# Assignment 6

## 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.

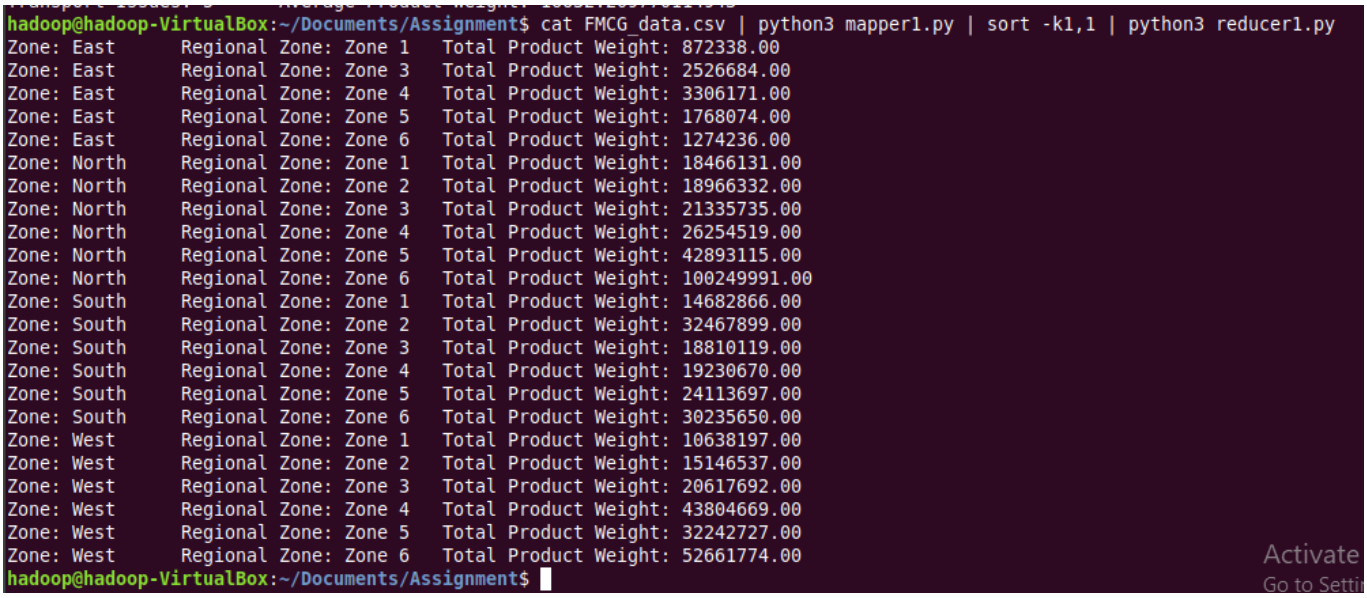
#### Mapper

```
#!/usr/bin/python3
"""mapper1.py"""
import sys

# input comes from standard input
for line in sys.stdin:
    line = line.strip()
    if line:
        columns = line.split(',')
        if columns[4] != "zone" or columns[5] != "WH_regional_zone" or columns[-1] != "product_wg_ton":
            zone = columns[4].strip()
            regional_zone = columns[5].strip()
            product_wg = columns[-1].strip()
            print('%s,%s,%s' % (zone, regional_zone, product_wg))
```

#### Reducer

```
#!/usr/bin/python3
"""reducer1.py"""
import sys
from collections import defaultdict

zone = None
reg_zn = None
zone_wt = defaultdict(lambda: defaultdict(list))

# Read input from stdin
for line in sys.stdin:
    zone, reg_zn, weight = line.strip().split(',')
    try:
        weight = float(weight)
        zone_wt[zone][reg_zn].append(weight)
    except ValueError:
        continue


for zone, WH in zone_wt.items():
    for region, weights in WH.items():
        total = sum(weights)
        print(f"Zone: {zone}\tRegional Zone: {region}\tTotal Product Weight: {total:.2f}")
```

### Output



```
hadoop@hadoop-VirtualBox:~/Documents/Assignment$ cat FMCG_data.csv | python3 mapper1.py | sort -k1,1 | python3 reducer1.py
Zone: East      Regional Zone: Zone 1   Total Product Weight: 872338.00
Zone: East      Regional Zone: Zone 3   Total Product Weight: 2526684.00
Zone: East      Regional Zone: Zone 4   Total Product Weight: 3306171.00
Zone: East      Regional Zone: Zone 5   Total Product Weight: 1768074.00
Zone: East      Regional Zone: Zone 6   Total Product Weight: 1274236.00
Zone: North     Regional Zone: Zone 1   Total Product Weight: 18466131.00
Zone: North     Regional Zone: Zone 2   Total Product Weight: 18966332.00
Zone: North     Regional Zone: Zone 3   Total Product Weight: 21335735.00
Zone: North     Regional Zone: Zone 4   Total Product Weight: 26254519.00
Zone: North     Regional Zone: Zone 5   Total Product Weight: 42893115.00
Zone: North     Regional Zone: Zone 6   Total Product Weight: 100249991.00
Zone: South     Regional Zone: Zone 1   Total Product Weight: 14682866.00
Zone: South     Regional Zone: Zone 2   Total Product Weight: 32467899.00
Zone: South     Regional Zone: Zone 3   Total Product Weight: 18810119.00
Zone: South     Regional Zone: Zone 4   Total Product Weight: 19230670.00
Zone: South     Regional Zone: Zone 5   Total Product Weight: 24113697.00
Zone: South     Regional Zone: Zone 6   Total Product Weight: 30235650.00
Zone: West      Regional Zone: Zone 1   Total Product Weight: 10638197.00
Zone: West      Regional Zone: Zone 2   Total Product Weight: 15146537.00
Zone: West      Regional Zone: Zone 3   Total Product Weight: 20617692.00
Zone: West      Regional Zone: Zone 4   Total Product Weight: 43804669.00
Zone: West      Regional Zone: Zone 5   Total Product Weight: 32242727.00
Zone: West      Regional Zone: Zone 6   Total Product Weight: 52661774.00
hadoop@hadoop-VirtualBox:~/Documents/Assignment$
```

## 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

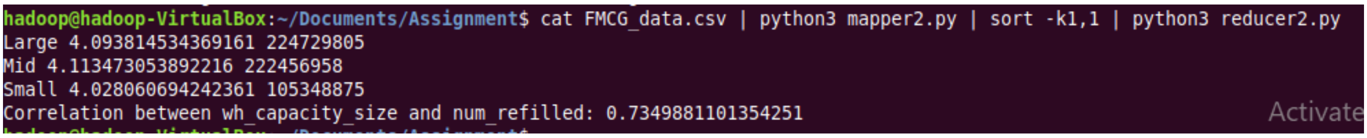
```
#!/usr/bin/python3
"""mapper2.py"""
import sys
# input comes from standard input
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # If the line is not empty
    if line:
        columns = line.split(',')
        if columns:
            capacity_size = columns[3].strip()
            num_refill_req_l3m = columns[6].strip()
            ton = columns[-1].strip()

            if capacity_size != "capacity_size" and num_refill_req_l3m != "num_refill_req_l3m":
                print('%s,%s,%s' % (capacity_size, num_refill_req_l3m, ton))
```

**Reduced**

```
#!/usr/bin/python3
"""reducer2.py"""

import sys
import numpy as np

data={}
encode = {'Large':3,'Mid':2, 'Small':1}

# input comes from STDIN

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

    capacity, refill, ton = line.split(",")
    try:
        refill = int(refill)
        ton = int(ton)
    except:
        continue

    if capacity in data:
        data[capacity][0]+=refill
        data[capacity][1]+=1
        data[capacity][2]+=ton
    else:
        data[capacity]=[refill,1,0]

values=[]
sizes=[]

for k, v in data.items():
    avg = v[0]/v[1]
    values.append(avg)
    sizes.append(encode[k])
    print(f"{k} {avg} {v[2]}")


correlation_matrix = np.corrcoef(sizes, values)
```

```
correlation_xy = correlation_matrix[0, 1]

print("Correlation between wh_capacity_size and num_refilled:", correlation_xy)
```

#### Ouput



```
hadoop@hadoop-VirtualBox:~/Documents/Assignment$ cat FMCG_data.csv | python3 mapper2.py | sort -k1,1 | python3 reducer2.py
Large 4.093814534369161 224729805
Mid 4.113473053892216 222456958
Small 4.028060694242361 105348875
Correlation between wh_capacity_size and num_refilled: 0.7349881101354251
```

## 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

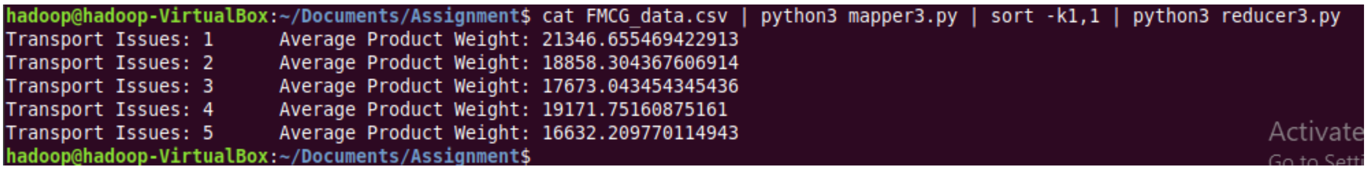
```
#!/usr/bin/python3
"""mapper2.py"""
import sys

# input comes from standard input

for line in sys.stdin:
        line = line.strip()
        if line:
                columns = line.split(',')
                if columns[7] != "transport_issue_l1y," or columns[-1] !=
"product_wg_ton":
                        transport_issue = columns[7].strip()
                        product_wg = columns[-1].strip()
                        print('%s,%s' % (transport_issue, product_wg))
```

#### Reducer

```
#!/usr/bin/python3
"""reducer3.py"""
import sys
from collections import defaultdict

issues = None
wt_issues = defaultdict(list)
ct_issue = defaultdict(int)
```

```
for line in sys.stdin:
    issues, weight = line.strip().split(',')
    try:
        weight = float(weight)
        if int(issues) > 0:
            wt_issues[issues].append(weight)
            ct_issue[issues] += 1
    except ValueError:
        continue

wt_issues = dict(sorted(wt_issues.items(), key=lambda x: x[0]))


for val in wt_issues:
    weights = wt_issues[val]
    average_weight = sum(weights) / len(weights)
    print(f"Transport Issues: {val}\t Average Product Weight: {average_weight}")
```

#### Output



```
hadoop@hadoop-VirtualBox:~/Documents/Assignment$ cat FMCG_data.csv | python3 mapper3.py | sort -k1,1 | python3 reducer3.py
Transport Issues: 1      Average Product Weight: 21346.655469422913
Transport Issues: 2      Average Product Weight: 18858.304367606914
Transport Issues: 3      Average Product Weight: 17673.043454345436
Transport Issues: 4      Average Product Weight: 19171.75160875161
Transport Issues: 5      Average Product Weight: 16632.209770114943
hadoop@hadoop-VirtualBox:~/Documents/Assignment$
```

## 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 weight shipped. Reduce: Aggregate the product weight by storage issue status to assess the impact.

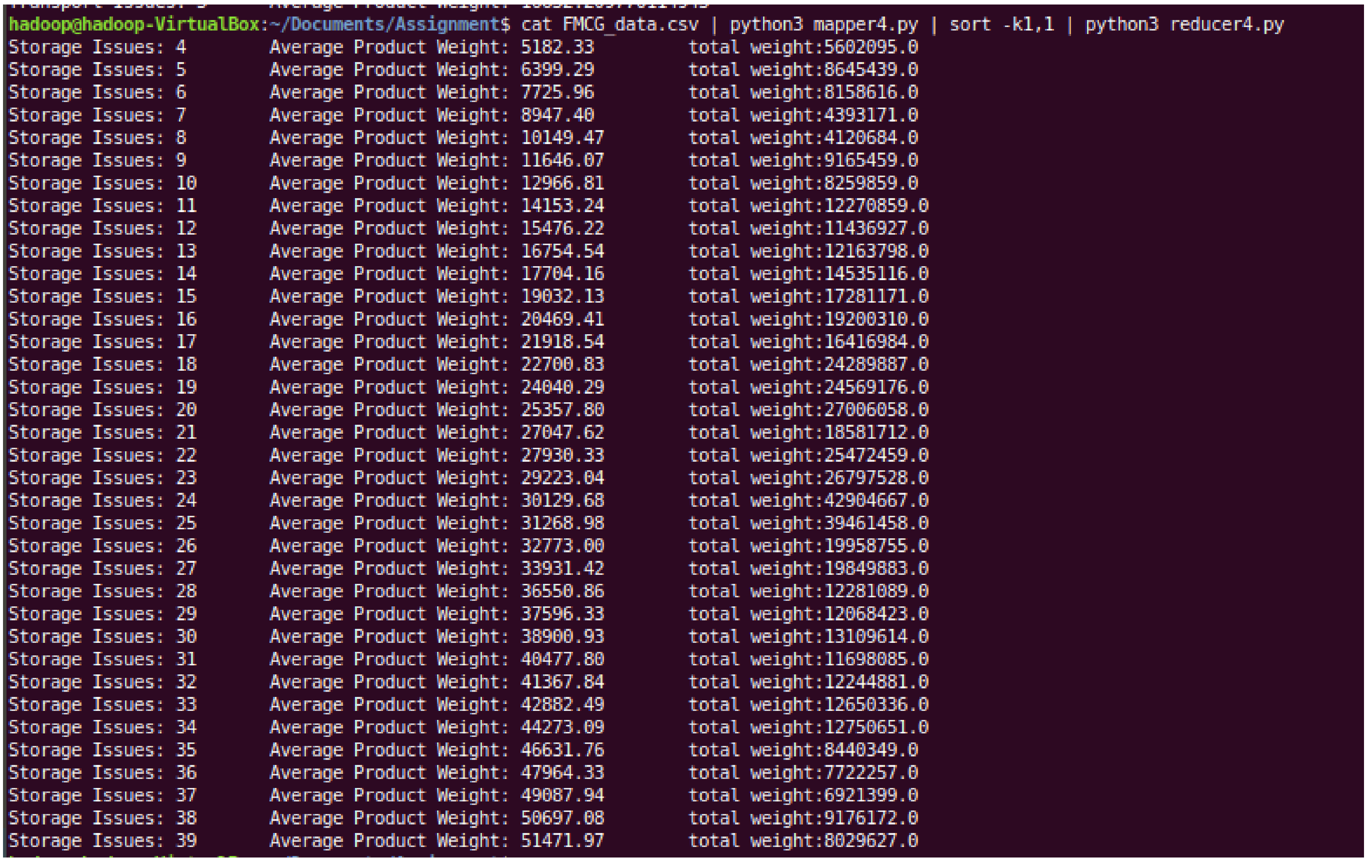
#### Mapper

```
#!/usr/bin/python3
"""mapper2.py"""
import sys

# input comes from standard input
for line in sys.stdin:
        line = line.strip()
        if line:
                columns = line.split(',')
                if columns[-6] != "storage_issue_reported_l3m,," or columns[-1] !=
"product_wg_ton":
                        storage_issue = columns[-6].strip()
                        product_wg = columns[-1].strip()
                        print('%s,%s' % (storage_issue, product_wg))
```

### Reducer

```
#!/usr/bin/python3
"""reducer4.py"""
import sys
from collections import defaultdict

issues = None
wt_issue = defaultdict(float)
ct_issue = defaultdict(float)


for line in sys.stdin:
    issues, weight = line.strip().split(',')
    try:
        issues = int(issues)
        weight = float(weight)
        if issues > 0:
            wt_issue[issues] += weight
            ct_issue[issues] += 1
    except ValueError:
            continue

wt_issue = dict(sorted(wt_issue.items(), key=lambda x: x[0]))

for val in wt_issue:
    print(f"Storage Issues: {val}\t Average Product Weight:
{wt_issue[val]/ct_issue[val]:.2f}\t total weight:{wt_issue[val]}")
```

### Ouput



```
hadoop@hadoop-VirtualBox:~/Documents/Assignment$ cat FMCG_data.csv | python3 mapper4.py | sort -k1,1 | python3 reducer4.py
Storage Issues: 4         Average Product Weight: 5182.33         total weight:5602095.0
Storage Issues: 5         Average Product Weight: 6399.29         total weight:8645439.0
Storage Issues: 6         Average Product Weight: 7725.96         total weight:8158616.0
Storage Issues: 7         Average Product Weight: 8947.40         total weight:4393171.0
Storage Issues: 8         Average Product Weight: 10149.47        total weight:4120684.0
Storage Issues: 9         Average Product Weight: 11646.07        total weight:9165459.0
Storage Issues: 10        Average Product Weight: 12966.81        total weight:8259859.0
Storage Issues: 11        Average Product Weight: 14153.24        total weight:12270859.0
Storage Issues: 12        Average Product Weight: 15476.22        total weight:11436927.0
Storage Issues: 13        Average Product Weight: 16754.54        total weight:12163798.0
Storage Issues: 14        Average Product Weight: 17704.16        total weight:14535116.0
Storage Issues: 15        Average Product Weight: 19032.13        total weight:17281171.0
Storage Issues: 16        Average Product Weight: 20469.41        total weight:19200310.0
Storage Issues: 17        Average Product Weight: 21918.54        total weight:16416984.0
Storage Issues: 18        Average Product Weight: 22700.83        total weight:24289887.0
Storage Issues: 19        Average Product Weight: 24040.29        total weight:24569176.0
Storage Issues: 20        Average Product Weight: 25357.80        total weight:27006058.0
Storage Issues: 21        Average Product Weight: 27047.62        total weight:18581712.0
Storage Issues: 22        Average Product Weight: 27930.33        total weight:25472459.0
Storage Issues: 23        Average Product Weight: 29223.04        total weight:26797528.0
Storage Issues: 24        Average Product Weight: 30129.68        total weight:42904667.0
Storage Issues: 25        Average Product Weight: 31268.98        total weight:39461458.0
Storage Issues: 26        Average Product Weight: 32773.00        total weight:19958755.0
Storage Issues: 27        Average Product Weight: 33931.42        total weight:19849883.0
Storage Issues: 28        Average Product Weight: 36550.86        total weight:12281089.0
Storage Issues: 29        Average Product Weight: 37596.33        total weight:12068423.0
Storage Issues: 30        Average Product Weight: 38900.93        total weight:13109614.0
Storage Issues: 31        Average Product Weight: 40477.80        total weight:11698085.0
Storage Issues: 32        Average Product Weight: 41367.84        total weight:12244881.0
Storage Issues: 33        Average Product Weight: 42882.49        total weight:12650336.0
Storage Issues: 34        Average Product Weight: 44273.09        total weight:12750651.0
Storage Issues: 35        Average Product Weight: 46631.76        total weight:8440349.0
Storage Issues: 36        Average Product Weight: 47964.33        total weight:7722257.0
Storage Issues: 37        Average Product Weight: 49087.94        total weight:6921399.0
Storage Issues: 38        Average Product Weight: 50697.08        total weight:9176172.0
Storage Issues: 39        Average Product Weight: 51471.97        total weight:8029627.0
```