Assignment 6 - FMCG MapReduce Analysis

Task 1: Demand-Supply Mismatch Analysis

- Objective: Identify zones and regional zones with the highest mismatch between demand and supply.
- Required Fields: zone, WH_regional_zone, product_wg_ton

Description:

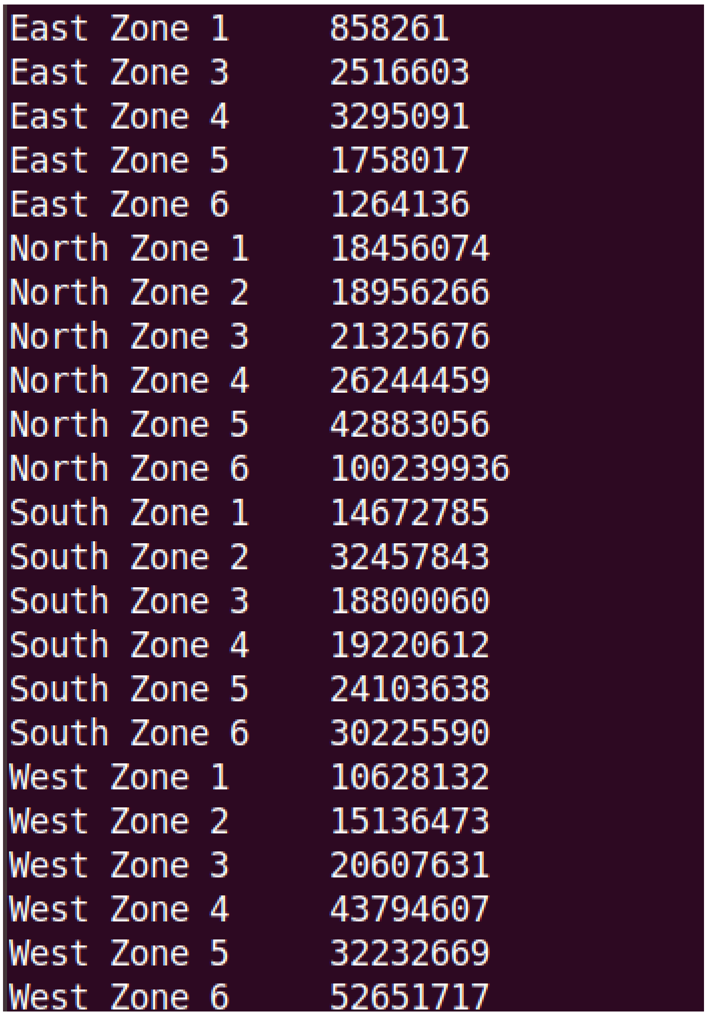
- Map: For each warehouse, emit the zone and regional zone as the key and the product weight shipped in the last three months as the value.
- Reduce: Aggregate the product weight by zone and regional zone to calculate the total supply. Compare this with known demand data to identify mismatches.

mapper:

```
#!/usr/bin/python3
"""mapper.py"""
import sys

for line in sys.stdin:

    try:
        line = line.strip().split(',')

        zone = line[4]
        WH_regional_zone = line[5]
        product_wg_ton = int(line[-1])
    except:
        continue

    print('%s %s\t%s' % (zone, WH_regional_zone, product_wg_ton))
```

reducer:

```
#!/usr/bin/python3
"""reducer.py"""
import sys

current_zone = None
current_wg = 0
word = None

for line in sys.stdin:

    line = line.strip()
    zone, product_wg = line.split('\t')

    try:
        product_wg = int(product_wg)
    except ValueError:
        continue

    if current_zone == zone:
        current_wg += product_wg
    else:
        if current_zone:
            print ('%s\t%s' % (current_zone, current_wg))
        current_zone = zone
        current_wg = 0

if current_zone == zone:
    print ('%s\t%s' % (current_zone, current_wg))
```

output 

```
East Zone 1     858261
East Zone 3     2516603
East Zone 4     3295091
East Zone 5     1758017
East Zone 6     1264136
North Zone 1    18456074
North Zone 2    18956266
North Zone 3    21325676
North Zone 4    26244459
North Zone 5    42883056
North Zone 6    100239936
South Zone 1    14672785
South Zone 2    32457843
South Zone 3    18800060
South Zone 4    19220612
South Zone 5    24103638
South Zone 6    30225590
West Zone 1     10628132
West Zone 2     15136473
West Zone 3     20607631
West Zone 4     43794607
West Zone 5     32232669
West Zone 6     52651717
```

Task 2: Warehouse Refill Frequency Correlation

- Objective: Determine the correlation between warehouse capacity and refill frequency.
- Required Fields: WH_capacity_size, num_refill_req_l3m

Description:

- Map: Extract the number of refill requests (num_refill_req_l3m) and warehouse capacity size (WH_capacity_size) for each warehouse. (For each warehouse, emit the capacity size and the number of refill requests as the value)

- Reduce: Aggregate the refill requests by capacity size and calculate the correlation.

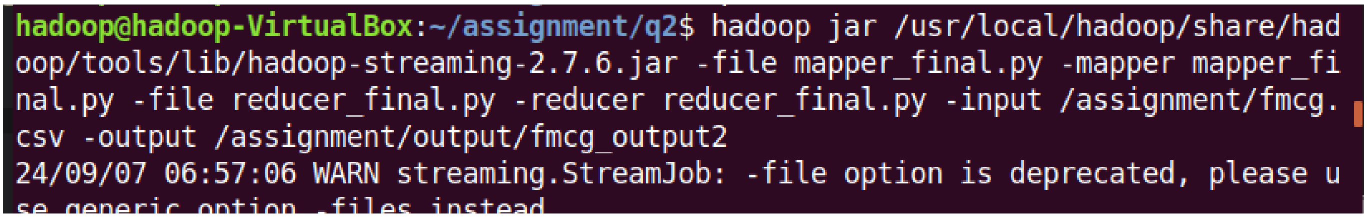
mapper:

```
#!/usr/bin/python3
"""mapper_final.py"""

import sys

for line in sys.stdin:

    lines = line.split(',')

    wh_capacity = lines[3].strip()
    num_req_fill = lines[6].strip()

    try:
        if wh_capacity == 'Small':
            wh_capacity = 0
        elif wh_capacity == 'Mid':
            wh_capacity = 1
        elif wh_capacity == 'Large':
            wh_capacity = 2
        else:
            continue
        num_req_fill = int(num_req_fill)
    except ValueError:
        continue

    print(f"{wh_capacity},{num_req_fill}")
```

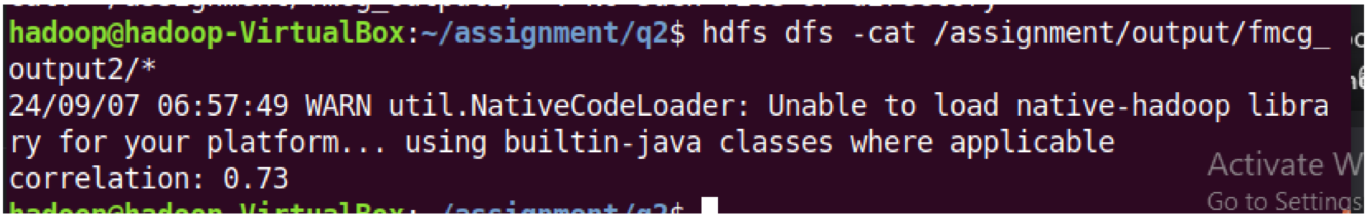
reducer:

```
#!/usr/bin/python3
"""reducer_task2.py"""
import sys
import numpy as np
from collections import defaultdict

capacity = defaultdict(list)

for line in sys.stdin:
    line = line.strip()

    try:
        wh_capacity, num_req_fill = line.split(',')
        wh_capacity = int(wh_capacity)
        num_req_fill = int(num_req_fill)
    except ValueError:
        continue
    capacity[wh_capacity].append(num_req_fill)

wh_capacities = np.array(list(capacity.keys()))
avg_fill = np.array([np.mean(val) for val in capacity.values()])

corr = np.corrcoef(wh_capacities, avg_fill)
print("correlation: %.2f" %  corr[0, 1])
```

output



```
hadoop@hadoop-VirtualBox:~/assignment/q2$ hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.7.6.jar -file mapper_final.py -mapper mapper_final.py -file reducer_final.py -reducer reducer_final.py -input /assignment/fmcg.csv -output /assignment/output/fmcg_output2
24/09/07 06:57:06 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead
```



```
hadoop@hadoop-VirtualBox:~/assignment/q2$ hdfs dfs -cat /assignment/output/fmcg_output2/*
24/09/07 06:57:49 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
correlation: 0.73
hadoop@hadoop-VirtualBox:~/assignment/q2$
```

Task 3. Transport Issue Impact Analysis

Objective: Analyse the impact of transport issues on warehouse supply efficiency.

- Required Fields: transport_issue_l1y, product_wg_ton

Description:

- Map: For each warehouse, emit whether a transport issue was reported and the product weight shipped.
- Reduce: Aggregate the product weight by transport issue status to assess the impact.

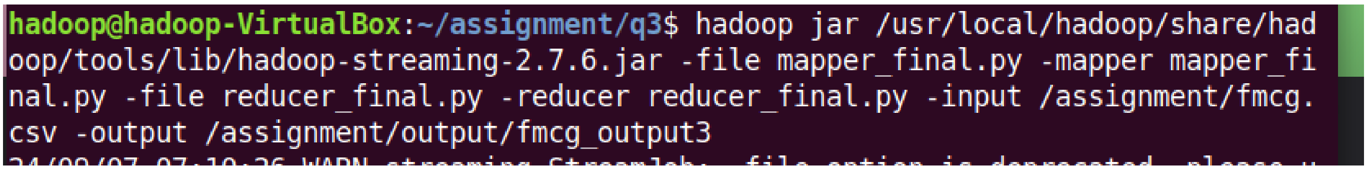
mapper:

```
#!/usr/bin/python3
"""mapper_final.py"""
import sys

for line in sys.stdin:
    lines = line.strip().split(',')
    product = lines[-1]
    transport = lines[7]
    try:
        if int(transport) > 0:
            transport = 'Issue Occured'
        else:
            transport = 'No Issues'
    except:
        continue
    print(f"{transport},{product}")
```

reducer:

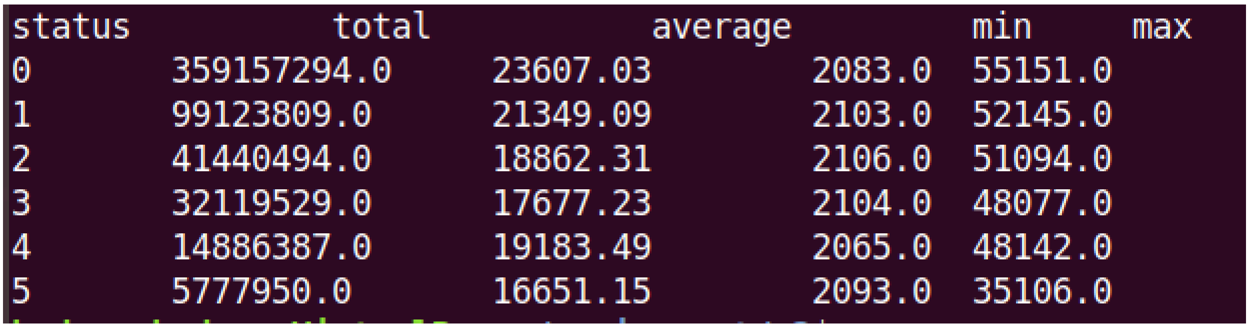
```
#!/usr/bin/python3
"""reducer_final.py"""
import sys

curr_transport = None
curr_sum = []
curr_count = 0

print('status\t\ttotal\t\taverage\t\tmin\tmax')
for line in sys.stdin:
    line = line.strip()

    try:
        transport, product = line.split(',')
        product = float(product)
    except ValueError:
        continue

    if transport != curr_transport:
        if curr_transport is not None:
            print("%s\t%s\t%.2f\t%s\t%s" % (curr_transport, sum(curr_sum),
(sum(curr_sum) / len(curr_sum)), min(curr_sum), max(curr_sum)))

        curr_transport = transport
        curr_sum = []
        curr_count = 1
    else:
        curr_sum.append(product)
        curr_count += 1

if curr_transport is not None:
    print("%s\t%s\t%.2f\t%s\t%s" % (curr_transport, sum(curr_sum), (sum(curr_sum)
/ len(curr_sum)), min(curr_sum), max(curr_sum)))
```

output



```
hadoop@hadoop-VirtualBox:~/assignment/q3$ hadoop jar /usr/local/hadoop/share/had
oop/tools/lib/hadoop-streaming-2.7.6.jar -file mapper_final.py -mapper mapper_fi
nal.py -file reducer_final.py -reducer reducer_final.py -input /assignment/fmcg.
csv -output /assignment/output/fmcg_output3
24/09/07 07:10:26 WARN streaming.StreamJob: -file option is deprecated, please u
```

method 2



```
status          total           average         min     max
0       359157294.0     23607.03        2083.0  55151.0
1       99123809.0      21349.09        2103.0  52145.0
2       41440494.0      18862.31        2106.0  51094.0
3       32119529.0      17677.23        2104.0  48077.0
4       14886387.0      19183.49        2065.0  48142.0
5       5777950.0       16651.15        2093.0  35106.0
```

Task 4. Storage Issue Analysis

- Objective: Evaluate the impact of storage issues on warehouse performance.
- Required Fields: storage_issue_reported_l3m, product_wg_ton

Description:

- Map: For each warehouse, emit whether a storage issue was reported and the product weightn shipped.
- Reduce: Aggregate the product weight by storage issue status to assess the impact

mapper:

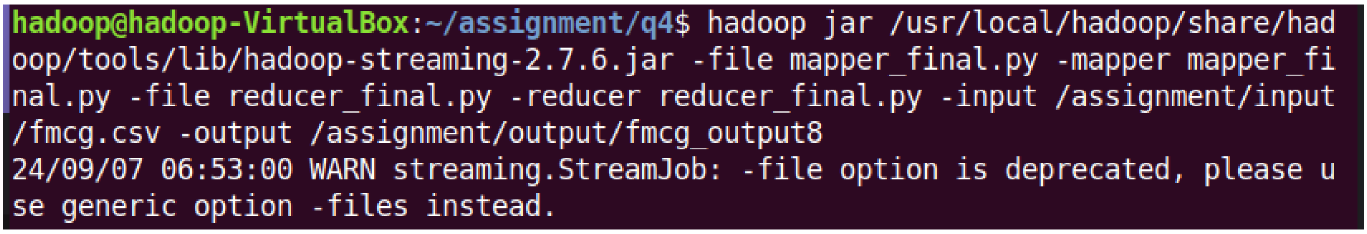
```
#!/usr/bin/python3
"""mapper_final.py"""
import sys

for line in sys.stdin:
    lines = line.strip().split(',')
    product = lines[-1]
    storage = lines[-6]

    try:
        if float(storage) > 0:
            storage = 'Issue Reported'
        else:
            storage = 'No Issues'
    except:
        continue

    print(f"{storage},{product}")
```

reducer:

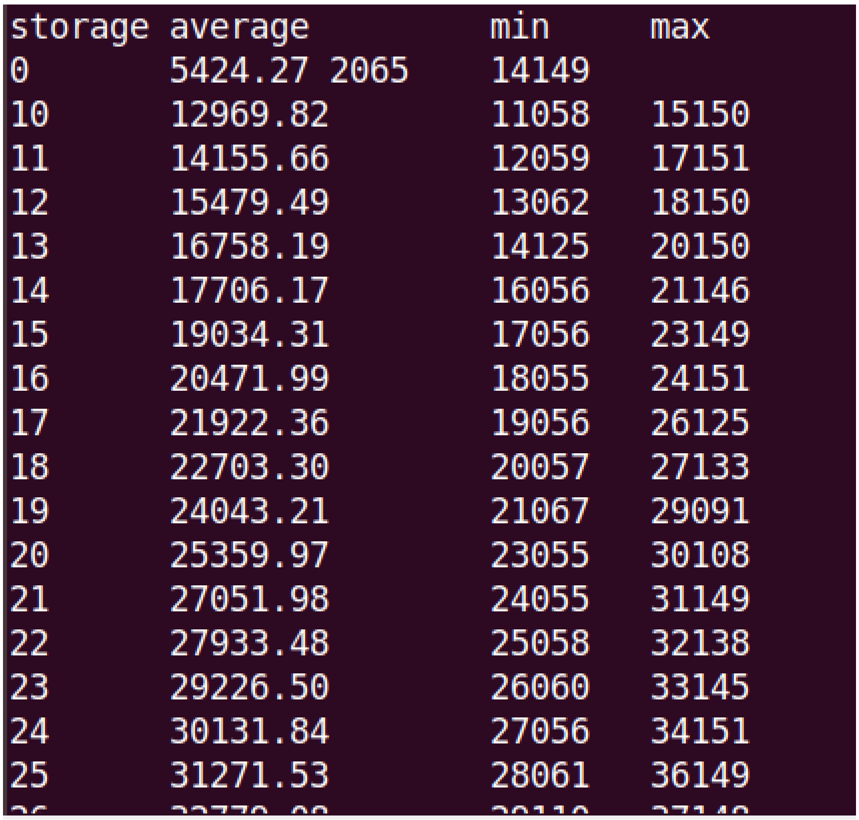
```
#!/usr/bin/python3
"""reducer_final.py"""
import sys

curr_storage = None
curr_sum = []
```

```
curr_count = 0

print("status\t\taverage\t\tmin\tmax")
for line in sys.stdin:
    line = line.strip()

    try:
        storage, product = line.split(',')
        product = int(product)
    except ValueError:
        continue

    if storage != curr_storage:
        if curr_storage is not None:
            print("%s\t%.2f\t%s\t%s" % (curr_storage, (sum(curr_sum) / len(curr_sum)), min(curr_sum), max(curr_sum)))

        curr_storage = storage
        curr_sum = []
        curr_count = 1
    else:
        curr_sum.append(product)
        curr_count += 1

if curr_storage is not None:
    print("%s\t%.2f\t\t%s\t%s" % (curr_storage, (sum(curr_sum) / len(curr_sum)), min(curr_sum), max(curr_sum)))
```

output



```
hadoop@hadoop-VirtualBox:~/assignment/q4$ hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.7.6.jar -file mapper_final.py -mapper mapper_final.py -file reducer_final.py -reducer reducer_final.py -input /assignment/input/fmcg.csv -output /assignment/output/fmcg_output8
24/09/07 06:53:00 WARN streaming.StreamJob: -file option is deprecated, please use generic option -files instead.
```



```
storage average         min     max
0       5424.27 2065    14149
10      12969.82        11058   15150
11      14155.66        12059   17151
12      15479.49        13062   18150
13      16758.19        14125   20150
14      17706.17        16056   21146
15      19034.31        17056   23149
16      20471.99        18055   24151
17      21922.36        19056   26125
18      22703.30        20057   27133
19      24043.21        21067   29091
20      25359.97        23055   30108
21      27051.98        24055   31149
22      27933.48        25058   32138
23      29226.50        26060   33145
24      30131.84        27056   34151
25      31271.53        28061   36149
```