.ZooKeeper: Client environment:host.name=localhost
2019-12-13 04:34:52,806 INFO zookeeper.ZooKeeper: Client environment:java.version=1.8.0_181
2019-12-13 04:34:52,806 INFO zookeeper.ZooKeeper: Client environment:java.vendor=Oracle Corporation
2019-12-13 04:34:52,806 INFO zookeeper.ZooKeeper: Client
environment:java.home=/Library/Java/JavaVirtualMachines/jdk1.8.0_181.jdk/Contents/Home/jre
2019-12-13 04:34:52,807 INFO zookeeper.ZooKeeper: Client
environment:java.library.path=/usr/local/Cellar/hadoop/3.2.1/lib/native
2019-12-13 04:34:52,807 INFO zookeeper.ZooKeeper: Client
environment:java.io.tmpdir=/var/folders/87/2sdw4zvx7s51ll9p6tw4clfh0000gn/T/
2019-12-13 04:34:52,807 INFO zookeeper.ZooKeeper: Client environment:java.compiler=<NA>
2019-12-13 04:34:52,807 INFO zookeeper.ZooKeeper: Client environment:os.name=Mac OS X
2019-12-13 04:34:52,807 INFO zookeeper.ZooKeeper: Client environment:os.arch=x86_64
2019-12-13 04:34:52,807 INFO zookeeper.ZooKeeper: Client environment:os.version=10.15.1
2019-12-13 04:34:52,807 INFO zookeeper.ZooKeeper: Client environment:user.name=kinyang
2019-12-13 04:34:52,807 INFO zookeeper.ZooKeeper: Client environment:user.home=/Users/kinyang
2019-12-13 04:34:52,807 INFO zookeeper.ZooKeeper: Client
environment:user.dir=/Users/kinyang/GitHub/INFO_7250/Final Project
2019-12-13 04:34:52,808 INFO zookeeper.ZooKeeper: Initiating client connection,
connectString=localhost:2181 sessionTimeout=90000
watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$12/1989169051@4c7f113c
2019-12-13 04:34:52,822 INFO zookeeper.ClientCnxn: Opening socket connection to server
localhost/0:0:0:0:0:0:0:1:2181. Will not attempt to authenticate using SASL (unknown error)
2019-12-13 04:34:52,837 INFO zookeeper.ClientCnxn: Socket connection established to
localhost/0:0:0:0:0:0:0:1:2181, initiating session
```

```
2019-12-13 04:34:52,852 INFO zookeeper.ClientCnxn: Session establishment complete on server
localhost/0:0:0:0:0:0:0:1:2181, sessionid = 0x1000d16cedc000b, negotiated timeout = 40000
2019-12-13 04:34:54,086 INFO zookeeper.ZooKeeper: Session: 0x1000d16cedc000b closed
2019-12-13 04:34:54,088 INFO zookeeper.ClientCnxn: EventThread shut down for session:
0x1000d16cedc000b
2019-12-13 04:34:54,274 INFO Configuration.deprecation: io.bytes.per.checksum is deprecated.
Instead, use dfs.bytes-per-checksum
2019-12-13 04:34:54,375 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2019-12-13 04:34:55,114 INFO Configuration.deprecation: io.bytes.per.checksum is deprecated.
Instead, use dfs.bytes-per-checksum
2019-12-13 04:34:55,115 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-
publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
2019-12-13 04:34:55,118 INFO zookeeper.ZooKeeper: Initiating client connection,
connectString=localhost:2181 sessionTimeout=90000
watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$12/1989169051@4c7f113c
2019-12-13 04:34:55,119 INFO zookeeper.ClientCnxn: Opening socket connection to server
localhost/0:0:0:0:0:0:0:1:2181. Will not attempt to authenticate using SASL (unknown error)
2019-12-13 04:34:55,119 INFO zookeeper.ClientCnxn: Socket connection established to
localhost/0:0:0:0:0:0:0:1:2181, initiating session
2019-12-13 04:34:55,122 INFO zookeeper.ClientCnxn: Session establishment complete on server
localhost/0:0:0:0:0:0:0:1:2181, sessionid = 0x1000d16cedc000c, negotiated timeout = 40000
2019-12-13 04:34:55,528 INFO zookeeper.ZooKeeper: Session: 0x1000d16cedc000c closed
2019-12-13 04:34:55,528 INFO zookeeper.ClientCnxn: EventThread shut down for session:
0x1000d16cedc000c
2019-12-13 04:34:55,590 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not
performed. Implement the Tool interface and execute your application with ToolRunner to remedy
this.
2019-12-13 04:34:55,605 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path:
/tmp/hadoop-yarn/staging/kinyang/.staging/job_1576224793999_0012
2019-12-13 04:34:55,753 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:55,873 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:55,944 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:55,964 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:55,983 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,002 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,023 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,074 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,092 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,210 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,429 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,454 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
```

```
2019-12-13 04:34:56,474 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,548 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,576 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,687 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,710 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,723 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,811 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,822 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,858 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,914 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:56,929 INFO zookeeper.ZooKeeper: Initiating client connection,
connectString=localhost:2181 sessionTimeout=90000
watcher=org.apache.hadoop.hbase.zookeeper.ReadOnlyZKClient$$Lambda$12/1989169051@4c7f113c
2019-12-13 04:34:56,930 INFO zookeeper.ClientCnxn: Opening socket connection to server
localhost/127.0.0.1:2181. Will not attempt to authenticate using SASL (unknown error)
2019-12-13 04:34:56,930 INFO zookeeper.ClientCnxn: Socket connection established to
localhost/127.0.0.1:2181, initiating session
2019-12-13 04:34:56,931 INFO zookeeper.ClientCnxn: Session establishment complete on server
localhost/127.0.0.1:2181, sessionid = 0x1000d16cedc000d, negotiated timeout = 40000
2019-12-13 04:34:56,933 INFO mapreduce.RegionSizeCalculator: Calculating region sizes for table
"review".
2019-12-13 04:34:57,036 INFO zookeeper.ZooKeeper: Session: 0x1000d16cedc000d closed
2019-12-13 04:34:57,036 INFO zookeeper.ClientCnxn: EventThread shut down for session:
0x1000d16cedc000d
2019-12-13 04:34:57,049 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:57,059 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:57,468 INFO mapreduce.JobSubmitter: number of splits:2
2019-12-13 04:34:57,537 INFO Configuration.deprecation: io.bytes.per.checksum is deprecated.
Instead, use dfs.bytes-per-checksum
2019-12-13 04:34:57,537 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-
publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
2019-12-13 04:34:57,610 INFO sasl.SaslDataTransferClient: SASL encryption trust check:
localHostTrusted = false, remoteHostTrusted = false
2019-12-13 04:34:57,623 INFO mapreduce.JobSubmitter: Submitting tokens for job:
job_1576224793999_0012
2019-12-13 04:34:57,624 INFO mapreduce.JobSubmitter: Executing with tokens: []
2019-12-13 04:34:57,794 INFO conf.Configuration: resource-types.xml not found
2019-12-13 04:34:57,794 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2019-12-13 04:34:57,853 INFO impl.YarnClientImpl: Submitted application
application_1576224793999_0012
```

```
2019-12-13 04:34:57,891 INFO mapreduce.Job: The url to track the job:
http://localhost:8088/proxy/application_1576224793999_0012/
2019-12-13 04:34:57,892 INFO mapreduce.Job: Running job: job_1576224793999_0012
2019-12-13 04:35:04,987 INFO mapreduce.Job: Job job_1576224793999_0012 running in uber mode : false
2019-12-13 04:35:04,990 INFO mapreduce.Job:  map 0% reduce 0%
2019-12-13 04:35:40,320 INFO mapreduce.Job:  map 100% reduce 0%
2019-12-13 04:35:53,435 INFO mapreduce.Job:  map 100% reduce 100%
2019-12-13 04:35:53,442 INFO mapreduce.Job: Job job_1576224793999_0012 completed successfully
2019-12-13 04:35:53,592 INFO mapreduce.Job: Counters: 63
    File System Counters
        FILE: Number of bytes read=187205230
        FILE: Number of bytes written=281597276
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=324
        HDFS: Number of bytes written=0
        HDFS: Number of read operations=2
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=0
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Launched map tasks=2
        Launched reduce tasks=1
        Data-local map tasks=2
        Total time spent by all maps in occupied slots (ms)=64857
        Total time spent by all reduces in occupied slots (ms)=10676
        Total time spent by all map tasks (ms)=64857
        Total time spent by all reduce tasks (ms)=10676
        Total vcore-milliseconds taken by all map tasks=64857
        Total vcore-milliseconds taken by all reduce tasks=10676
        Total megabyte-milliseconds taken by all map tasks=66413568
        Total megabyte-milliseconds taken by all reduce tasks=10932224
    Map-Reduce Framework
        Map input records=6685900
        Map output records=6685900
        Map output bytes=80230800
        Map output materialized bytes=93602612
        Input split bytes=324
        Combine input records=0
        Combine output records=0
        Reduce input groups=5
        Reduce shuffle bytes=93602612
        Reduce input records=6685900
        Reduce output records=5
        Spilled Records=20057700
        Shuffled Maps =2
        Failed Shuffles=0
        Merged Map outputs=2
        GC time elapsed (ms)=920
        CPU time spent (ms)=0
        Physical memory (bytes) snapshot=0
        Virtual memory (bytes) snapshot=0
        Total committed heap usage (bytes)=1297088512
    HBaseCounters
        BYTES_IN_REMOTE_RESULTS=0
        BYTES_IN_RESULTS=3169116600
        MILLIS_BETWEEN_NEXTS=53110
        NOT_SERVING_REGION_EXCEPTION=0
        NUM_SCANNER_RESTARTS=0
        NUM_SCAN_RESULTS_STALE=0
        REGIONS_SCANNED=2
        REMOTE_RPC_CALLS=0
        REMOTE_RPC_RETRIES=0
        ROWS_FILTERED=0
        ROWS_SCANNED=6685900
        RPC_CALLS=13374
        RPC_RETRIES=0
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=0
    File Output Format Counters
        Bytes Written=0
HBase Percentage of Ratings MapReduce Time: 59297 ms
```

MapReduce Table Result:

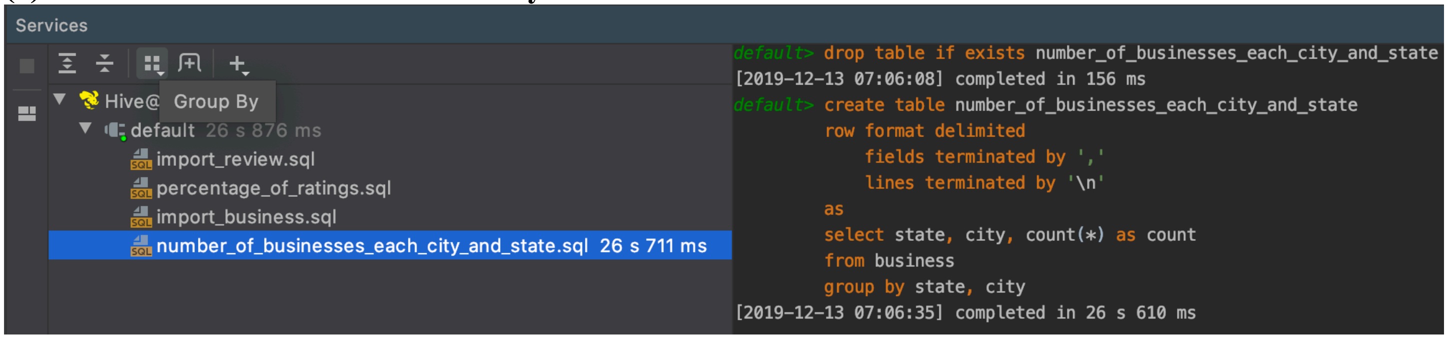
```
hbase(main):036:0> scan 'percentage_of_ratings'
ROW    COLUMN+CELL
1.0    column=results:percentage, timestamp=1576229751820, value=14.989141327270824
2.0    column=results:percentage, timestamp=1576229751820, value=8.11250542185794
3.0    column=results:percentage, timestamp=1576229751820, value=11.057299690393215
4.0    column=results:percentage, timestamp=1576229751820, value=21.971387546927115
5.0    column=results:percentage, timestamp=1576229751820, value=43.869666013550905
5 row(s)
Took 0.0267 seconds
```

5. Apache Hive Analysis

(1) Import into Hive

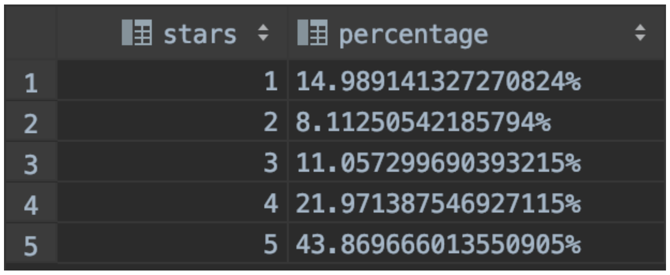
Because the dataset is json format, we can use the ***JsonSerDe*** class to help us to import json data into Hive.
The table structure and HiveQL Code is at Appendix Part.

(2) Numbers of Businesses in Each City and State


Services
Hive@ Group By
default 26 s 876 ms
import_review.sql
percentage_of_ratings.sql
import_business.sql
number_of_businesses_each_city_and_state.sql 26 s 711 ms

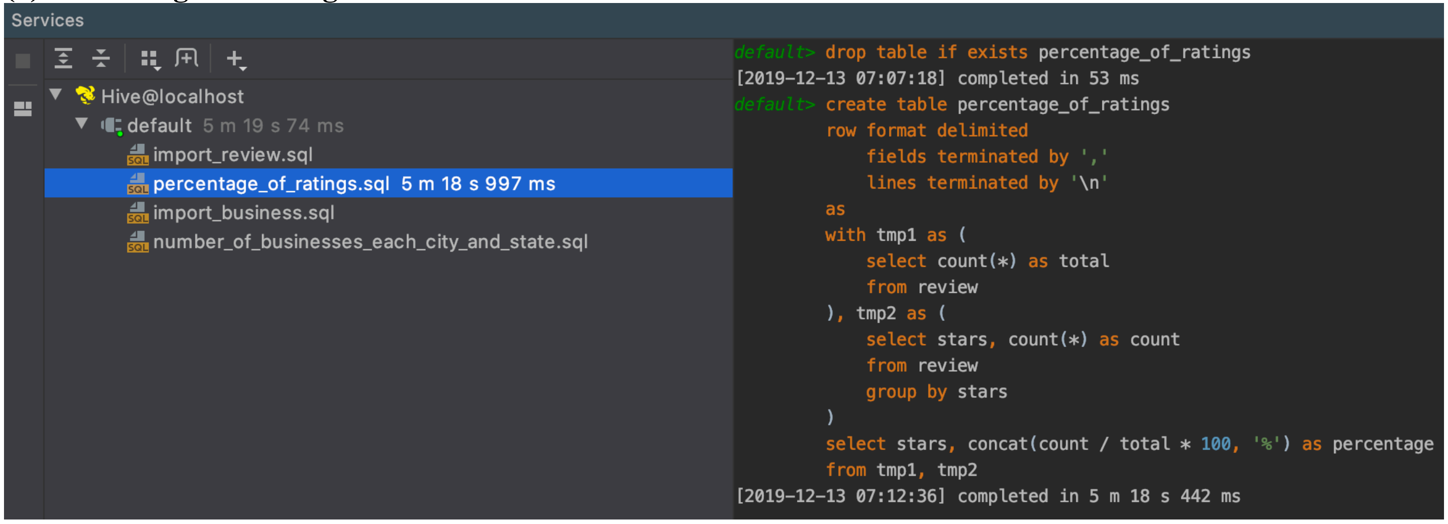
```
default> drop table if exists number_of_businesses_each_city_and_state
[2019-12-13 07:06:08] completed in 156 ms
default> create table number_of_businesses_each_city_and_state
         row format delimited
             fields terminated by ','
             lines terminated by '\n'
         as
         select state, city, count(*) as count
         from business
         group by state, city
[2019-12-13 07:06:35] completed in 26 s 610 ms
```

Table Result:



	stars	percentage
1	1	14.989141327270824%
2	2	8.11250542185794%
3	3	11.057299690393215%
4	4	21.971387546927115%
5	5	43.869666013550905%

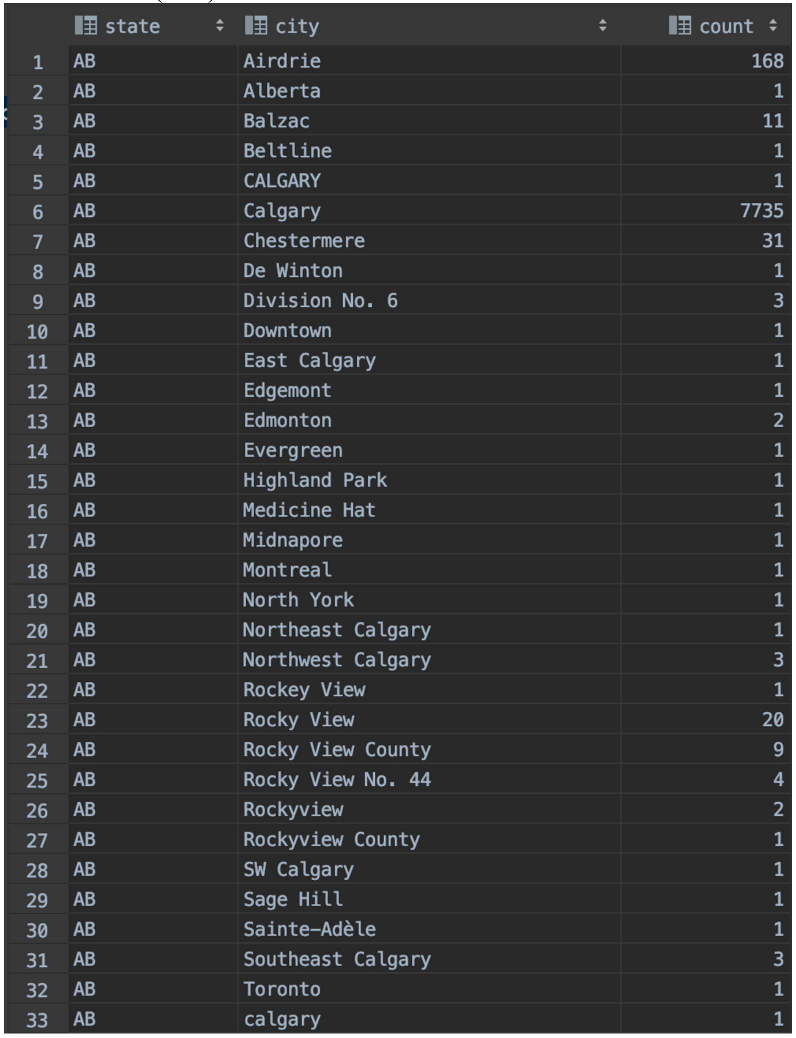
(3) Percentage of Ratings



Services
Hive@localhost
default 5 m 19 s 74 ms
import_review.sql
percentage_of_ratings.sql 5 m 18 s 997 ms
import_business.sql
number_of_businesses_each_city_and_state.sql

```
default> drop table if exists percentage_of_ratings
[2019-12-13 07:07:18] completed in 53 ms
default> create table percentage_of_ratings
         row format delimited
             fields terminated by ','
             lines terminated by '\n'
         as
         with tmp1 as (
             select count(*) as total
             from review
         ), tmp2 as (
             select stars, count(*) as count
             from review
             group by stars
         )
         select stars, concat(count / total * 100, '%') as percentage
         from tmp1, tmp2
[2019-12-13 07:12:36] completed in 5 m 18 s 442 ms
```

Table Result (Part):



	state	city	count
1	AB	Airdrie	168
2	AB	Alberta	1
3	AB	Balzac	11
4	AB	Beltline	1
5	AB	CALGARY	1
6	AB	Calgary	7735
7	AB	Chestermere	31
8	AB	De Winton	1
9	AB	Division No. 6	3
10	AB	Downtown	1
11	AB	East Calgary	1
12	AB	Edgemont	1
13	AB	Edmonton	2
14	AB	Evergreen	1
15	AB	Highland Park	1
16	AB	Medicine Hat	1
17	AB	Midnapore	1
18	AB	Montreal	1
19	AB	North York	1
20	AB	Northeast Calgary	1
21	AB	Northwest Calgary	3
22	AB	Rockey View	1
23	AB	Rocky View	20
24	AB	Rocky View County	9
25	AB	Rocky View No. 44	4
26	AB	Rockyview	2
27	AB	Rockyview County	1
28	AB	SW Calgary	1
29	AB	Sage Hill	1
30	AB	Sainte-Adèle	1
31	AB	Southeast Calgary	3
32	AB	Toronto	1
33	AB	calgary	1

6. Some Conclusions

According to the above analysis, we can check the running time of the same MapReduce running on different frameworks and do some analysis too. Here is the result.

	Numbers of Businesses in Each City and State	Percentage of Ratings
Apache Hadoop	20980 ms	147569 ms
Apache HBase	26692 ms	59297 ms
Apache Hive	26 s 711 ms = 26711 ms	5 min 18 s 997 ms = 3018997 ms

According to the result, we can see that the first MapReduce performs almost the same between HBase and Hive, but a little faster in Hadoop. But the second MapReduce performance is really different. HBase performs about 2.5x faster than Hadoop and Hive runs really slow in this analysis.

APPENDIX

1. Hadoop

(1) MapReduce – Numbers of Businesses in Each City and State

1) StateCityKeyWritableComparable.java

```java
package Business.NumersInEachCityAndState;

import org.apache.hadoop.io.WritableComparable;

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

public class StateCityKeyWritableComparable implements
WritableComparable<StateCityKeyWritableComparable> {

    private String state;
    private String city;

    public StateCityKeyWritableComparable() {
    }

    public StateCityKeyWritableComparable(String state, String city) {
        this.state = state;
        this.city = city;
    }

    @Override
    public int compareTo(StateCityKeyWritableComparable o) {
        if (state.compareTo(o.state) != 0) {
            return state.compareTo(o.state);
        }
        return city.compareTo(o.city);
    }

    @Override
    public void write(DataOutput dataOutput) throws IOException {
        dataOutput.writeUTF(state);
        dataOutput.writeUTF(city);
    }

    @Override
    public void readFields(DataInput dataInput) throws IOException {
        state = dataInput.readUTF();
        city = dataInput.readUTF();
    }

    public String getState() {
        return state;
    }

    public void setState(String state) {
        this.state = state;
    }

    public String getCity() {
        return city;
    }

    public void setCity(String city) {
        this.city = city;
    }

    @Override
    public String toString() {
        return state + "\t" + city;
    }
}
```

2) StateCityMapper.java

```java
package Business.NumersInEachCityAndState;

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

import java.io.IOException;

public class StateCityMapper extends Mapper<LongWritable, Text, StateCityKeyWritableComparable,
IntWritable> {

    private IntWritable one = new IntWritable(1);

    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
        String jsonStr = value.toString();
        JSONObject object = new JSONObject(jsonStr);

        String city = object.getString("city");
        String state = object.getString("state");
        StateCityKeyWritableComparable stateCityKeyWritableComparable = new
StateCityKeyWritableComparable(state, city);

        context.write(stateCityKeyWritableComparable, one);
    }
}
```

3) StateCityReducer.java

```java
package Business.NumersInEachCityAndState;

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

import java.io.*;

public class StateCityReducer extends Reducer<StateCityKeyWritableComparable, IntWritable,
StateCityKeyWritableComparable, IntWritable> {

    private IntWritable result = new IntWritable();

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

4) MapReduce.java

```java
package Business.NumersInEachCityAndState;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
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;

public class MapReduce {
    public static void main(String[] args) throws IOException, InterruptedException,
ClassNotFoundException {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Number of Businesses in each city and state");

        job.setJarByClass(MapReduce.class);

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

        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);

        job.setMapperClass(StateCityMapper.class);
        job.setReducerClass(StateCityReducer.class);

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

        FileSystem fs = FileSystem.get(conf);
        fs.delete(new Path(args[1]), true);

        long startTime = System.currentTimeMillis();
        boolean result = job.waitForCompletion(true);
        long endTime = System.currentTimeMillis();
        System.out.println("Hadoop Number of Businesses in each city and state MapReduce Time: " +
(endTime - startTime) + " ms");

        System.exit(result ? 0 : 1);
    }
}
```

(2) MapReduce – Top 5 Rated Business in Each State

1) BusinessWritable.java

```java
package Business.Top5RatedBusinessInEachState;

import org.apache.hadoop.io.WritableComparable;

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

public class BusinessWritable implements WritableComparable<BusinessWritable> {

    private String businessId;
    private String businessName;
    private String state;
    private double rate;

    public BusinessWritable() {
    }

    public BusinessWritable(String businessId, String businessName, String state, double rate) {
        this.businessId = businessId;
        this.businessName = businessName;
        this.state = state;
        this.rate = rate;
    }

    @Override
    public int compareTo(BusinessWritable o) {
        if (rate != o.getRate()) {
            return o.getRate() > rate ? 1 : -1;
        }
        return businessName.compareTo(o.getBusinessName());
    }

    @Override
    public void write(DataOutput dataOutput) throws IOException {
        dataOutput.writeUTF(businessId);
        dataOutput.writeUTF(businessName);
        dataOutput.writeUTF(state);
        dataOutput.writeDouble(rate);
    }

    @Override
    public void readFields(DataInput dataInput) throws IOException {
        businessId = dataInput.readUTF();
        businessName = dataInput.readUTF();
        state = dataInput.readUTF();
        rate = dataInput.readDouble();
    }

    public String getBusinessId() {
        return businessId;
    }

    public void setBusinessId(String businessId) {
        this.businessId = businessId;
    }

    public String getBusinessName() {
        return businessName;
    }

    public void setBusinessName(String businessName) {
        this.businessName = businessName;
    }

    public String getState() {
        return state;
    }

    public void setState(String state) {
        this.state = state;
    }

    public double getRate() {
        return rate;
    }

    public void setRate(double rate) {
        this.rate = rate;
    }

    @Override
    public String toString() {
        return "{businessId: '" + businessId + '\'' +
                ", businessName: '" + businessName + '\'' +
                ", state: '" + state + '\'' +
                ", rate: " + rate +
                '}';
    }
}
```

2) StateTop5Mapper.java

```java
package Business.Top5RatedBusinessInEachState;

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


public class StateTop5Mapper extends Mapper<LongWritable, Text, Text, BusinessWritable> {

    private HashMap<String, TreeSet<BusinessWritable>> tmap;

    @Override
    public void setup(Context context) {
        tmap = new HashMap<>();
    }

    @Override
    public void map(LongWritable key, Text value, Context context) {
        String jsonStr = value.toString();
        JSONObject object = new JSONObject(jsonStr);

        String businessId = object.getString("business_id");
        String businessName = object.getString("name");
        String state = object.getString("state");
        double rate = object.getDouble("stars");
        BusinessWritable businessWritable = new BusinessWritable(businessId, businessName, state,
rate);

        TreeSet<BusinessWritable> top5set = tmap.getOrDefault(state, new TreeSet<>());
        top5set.add(businessWritable);

        if (top5set.size() > 5) {
            top5set.remove(top5set.last());
        }

        tmap.put(state, top5set);
    }

    @Override
    public void cleanup(Context context) throws IOException, InterruptedException {
        for (Map.Entry<String, TreeSet<BusinessWritable>> e : tmap.entrySet()) {
            TreeSet<BusinessWritable> top5set = e.getValue();
            for (BusinessWritable businessWritable: top5set) {
                Text key = new Text(e.getKey());
                context.write(key, businessWritable);
            }
        }
    }

}
```

3) StateTop5Reducer.java

```java
package Business.Top5RatedBusinessInEachState;

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

import java.io.IOException;

public class StateTop5Reducer extends Reducer<Text, BusinessWritable, BusinessWritable,
NullWritable> {
    @Override
    public void reduce(Text key, Iterable<BusinessWritable> values, Context context) throws
IOException, InterruptedException {
        for (BusinessWritable value : values) {
            context.write(value, NullWritable.get());
        }
    }
}
```

4) MapReduce.java

```java
package Business.Top5RatedBusinessInEachState;

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;
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;

public class MapReduce {
    public static void main(String[] args) throws IOException, InterruptedException,
ClassNotFoundException {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf,  "Top 5 rated businesses in each state");

        job.setJarByClass(MapReduce.class);

        job.setMapperClass(StateTop5Mapper.class);
        job.setReducerClass(StateTop5Reducer.class);

        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);

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

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

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

        FileSystem fs = FileSystem.get(conf);
        fs.delete(new Path(args[1]), true);

        long startTime = System.currentTimeMillis();
        boolean result = job.waitForCompletion(true);
        long endTime = System.currentTimeMillis();
        System.out.println("Hadoop Top 5 rated businesses in each state MapReduce Time: " +
(endTime - startTime) + " ms");

        System.exit(result ? 0 : 1);
    }
}
```

(3) MapReduce – Percentage of Ratings

1) RateMapper.java

```java
package Review.PercentageOfRatings;

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

import java.io.IOException;

public class RateMapper extends Mapper<LongWritable, Text, DoubleWritable, IntWritable> {
    private IntWritable one = new IntWritable(1);
    private DoubleWritable rate = new DoubleWritable();

    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
        String jsonStr = value.toString();
        JSONObject object = new JSONObject(jsonStr);
        double stars = object.getDouble("stars");
        rate.set(stars);
        context.write(rate, one);
    }

}
```

2) RateReducer.java

```java
package Review.PercentageOfRatings;

import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.mapreduce.Reducer;

import java.io.IOException;
import java.util.*;

public class RateReducer extends Reducer<DoubleWritable, IntWritable, DoubleWritable,
DoubleWritable> {

    private TreeMap<Double, Integer> ratings;
    private int total;

    @Override
    public void setup(Context context) {
        ratings = new TreeMap<>();
        total = 0;
    }

    @Override
    public void reduce(DoubleWritable key, Iterable<IntWritable> values, Context context) {
        double rate = key.get();
        int sum = ratings.getOrDefault(rate, 0);
        for (IntWritable value : values) {
            sum += value.get();
            total += value.get();
        }
        ratings.put(rate, sum);
    }

    @Override
    public void cleanup(Context context) throws IOException, InterruptedException {
        for (Map.Entry<Double, Integer> entry : ratings.entrySet()) {
            double percentage = (double) entry.getValue() / total * 100.0;
            context.write(new DoubleWritable(entry.getKey()), new DoubleWritable(percentage));
        }
    }

}
```

3) MapReduce.java

```java
package Review.PercentageOfRatings;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.*;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.IntWritable;
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;

public class MapReduce {
    public static void main(String[] args) throws IOException, InterruptedException,
ClassNotFoundException {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Percentage of Ratings");

        job.setJarByClass(MapReduce.class);

        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);

        job.setMapperClass(RateMapper.class);
        job.setReducerClass(RateReducer.class);

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

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

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

        FileSystem fs = FileSystem.get(conf);
        fs.delete(new Path(args[1]), true);

        long startTime = System.currentTimeMillis();
        boolean result = job.waitForCompletion(true);
        long endTime = System.currentTimeMillis();
        System.out.println("Hadoop Percentage of Ratings MapReduce Time: " + (endTime - startTime)
+ " ms");

        System.exit(result ? 0 : 1);
    }
}
```

(4) MapReduce – Moving Average Rating of Each Business – 10 Reviews Each

1) CompositeKey.java

```java
package Review.MovingAverageRatingOfEachBusiness;

import org.apache.hadoop.io.WritableComparable;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.*;

public class CompositeKey implements WritableComparable<CompositeKey> {

    private String businessId;
    private long timestamp;

    public CompositeKey() {
    }

    public CompositeKey(String businessId, long timestamp) {
        this.businessId = businessId;
        this.timestamp = timestamp;
    }

    @Override
    public int compareTo(CompositeKey o) {
        if (businessId.compareTo(o.getBusinessId()) != 0) {
            return businessId.compareTo(o.getBusinessId());
        }
        return Long.compare(timestamp, o.getTimestamp());
    }

    @Override
    public void write(DataOutput dataOutput) throws IOException {
        dataOutput.writeUTF(businessId);
        dataOutput.writeLong(timestamp);
    }

    @Override
    public void readFields(DataInput dataInput) throws IOException {
        businessId = dataInput.readUTF();
        timestamp = dataInput.readLong();
    }

    public String getBusinessId() {
        return businessId;
    }

    public void setBusinessId(String businessId) {
        this.businessId = businessId;
    }

    public long getTimestamp() {
        return timestamp;
    }

    public void setTimestamp(long timestamp) {
        this.timestamp = timestamp;
    }

    @Override
    public String toString() {
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        String dateString = sdf.format(new Date(timestamp));
        return businessId + "\t" + dateString;
    }
}
```

2) CompositeKeyComparator.java

```java
package Review.MovingAverageRatingOfEachBusiness;

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

public class CompositeKeyComparator extends WritableComparator {

    public CompositeKeyComparator() {
        super(CompositeKey.class, true);
    }

    public int compare(WritableComparable a, WritableComparable b) {
        CompositeKey ck1 = (CompositeKey) a;
        CompositeKey ck2 = (CompositeKey) b;

        if (ck1.getBusinessId().compareTo(ck2.getBusinessId()) != 0) {
            return ck1.getBusinessId().compareTo(ck2.getBusinessId());
        }
        return Long.compare(ck1.getTimestamp(), ck2.getTimestamp());
    }
}
```

3) NaturalKeyGroupingComparator.java

```java
package Review.MovingAverageRatingOfEachBusiness;

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

public class NaturalKeyGroupingComparator extends WritableComparator {
    public NaturalKeyGroupingComparator() {
        super(CompositeKey.class, true);
    }

    public int compare(WritableComparable a, WritableComparable b) {
        CompositeKey ck1 = (CompositeKey) a;
        CompositeKey ck2 = (CompositeKey) b;
        return ck1.getBusinessId().compareTo(ck2.getBusinessId());
    }

}
```

4) NaturalKeyPartitioner.java

```java
package Review.MovingAverageRatingOfEachBusiness;

import org.apache.hadoop.mapreduce.Partitioner;

public class NaturalKeyPartitioner extends Partitioner<CompositeKey, BusinessRatingData> {

    @Override
    public int getPartition(CompositeKey key, BusinessRatingData value, int numPartitions) {
        return key.getBusinessId().hashCode() % numPartitions;
    }

}
```

5) BusinessRatingData.java

```java
package Review.MovingAverageRatingOfEachBusiness;

import org.apache.hadoop.io.WritableComparable;

import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;

public class BusinessRatingData implements WritableComparable<BusinessRatingData> {

    private long timestamp;
    private double rate;

    public BusinessRatingData() {
    }

    public BusinessRatingData(long timestamp, double rate) {
        this.timestamp = timestamp;
        this.rate = rate;
    }

    @Override
    public int compareTo(BusinessRatingData o) {
        return Long.compare(timestamp, o.getTimestamp());
    }

    @Override
    public void write(DataOutput dataOutput) throws IOException {
        dataOutput.writeLong(timestamp);
        dataOutput.writeDouble(rate);
    }

    @Override
    public void readFields(DataInput dataInput) throws IOException {
        timestamp = dataInput.readLong();
        rate = dataInput.readDouble();
    }

    public long getTimestamp() {
        return timestamp;
    }

    public void setTimestamp(long timestamp) {
        this.timestamp = timestamp;
    }

    public double getRate() {
        return rate;
    }

    public void setRate(double rate) {
        this.rate = rate;
    }

    @Override
    public String toString() {
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
        String dateString = sdf.format(new Date(timestamp));
        return dateString + "\t" + rate;
    }
}
```

6) ReviewMapper.java

```java
package Review.MovingAverageRatingOfEachBusiness;

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

import java.io.IOException;
import java.text.ParseException;
import java.text.SimpleDateFormat;

public class ReviewMapper extends Mapper<LongWritable, Text, CompositeKey, BusinessRatingData> {

    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
        String jsonStr = value.toString();
        JSONObject object = new JSONObject(jsonStr);

        String businessId = object.getString("business_id");
        String dateString = object.getString("date");
        double rate = object.getDouble("stars");

        long timestamp = 0;

        try {
            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
            timestamp = sdf.parse(dateString).getTime();
        } catch (ParseException e) {
            e.printStackTrace();
        }

        CompositeKey ck = new CompositeKey(businessId, timestamp);
        BusinessRatingData brd = new BusinessRatingData(timestamp, rate);
        context.write(ck, brd);
    }

}
```

7) MovingAverage.java

```java
package Review.MovingAverageRatingOfEachBusiness;

import java.util.*;

public class MovingAverage {

    private double sum = 0.0;
    private final int period;
    private final Queue<Double> window = new LinkedList<>();

    public MovingAverage(int period) {
        this.period = period;
    }

    public void addNewNumber(double number) {
        sum += number;
        window.add(number);
        if (window.size() > period) {
            sum -= window.remove();
        }
    }

    public double getMovingAverage() {
        return sum / window.size();
    }
}
```

8) BusinessReviewReducer.java

```java
package Review.MovingAverageRatingOfEachBusiness;

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

import java.io.IOException;
import java.text.*;
import java.util.*;

public class BusinessReviewReducer extends Reducer<CompositeKey, BusinessRatingData, Text, Text> {

    private Text outputKey = new Text();
    private Text outputValue = new Text();

    public void reduce(CompositeKey key, Iterable<BusinessRatingData> values, Context context)
throws IOException, InterruptedException {
        MovingAverage ma = new MovingAverage(10);
        for (BusinessRatingData value : values) {
            ma.addNewNumber(value.getRate());
            double movingAverage = ma.getMovingAverage();
            DecimalFormat df = new DecimalFormat("##0.00");

            long timestamp = value.getTimestamp();
            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
            String dateString = sdf.format(new Date(timestamp));

            outputKey.set(key.getBusinessId());
            outputValue.set(dateString + "\t" + df.format(movingAverage));

            context.write(outputKey, outputValue);
        }
    }

}
```

9) MapReduce.java

```java
package Review.MovingAverageRatingOfEachBusiness;

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

public class MapReduce {
    public static void main(String args[]) throws Exception {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Moving Average Rating of Each Business");

        job.setJarByClass(MapReduce.class);

        job.setPartitionerClass(NaturalKeyPartitioner.class);
        job.setGroupingComparatorClass(NaturalKeyGroupingComparator.class);
        job.setSortComparatorClass(CompositeKeyComparator.class);

        job.setMapOutputKeyClass(CompositeKey.class);
        job.setMapOutputValueClass(BusinessRatingData.class);

        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);

        job.setMapperClass(ReviewMapper.class);
        job.setReducerClass(BusinessReviewReducer.class);

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

        FileSystem fs = FileSystem.get(conf);
        fs.delete(new Path(args[1]), true);

        long startTime = System.currentTimeMillis();
        boolean result = job.waitForCompletion(true);
        long endTime = System.currentTimeMillis();
        System.out.println("Hadoop Moving Average Rating of Each Business MapReduce Time: " +
(endTime - startTime) + " ms");

        System.exit(result ? 0 : 1);

    }
}
```

2. HBase

(1) Import Business Data

1) Insert.java

```
package Business;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.TableName;
import org.apache.hadoop.hbase.client.*;
import org.apache.hadoop.hbase.util.Bytes;
import org.json.JSONObject;

import java.io.*;
import java.util.*;

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

        Configuration conf = HBaseConfiguration.create();
        Connection connection = ConnectionFactory.createConnection(conf);

        Admin admin = connection.getAdmin();
        TableName tableName = TableName.valueOf("business");
        if (!admin.isTableAvailable(tableName)) {
            TableDescriptorBuilder tdb = TableDescriptorBuilder.newBuilder(tableName);

            List<ColumnFamilyDescriptor> cfd = new ArrayList<>();
            cfd.add(ColumnFamilyDescriptorBuilder.newBuilder(Bytes.toBytes("info")).build());
            cfd.add(ColumnFamilyDescriptorBuilder.newBuilder(Bytes.toBytes("position")).build());
            tdb.setColumnFamilies(cfd);

            admin.createTable(tdb.build());
        }
        Table table = connection.getTable(tableName);

        FileInputStream is = new FileInputStream("/Users/kinyang/yelp_dataset/business.json");
        BufferedReader bf = new BufferedReader(new InputStreamReader(is));
        String line;
        while ((line = bf.readLine()) != null) {
            JSONObject object = new JSONObject(line);
            Put p = new Put(Bytes.toBytes(object.getString("business_id")));
            p.addColumn(Bytes.toBytes("info"), Bytes.toBytes("name"),
Bytes.toBytes(object.getString("name")));
            p.addColumn(Bytes.toBytes("info"), Bytes.toBytes("stars"),
Bytes.toBytes(object.getDouble("stars")));
            p.addColumn(Bytes.toBytes("info"), Bytes.toBytes("review_count"),
Bytes.toBytes(object.getInt("review_count")));
            p.addColumn(Bytes.toBytes("info"), Bytes.toBytes("is_open"),
Bytes.toBytes(object.getInt("is_open")));
            p.addColumn(Bytes.toBytes("position"), Bytes.toBytes("address"),
Bytes.toBytes(object.getString("address")));
            p.addColumn(Bytes.toBytes("position"), Bytes.toBytes("city"),
Bytes.toBytes(object.getString("city")));
            p.addColumn(Bytes.toBytes("position"), Bytes.toBytes("state"),
Bytes.toBytes(object.getString("state")));
            p.addColumn(Bytes.toBytes("position"), Bytes.toBytes("postal_code"),
Bytes.toBytes(object.getString("postal_code")));
            p.addColumn(Bytes.toBytes("position"), Bytes.toBytes("latitude"),
Bytes.toBytes(object.getDouble("latitude")));
            p.addColumn(Bytes.toBytes("position"), Bytes.toBytes("longitude"),
Bytes.toBytes(object.getDouble("longitude")));
            table.put(p);
        }
        bf.close();
        is.close();
        table.close();
        connection.close();

    }
}
```

(2) Import Review Data

1) Insert.java

```
package Review;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.TableName;
import org.apache.hadoop.hbase.client.*;
import org.apache.hadoop.hbase.util.Bytes;
import org.json.JSONObject;

import java.io.*;
import java.util.*;

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

        Configuration conf = HBaseConfiguration.create();
        Connection connection = ConnectionFactory.createConnection(conf);

        Admin admin = connection.getAdmin();
        TableName tableName = TableName.valueOf("review");
        if (!admin.isTableAvailable(tableName)) {
            TableDescriptorBuilder tdb = TableDescriptorBuilder.newBuilder(tableName);

            List<ColumnFamilyDescriptor> cfd = new ArrayList<>();
            cfd.add(ColumnFamilyDescriptorBuilder.newBuilder(Bytes.toBytes("info")).build());
            cfd.add(ColumnFamilyDescriptorBuilder.newBuilder(Bytes.toBytes("vote")).build());
            tdb.setColumnFamilies(cfd);

            admin.createTable(tdb.build());
        }


        Table table = connection.getTable(tableName);

        FileInputStream is = new FileInputStream("/Users/kinyang/yelp_dataset/review.json");
        BufferedReader bf = new BufferedReader(new InputStreamReader(is));
        String line;
        while ((line = bf.readLine()) != null) {
            JSONObject object = new JSONObject(line);
            Put p = new Put(Bytes.toBytes(object.getString("review_id")));
            p.addColumn(Bytes.toBytes("info"), Bytes.toBytes("user_id"),
Bytes.toBytes(object.getString("user_id")));
            p.addColumn(Bytes.toBytes("info"), Bytes.toBytes("business_id"),
Bytes.toBytes(object.getString("business_id")));
            p.addColumn(Bytes.toBytes("info"), Bytes.toBytes("rate"),
Bytes.toBytes(object.getDouble("stars")));
            p.addColumn(Bytes.toBytes("info"), Bytes.toBytes("date"),
Bytes.toBytes(object.getString("date")));
            p.addColumn(Bytes.toBytes("vote"), Bytes.toBytes("useful"),
Bytes.toBytes(object.getInt("useful")));
            p.addColumn(Bytes.toBytes("vote"), Bytes.toBytes("funny"),
Bytes.toBytes(object.getInt("funny")));
            p.addColumn(Bytes.toBytes("vote"), Bytes.toBytes("cool"),
Bytes.toBytes(object.getInt("cool")));
            table.put(p);
        }
        bf.close();
        is.close();
        table.close();
        connection.close();

    }
}
```

(3) MapReduce – Numbers of Businesses in Each City and State

1) StateCityMapper.java

```
package Business.NumbersInEachCityAndState;

import org.apache.hadoop.hbase.client.Result;
import org.apache.hadoop.hbase.io.ImmutableBytesWritable;
import org.apache.hadoop.hbase.mapreduce.TableMapper;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;

import java.io.IOException;

public class StateCityMapper extends TableMapper<Text, IntWritable> {

    private static final byte[] POSITION = "position".getBytes();
    private static final byte[] STATE = "state".getBytes();
    private static final byte[] CITY = "city".getBytes();

    private Text key = new Text();
    private IntWritable one = new IntWritable(1);

    public void map(ImmutableBytesWritable row, Result value, Context context) throws IOException,
InterruptedException {
        byte[] tmp1 = value.getValue(POSITION, STATE);
        byte[] tmp2 = value.getValue(POSITION, CITY);

        String state = new String(tmp1);
        String city = new String(tmp2);
        key.set(state + "\t" + city);
        context.write(key, one);
    }

}
```

2) StateCityReducer.java

```
package Business.NumbersInEachCityAndState;

import org.apache.hadoop.hbase.client.Put;
import org.apache.hadoop.hbase.io.ImmutableBytesWritable;
import org.apache.hadoop.hbase.mapreduce.TableReducer;
import org.apache.hadoop.hbase.util.Bytes;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;

import java.io.IOException;

public class StateCityReducer extends TableReducer<Text, IntWritable, ImmutableBytesWritable> {

    private static final byte[] POSITION = "position".getBytes();
    private static final byte[] STATE = "state".getBytes();
    private static final byte[] CITY = "city".getBytes();
    private static final byte[] RESULTS = "results".getBytes();
    private static final byte[] COUNT = "count".getBytes();
    private static int row = 0;

    public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException,
InterruptedException {
        String[] tmp = key.toString().split("\t");
        String state = tmp[0];
        String city = tmp.length == 2 ? tmp[1] : "";

        int sum = 0;
        for (IntWritable value : values) {
            sum += value.get();
        }

        Put p = new Put(Bytes.toBytes(++row));
        p.addColumn(POSITION, STATE, Bytes.toBytes(state));
        p.addColumn(POSITION, CITY, Bytes.toBytes(city));
        p.addColumn(RESULTS, COUNT, Bytes.toBytes(String.valueOf(sum)));

        context.write(null, p);
    }

}
```

3) MapReduce.java

```
package Business.NumbersInEachCityAndState;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.TableName;
import org.apache.hadoop.hbase.client.*;
import org.apache.hadoop.hbase.mapreduce.TableMapReduceUtil;
import org.apache.hadoop.hbase.util.Bytes;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;

import java.util.*;

public class MapReduce {
    public static void main(String[] args) throws Exception {
        Configuration conf = HBaseConfiguration.create();
        Job job = Job.getInstance(conf, "Numbers In Each City And State");
        job.setJarByClass(Review.PercentageOfRatings.MapReduce.class);

        Scan scan = new Scan();
```

```java
        scan.setCaching(500);
        scan.setCacheBlocks(false);

        Connection connection = ConnectionFactory.createConnection(conf);
        Admin admin = connection.getAdmin();
        if (!admin.isTableAvailable(TableName.valueOf("numbers_in_each_city_and_state"))) {
            TableDescriptorBuilder tdb =
TableDescriptorBuilder.newBuilder(TableName.valueOf("numbers_in_each_city_and_state"));
            List<ColumnFamilyDescriptor> cfd = new ArrayList<>();
            cfd.add(ColumnFamilyDescriptorBuilder.newBuilder(Bytes.toBytes("position")).build());
            cfd.add(ColumnFamilyDescriptorBuilder.newBuilder(Bytes.toBytes("results")).build());
            tdb.setColumnFamilies(cfd);
            admin.createTable(tdb.build());
        }
        connection.close();

        TableMapReduceUtil.initTableMapperJob("business", scan, StateCityMapper.class, Text.class,
IntWritable.class, job);
        TableMapReduceUtil.initTableReducerJob("numbers_in_each_city_and_state",
StateCityReducer.class, job);

        long startTime = System.currentTimeMillis();
        boolean result = job.waitForCompletion(true);
        long endTime = System.currentTimeMillis();
        System.out.println("HBase Numbers In Each City And State MapReduce Time: " + (endTime -
startTime) + " ms");

        System.exit(result ? 0 : 1);
    }
}
```

(4) MapReduce – Percentage of Ratings

1) RateMapper.java

```java
package Review.PercentageOfRatings;

import org.apache.hadoop.hbase.client.Result;
import org.apache.hadoop.hbase.io.ImmutableBytesWritable;
import org.apache.hadoop.hbase.mapreduce.TableMapper;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.IntWritable;

import java.io.IOException;
import java.nio.ByteBuffer;

public class RateMapper extends TableMapper<DoubleWritable, IntWritable> {
    private static final byte[] info = "info".getBytes();
    private static final byte[] rate = "rate".getBytes();

    private DoubleWritable key = new DoubleWritable();
    private IntWritable one = new IntWritable(1);

    public void map(ImmutableBytesWritable row, Result value, Context context) throws IOException,
InterruptedException {
        byte[] tmp = value.getValue(info, rate);
        double rating = ByteBuffer.wrap(tmp).getDouble();
        key.set(rating);
        context.write(key, one);
    }
}
```

2) RateReducer.java

```java
package Review.PercentageOfRatings;

import org.apache.hadoop.hbase.client.Put;
import org.apache.hadoop.hbase.io.ImmutableBytesWritable;
import org.apache.hadoop.hbase.mapreduce.TableReducer;
import org.apache.hadoop.hbase.util.Bytes;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.IntWritable;

import java.io.IOException;
import java.util.*;

public class RateReducer extends TableReducer<DoubleWritable, IntWritable, ImmutableBytesWritable>
{

    private static final byte[] RESULTS = "results".getBytes();
    private static final byte[] PERCENTAGE = "percentage".getBytes();

    private HashMap<Double, Integer> ratings;
    private int total;

    public void setup(Context context) {
        ratings = new HashMap<>();
        total = 0;
    }

    public void reduce(DoubleWritable key, Iterable<IntWritable> values, Context context) {
        double rate = key.get();
        int sum = ratings.getOrDefault(rate, 0);
        for (IntWritable value : values) {
            sum += value.get();
            total += value.get();
        }
        ratings.put(rate, sum);
    }

    public void cleanup(Context context) throws IOException, InterruptedException {
        for (Map.Entry<Double, Integer> entry : ratings.entrySet()) {
            double rate = entry.getKey();
            double percentage = (double) entry.getValue() / total * 100.0;
            Put p = new Put(Bytes.toBytes(String.valueOf(rate)));
            p.addColumn(RESULTS, PERCENTAGE, Bytes.toBytes(String.valueOf(percentage)));
            context.write(null, p);
        }
    }

}
```

3) MapReduce.java

```java
package Review.PercentageOfRatings;


import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.TableName;
import org.apache.hadoop.hbase.client.*;
import org.apache.hadoop.hbase.mapreduce.TableMapReduceUtil;
import org.apache.hadoop.hbase.util.Bytes;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.mapreduce.Job;

public class MapReduce {
    public static void main(String[] args) throws Exception {
        Configuration conf = HBaseConfiguration.create();
        Job job = Job.getInstance(conf, "Percentage of Ratings");
        job.setJarByClass(MapReduce.class);

        Scan scan = new Scan();
        scan.setCaching(500);
        scan.setCacheBlocks(false);

        Connection connection = ConnectionFactory.createConnection(conf);
        Admin admin = connection.getAdmin();
        if (!admin.isTableAvailable(TableName.valueOf("percentage_of_ratings"))) {
            TableDescriptorBuilder tdb =
TableDescriptorBuilder.newBuilder(TableName.valueOf("percentage_of_ratings"));

tdb.setColumnFamily(ColumnFamilyDescriptorBuilder.newBuilder(Bytes.toBytes("results")).build());
            admin.createTable(tdb.build());
        }
        connection.close();

        TableMapReduceUtil.initTableMapperJob("review", scan, RateMapper.class,
DoubleWritable.class, IntWritable.class, job);
        TableMapReduceUtil.initTableReducerJob("percentage_of_ratings", RateReducer.class, job);

        long startTime = System.currentTimeMillis();
        boolean result = job.waitForCompletion(true);
        long endTime = System.currentTimeMillis();
        System.out.println("HBase Percentage of Ratings MapReduce Time: " + (endTime - startTime) +
" ms");

        System.exit(result ? 0 : 1);
    }
}
```

3. Hive

(1) Create Table and Import Data

1) import_business.sql

```
add jar /Users/kinyang/Github/INFO_7250/Jars/json-serde-1.3.8-jar-with-dependencies.jar;

drop table if exists business;
create table business (
    business_id string,
    name string,
    address string,
    state string,
    city string,
    postal_code string,
    latitude double,
    longitude double,
    stars double,
    review_count int,
    is_open int,
    attributes string,
    categories array<string>,
    hours map<string, string>
)
row format serde 'org.openx.data.jsonserde.JsonSerDe'
stored as textfile;

load data local inpath '/Users/kinyang/yelp_dataset/business.json' overwrite into table business;
```

2) import_review.sql

```
add jar /Users/kinyang/Github/INFO_7250/Jars/hive-hcatalog-core-3.1.2.jar;

drop table if exists review;
create table review (
    review_id string,
    user_id string,
    business_id string,
    stars int,
    `date` string,
    text string,
    useful int,
    funny int,
    cool int
)
row format serde 'org.apache.hive.hcatalog.data.JsonSerDe'
stored as textfile;

load data local inpath '/Users/kinyang/yelp_dataset/review.json' overwrite into table review;
```

(2) Numbers of Businesses in Each City and State

1) number_of_business_each_city_and_state.sql

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

(3) Percentage of Ratings

1) percentage_of_ratings.sql

```
drop table if exists percentage_of_ratings;
create table percentage_of_ratings
row format delimited
    fields terminated by ','
    lines terminated by '\n'
as
with tmp1 as (
    select count(*) as total
    from review
), tmp2 as (
    select stars, count(*) as count
    from review
    group by stars
)
select stars, concat(count / total * 100, '%') as percentage
from tmp1, tmp2;
```