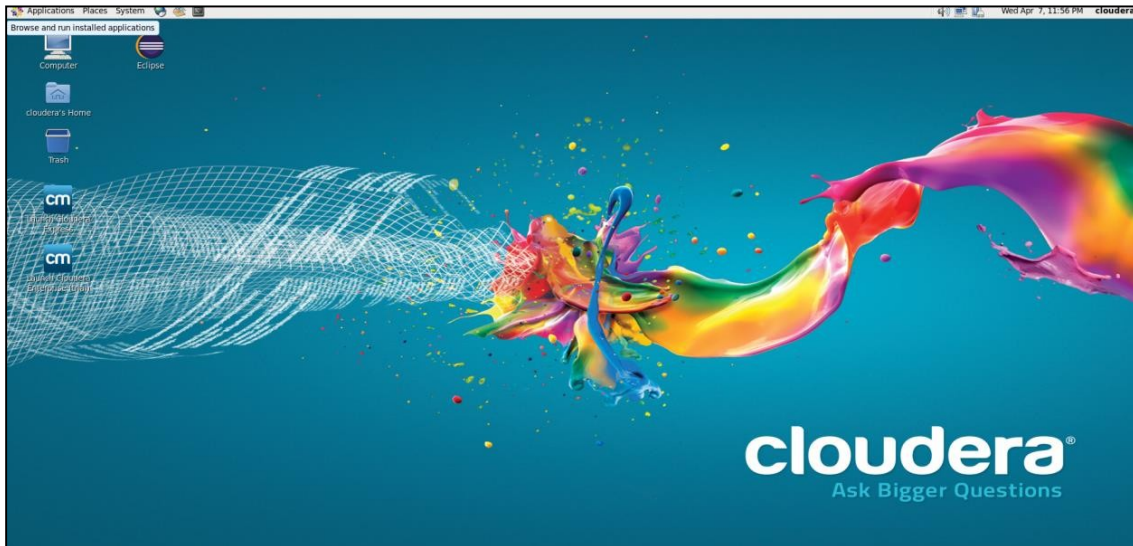


MapReduce for Frequent Itemset Mining:

Let's solve a frequent itemset mining problem using MapReduce.

Step 1: Open Cloudera Quickstart VM



Step 2: Clone the following repository on your local machine.

[www.github.com/NSTiwari/Hadoop-MapReduce-Programs](https://github.com/NSTiwari/Hadoop-MapReduce-Programs)

You'll find the **Frequent-Itemset-Mining** folder along with some other folders. The **Frequent-Itemset-Mining** directory contains three files –

- **frequent_itemset_data.txt** – The text data file that contains transactions.
- **frequent_itemset_mapper.py** – The mapper file for Frequent Itemset Mining.
- **frequent_itemset_reducer.py** – The reducer file for Frequent Itemset Mining.

```
frequent_itemset_data.txt X
apple orange grapes
apple mango
orange apple chickoo
apple grapes orange
milk butter
butter cheese
bread milk butter
cheese bread butter milk
cheese apple orange
```

frequent_itemset_mapper.py:

```
#!/usr/bin/env python

import sys
import re

# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # split the line into digits
    words = re.findall(r'\w+', line)

    lst = []
    for i in words:
        lst.append(i)    #lst contains items in each transaction

    for item1 in words:
        for item2 in lst:
            if item1!=item2 and lst:
                print('%s\t%s\t%s' % (item1, item2, 1))
#outputs frequent 2itemsets with value 1 and key=item1, item2

        lst.pop(0)
```

frequent_itemset_reducer.py:

```
#!/usr/bin/env python

import sys
item1 = None
item2 = None
current_count = 0

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    key1, key2, count = line.split('\t', 2)
    # print(key1, key2, count)

    # convert count (currently a string) to int
    try:
        count = int(count)
    except ValueError:
        # count was not a number, ignore
        continue

    if item1 == key1 and item2 == key2:
        current_count += count
    else:
        if item1!=None and item2!=None and current_count>=2:
```

```

# output as a frequent 2 itemset if support>2
    print('%s\t%s' % (item1, item2))

current_count = count
item1 = key1
item2 = key2

# last itemset

if item1 == key1 and item2 == key2 and current_count>=2:
    print('%s\t%s' % (item1, item2))

```

Copy these three files inside `/home/cloudera` directory. Once done, confirm their presence.

ls

```

cloudera@quickstart:~$ ls
cloudera-manager  Documents          fm_data.txt      frequent_itemset_data.txt  input~  Music  Videos
cm_api.py         Downloads          fm_mapper.py     frequent_itemset_mapper.py  input.txt~  Pictures  workspace
data2.txt~        eclipse            fm_mapper.py~   frequent_itemset_reducer.py  kerberos  Public
data.txt~         enterprise-deployment.json  fm_reducer.py~  Hadoop-MapReduce-Programs  lib       reducer.py~
Desktop           express-deployment.json    fm_reducer.py~  hadoop-streaming-2.7.3.jar  mapper.py~  Templates
cloudera@quickstart ~$

```

All three files are present.

Step 3: Test MapReduce program locally.

Now that we have the input data file of transactions and the mapper, reducer files, let's test them locally to see if the program is correct.

Just run the mapper file on the input data.

cat frequent_itemset_data.txt | python frequent_itemset_mapper.py | sort

```

[cloudera@quickstart ~]$ cat frequent_itemset_data.txt | python frequent_itemset_mapper.py | sort
apple  chickoo  1
apple  grapes   1
apple  grapes   1
apple  mango    1
apple  orange   1
apple  orange   1
apple  orange   1
apple  orange   1
bread  butter   1
bread  butter   1
bread  milk     1
bread  milk     1
butter cheese  1
butter milk    1
cheese apple   1
cheese bread   1
cheese butter  1
cheese milk    1
cheese orange  1
grapes orange   1
milk  butter   1
milk  butter   1
orange apple   1
orange chickoo  1
orange grapes   1

```

As you can see, for each transaction, all possible pairs of items have been created, mapped with count 1 each and then sorted in lexicographical order.

So far, so good. Now let's run the complete MapReduce program.

```
cat frequent_itemset_data.txt | python frequent_itemset_mapper.py | sort | python frequent_itemset_reducer.py
```

```
[cloudera@quickstart ~]$ cat frequent_itemset_data.txt | python frequent_itemset_mapper.py | sort | python frequent_itemset_reducer.py
apple  grapes
apple  orange
bread  butter
bread  milk
milk   butter
```

And here we go. These are the pair of items that are frequently bought for minimum support of 2.

Results obtained are as expected. Now we can run this on Hadoop.

Step 4: Create a directory on HDFS.

```
sudo -u hdfs hadoop fs -mkdir /frequent_itemset
hdfs dfs -ls /
```

```
cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -mkdir /frequent_itemset
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 7 items
drwxr-xr-x - cloudera supergroup 0 2021-04-06 01:49 /flajolet_martin
drwxr-xr-x - hdfs supergroup 0 2021-04-08 00:53 /frequent_itemset
drwxr-xr-x - hbase supergroup 0 2021-04-05 06:21 /hbase
drwxr-xr-x - solr solr 0 2015-06-09 03:38 /solr
drwxrwxrwx - hdfs supergroup 0 2021-04-05 06:45 /tmp
drwxr-xr-x - hdfs supergroup 0 2021-04-05 07:36 /user
drwxr-xr-x - hdfs supergroup 0 2015-06-09 03:36 /var
```

Step 5: Copy input file to HDFS.

```
sudo -u hdfs hadoop fs -put /home/cloudera/frequent_itemset_data.txt /frequent_itemset
hdfs dfs -ls /frequent_itemset
```

```
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -put /home/cloudera/frequent_itemset_data.txt /frequent_itemset
[cloudera@quickstart ~]$ hdfs dfs -ls /frequent_itemset
Found 1 items
-rw-r--r-- 1 hdfs supergroup 169 2021-04-08 00:56 /frequent_itemset/frequent_itemset_data.txt
[cloudera@quickstart ~]$
```

The input file is copied successfully inside **frequent_itemset** directory on HDFS.

Step 6: Configure permissions to run MapReduce for Frequent Itemset Mining on Hadoop.

Now, we have to provide permission to read, write and execute the MapReduce program before the MapReduce job is executed on Hadoop. We also need to provide permission for the default user (cloudera) to write the output file on HDFS.

```
chmod 777 frequent_itemset_mapper.py frequent_itemset_reducer.py
sudo -u hdfs hadoop fs -chown cloudera /frequent_itemset
```

```
[cloudera@quickstart ~]$ chmod 777 frequent_itemset_mapper.py frequent_itemset_reducer.py
[cloudera@quickstart ~]$ sudo -u hdfs hadoop fs -chown cloudera /frequent_itemset
[cloudera@quickstart ~]$
```

All the required permissions have now been configured. Let's now execute Hadoop streaming.

Step 7: Run MapReduce on Hadoop.

Run the following command on terminal.

```
hadoop jar /home/cloudera/hadoop-streaming-2.7.3.jar \  
> -input /frequent_itemset/frequent_data.txt \  
> -output /frequent_itemset/output \  
> -mapper /home/cloudera/frequent_map.py \  
> -reducer /home/cloudera/frequent_reduce.py
```

```
[cloudera@quickstart ~]$ hadoop jar /home/cloudera/hadoop-streaming-2.7.3.jar \  
-input /frequent_itemset/frequent_data.txt \  
-output /frequent_itemset/output \  
-mapper /home/cloudera/frequent_map.py \  
-reducer /home/cloudera/frequent_reduce.py  
packageJobJar: [/usr/jars/hadoop-streaming-2.6.0-cdh5.4.2.jar] /tmp/streamjob860207932224186303.jar tmpDir=null  
21/04/08 06:59:06 INFO client.RMProxy: Connecting to ResourceManager at quickstart.cloudera/127.0.0.1:8032  
21/04/08 06:59:07 INFO client.RMProxy: Connecting to ResourceManager at quickstart.cloudera/127.0.0.1:8032  
21/04/08 06:59:08 INFO mapred.FileInputFormat: Total input paths to process : 1  
21/04/08 06:59:08 INFO mapreduce.JobSubmitter: number of splits:2  
21/04/08 06:59:09 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1617889578702_0001  
21/04/08 06:59:09 INFO impl.YarnClientImpl: Submitted application application_1617889578702_0001  
21/04/08 06:59:09 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1617889578702_0001/  
21/04/08 06:59:09 INFO mapreduce.Job: Running job: job_1617889578702_0001  
21/04/08 06:59:25 INFO mapreduce.Job: Job job_1617889578702_0001 running in uber mode : false  
21/04/08 06:59:25 INFO mapreduce.Job: map 0% reduce 0%  
21/04/08 06:59:42 INFO mapreduce.Job: map 50% reduce 0%  
21/04/08 06:59:43 INFO mapreduce.Job: map 100% reduce 0%  
21/04/08 06:59:55 INFO mapreduce.Job: map 100% reduce 100%  
21/04/08 06:59:55 INFO mapreduce.Job: Job job_1617889578702_0001 completed successfully  
21/04/08 06:59:55 INFO mapreduce.Job: Counters: 49  
File System Counters  
FILE: Number of bytes read=221
```

Execute MapReduce job

MapReduce job status

```
Combine output records=0  
Reduce input groups=7  
Reduce shuffle bytes=283  
Reduce input records=24  
Reduce output records=4  
Spilled Records=48  
Shuffled Maps =2  
Failed Shuffles=0  
Merged Map outputs=2  
GC time elapsed (ms)=447  
CPU time spent (ms)=2860  
Physical memory (bytes) snapshot=379260928  
Virtual memory (bytes) snapshot=2100383744  
Total committed heap usage (bytes)=152174592  
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=255  
File Output Format Counters  
Bytes Written=49  
21/04/08 06:59:55 INFO streaming.StreamJob: Output directory: /frequent_itemset/output  
[cloudera@quickstart ~]$  
[cloudera@quickstart ~]$ hdfs dfs -ls /frequent_itemset/output  
Found 2 items  
-rw-r--r-- 1 hdfs supergroup 0 2021-04-08 06:59 /frequent_itemset/output/_SUCCESS  
-rw-r--r-- 1 hdfs supergroup 49 2021-04-08 06:59 /frequent_itemset/output/part-000000
```

Output directory

Output files

Step 8: Read MapReduce output.

hdfs dfs -cat /frequent_itemset/output/part-00000

```
[cloudera@quickstart ~]$ hdfs dfs -cat /frequent_itemset/output/part-00000  
apple orange  
bread butter  
bread milk  
milk butter
```

Here we go, the output is right here.