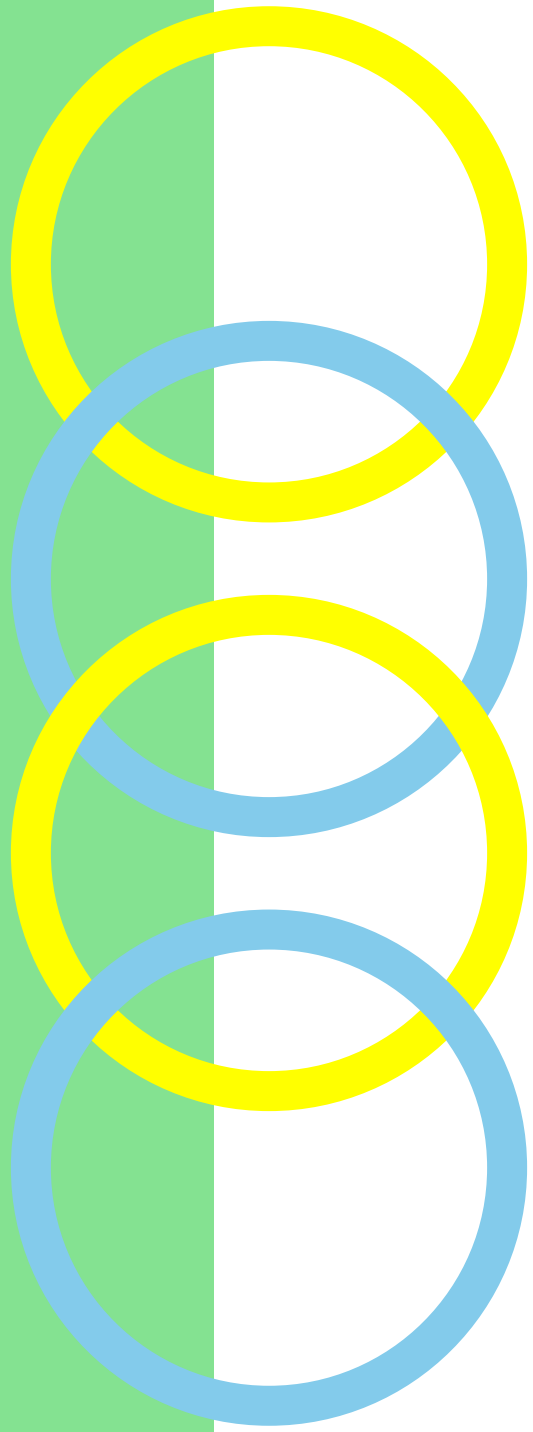


Warehouse Messy Dataset Analysis

- Kamal MUSTAFAYEV
- Elvin MAHMUDZADA
- Huseyn BABAYEV

Abstract

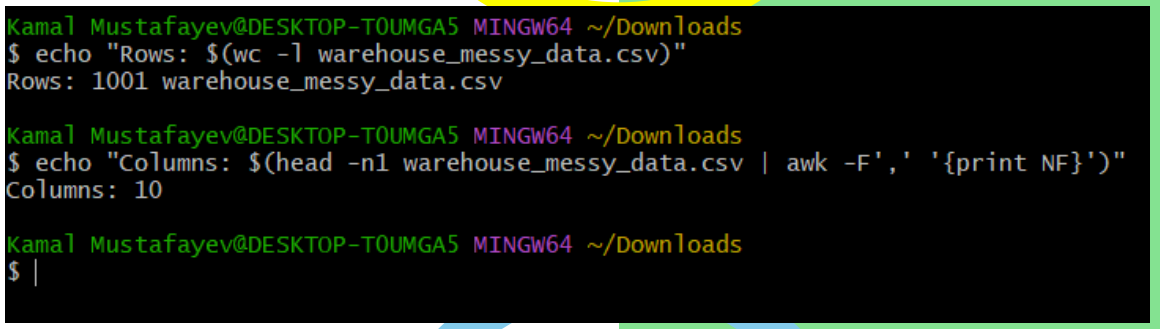
Dataset: Messy warehouse inventory CSV (product ID, name, category, etc.) from GitHub, with missing values, inconsistent types,etc.
Question: How to clean disordered data and extract business insights for better inventory management ?
Method: Bash for data prep (impute numericals with mean, categoricals with mode, dates with latest; standardize types), compute stats (mean, mode, min, max, totals by category/warehouse), gnuplot for visualizations.
Findings: Analysis revealed near-zero correlation between quantity and price, highlighting independent factors in stock valuation; identified top suppliers and category breakdowns for optimizing supply chains, with visualizations showing stock imbalances across warehouses and locations to reduce overstocking and shortages.

Motivation

Messy Inventory Data Challenge: Warehouse records are disorganized with missing values and inconsistent categories, leading to inaccurate stock tracking and decision-making.
Business Impact: Inefficiencies cause stockouts, overstocking, wasted resources, and potential revenue losses - critical in competitive retail and logistics sectors.
Problem to Solve: Clean and analyze the data to uncover patterns in stock levels and pricing, enabling optimized inventory management.
Value of Insights: Provides actionable recommendations for warehouse managers and suppliers to reduce costs, improve efficiency, and enhance supply chain reliability.
For Whom: Benefits business owners, inventory teams, and analysts seeking data-driven strategies in a fast-paced market.

Dataset

Dataset Overview: Messy warehouse inventory data in CSV format, containing details on products stored across multiple warehouses.
Data Contents: Includes 10 columns: Product ID, Product Name, Category, Warehouse, Location, Quantity, Price, Supplier, Status and Last Restocked date.
Data Size: 1001 rows (including header), representing approximately 1000 product entries; 10 columns total.
Source: Sourced from GitHub at https://github.com/eyowhite/Messy-dataset/blob/main/warehouse_messy_data.csv.
Key Characteristics: Features inconsistencies like missing values and mismatched data types, requiring cleaning for analysis.



Dataset

1	Product ID	Product Name	Category	Warehouse	Location	Quantity	Price	Supplier	Status	Last Restocked
2	1102	gadget y	ELECTRONICS	Warehouse 2	Aisle 1	300	9.99	Supplier C	In Stock	NaN
3	1435	gadget y	ELECTRONICS	Warehouse 2	Aisle 4	two hundred	19.99	Supplier C	Out of Stock	NaN
4	1860	widget a	CLOTHING	Warehouse 2	Aisle 3	100	19.99	Supplier B	In Stock	20/12/2022
5	1270	gadget z	TOYS	Warehouse 2	Aisle 4	50	49.99	Supplier B	In Stock	20/12/2022
6	1106	widget a	FURNITURE	Warehouse 3	Aisle 3	two hundred	9.99	Supplier D	Out of Stock	25/04/2023
7	1071	widget b	CLOTHING	Warehouse 3	Aisle 5	300	NaN	Supplier A	In Stock	20/12/2022
8	1700	widget a	CLOTHING	Warehouse 2	Aisle 2	two hundred	49.99	Supplier B	In Stock	20/12/2022
9	1020	widget c	CLOTHING	Warehouse 1	Aisle 5	two hundred	9.99	Supplier D	Out of Stock	20/12/2022
10	1614	gadget y	ELECTRONICS	Warehouse 3	Aisle 3	300	9.99	Supplier B	Out of Stock	05/03/2023
11	1121	widget b	TOYS	Warehouse 1	Aisle 2	50	19.99	Supplier C	Out of Stock	20/12/2022
12	1466	widget a	CLOTHING	Warehouse 3	Aisle 1	NaN	49.99	Supplier B	In Stock	15/01/2023
13	1214	gadget z	FURNITURE	Warehouse 2	Aisle 1	50	9.99	Supplier A	Low Stock	15/01/2023
14	1330	widget c	TOYS	Warehouse 3	Aisle 4	150	9.99	Supplier B	In Stock	25/04/2023
15	1458	widget c	FURNITURE	Warehouse 2	Aisle 3	300	49.99	Supplier A	Low Stock	05/03/2023

Data Preparation and Cleaning

Initial Assessment: Identified issues like missing values in various columns, inconsistent data types (e.g., quantities or prices as strings)

Data Preparation and Cleaning STEP BY STEP:

- **Step 1**: The *awk* command trims leading and trailing whitespaces from each field in the CSV file using a *loop* and *gsub*, outputs to a temp file, then overwrites the original for safe in-place cleaning.

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ head -5 warehouse_messy_data.csv
Product ID,Product Name,Category,Warehouse,Location,Quantity,Price,Supplier,Status,Last Restocked
1102, gadget y ,ELECTRONICS,Warehouse 2,Aisle 1,300,9.99,Supplier C,In Stock,NaN
1435, gadget y ,ELECTRONICS,Warehouse 2,Aisle 4,two hundred,19.99,Supplier C,Out of Stock,NaN
1860, widget a ,CLOTHING,Warehouse 2,Aisle 3,100,19.99,Supplier B,In Stock,20/12/2022
1270, gadget z ,TOYS,Warehouse 2,Aisle 4,50,49.99,Supplier B,In Stock,20/12/2022

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' '{
    for(i=1;i<=NF;i++) gsub(/^ +| +$/,"",$i)
    print
}' OFS=',' warehouse_messy_data.csv > temp.csv && mv temp.csv warehouse_messy_data.csv
```

Data Preparation and Cleaning

- **Step 2**: Check each column for missing values (empty or "NaN") by running the command with the respective column index.

- As a result, we see that column 6 contains such values. Then we check the values to identify their types, and we find a type mismatch (some rows contain strings) :

```
Kamal Mustafayev@DESKTOP-T0UMGA5 MINGW64 ~/Downloads
$ awk -F ',' 'NR>1 {print $6}' warehouse_messy_data.csv
300
two hundred
100
50
two hundred
300
two hundred
two hundred
300
50
NaN
50
150
```

```
Kamal Mustafayev@DESKTOP-T0UMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($1=="" || $1=="NaN") print }' warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-T0UMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($2=="" || $2=="NaN") print }' warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-T0UMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($3=="" || $3=="NaN") print }' warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-T0UMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($4=="" || $4=="NaN") print }' warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-T0UMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($5=="" || $5=="NaN") print }' warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-T0UMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($6=="" || $6=="NaN") print }' warehouse_messy_data.csv
1466,widget a,CLOTHING,Warehouse 3,Aisle 1,NaN,49.99,Supplier B,In Stock,15/01/2023
1276,gadget y,CLOTHING,Warehouse 2,Aisle 1,NaN,19.99,Supplier C,Out of Stock,25/04/2023
1160,gadget y,FURNITURE,Warehouse 1,Aisle 5,NaN,29.99,Supplier B,Low Stock,NaN
1699,gadget y,CLOTHING,Warehouse 1,Aisle 1,NaN,9.99,Supplier B,Out of Stock,NaN
1189,widget a,FURNITURE,Warehouse 2,Aisle 1,NaN,29.99,Supplier A,Low Stock,25/04/2023
1686,widget b,CLOTHING,Warehouse 1,Aisle 4,NaN,NaN,Supplier A,Low Stock,05/03/2023
1875,widget b,CLOTHING,Warehouse 1,Aisle 5,NaN,49.99,Supplier D,In Stock,20/12/2022
1646,widget a,ELECTRONICS,Warehouse 1,Aisle 2,NaN,49.99,Supplier A,In Stock,15/01/2023
1387,gadget z,TOYS,Warehouse 2,Aisle 3,NaN,29.99,Supplier C,Low Stock,NaN
1600,widget c,CLOTHING,Warehouse 2,Aisle 1,NaN,29.99,Supplier D,Low Stock,20/12/2022
1775,widget b,TOYS,Warehouse 3,Aisle 5,NaN,9.99,Supplier A,In Stock,20/12/2022
1931,gadget x,TOYS,Warehouse 1,Aisle 3,NaN,49.99,Supplier D,Out of Stock,25/04/2023
1387,widget a,FURNITURE,Warehouse 2,Aisle 2,NaN,29.99,Supplier D,In Stock,NaN
1565,widget b,CLOTHING,Warehouse 3,Aisle 5,NaN,49.99,Supplier C,In Stock,20/12/2022
1105,widget c,FURNITURE,Warehouse 1,Aisle 2,NaN,49.99,Supplier B,Out of Stock,15/01/2023
1771,widget a,TOYS,Warehouse 3,Aisle 5,NaN,19.99,Supplier C,Out of Stock,15/01/2023
1161,widget a,CLOTHING,Warehouse 2,Aisle 3,NaN,19.99,Supplier C,In Stock,NaN
1862,gadget z,ELECTRONICS,Warehouse 1,Aisle 1,NaN,NaN,Supplier D,In Stock,20/12/2022
1492,widget b,CLOTHING,Warehouse 1,Aisle 5,NaN,29.99,Supplier A,Out of Stock,05/03/2023
1128,gadget y,ELECTRONICS,Warehouse 3,Aisle 3,NaN,49.99,Supplier A,In Stock,15/01/2023
1391,gadget z,FURNITURE,Warehouse 2,Aisle 3,NaN,9.99,Supplier B,In Stock,20/12/2022
1032,widget a,ELECTRONICS,Warehouse 1,Aisle 1,NaN,49.99,Supplier D,In Stock,25/04/2023
1502,gadget x,FURNITURE,Warehouse 1,Aisle 4,NaN,9.99,Supplier B,Out of Stock,15/01/2023
1804,gadget y,CLOTHING,Warehouse 2,Aisle 2,NaN,19.99,Supplier B,Low Stock,15/01/2023
1960,widget b,CLOTHING,Warehouse 1,Aisle 5,NaN,NaN,Supplier A,Out of Stock,20/12/2022
```

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 && $6!="" && $6!="NaN" && $6 !~ /^[0-9]+$/ {print $6}' warehouse_messy_data.csv | sort | uniq
two hundred

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$
```

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR==1 {print; next}
{
    if($6=="two hundred") $6=200
    print
}' OFS=',' warehouse_messy_data.csv > temp.csv && mv temp.csv warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F ',' 'NR>1 {print $6}' warehouse_messy_data.csv
300
200
100
50
200
300
200
200
300
50
NaN
```

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ mean=$(awk -F',' 'NR>1 && $6!="" && $6!="NaN" {sum+=$6; count++} END {print sum/count}' warehouse_messy_data.csv)

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' -v avg="$mean" 'NR==1 {print; next}
{
    if($6=="" || $6=="NaN") $6=avg
    print
}' OFS=',' warehouse_messy_data.csv > temp.csv && mv temp.csv warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F ',' 'NR>1 {print $6}' warehouse_messy_data.csv
300
200
100
50
200
300
200
200
300
50
161.401
50
```

- **Step 3:** Check the number of unique values in the column that are neither numeric nor empty/NaN. We find only one such value—"two hundred," which we replace with 200 and save the table.

- **Step 4:** Replace all empty/NaN values in the column with the mean value.

- At first, I made *a mistake*: although all values in the column were integers, the mean turned out to be a float. I replaced the missing values with this float, and later had to correct it by converting the replacement value to an integer to keep the column consistent.

- Fixing *mistake*:

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR==1 {print; next}
{
    if($6=="161.401") $6=161
    print
}' OFS=',' warehouse_messy_data.csv > temp.csv && mv temp.csv warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F ',' 'NR>1 {print $6}' warehouse_messy_data.csv
300
200
100
50
200
300
200
200
300
50
161
```

Data Preparation and Cleaning

Data Preparation and Cleaning

- **Step 5:** Check column 7 for empty/NaN values, then replace all such values with the mean. In this case, the mean was calculated as a float with two decimal places, matching the column's numeric type.

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($7=="" || $7=="NaN") print }' warehouse_messy_data.csv
1071,widget b,CLOTHING,Warehouse 3,Aisle 5,300,NaN,Supplier A,In Stock,20/12/2022
1087,widget a,TOYS,Warehouse 3,Aisle 3,50,NaN,Supplier C,Out of Stock,15/01/2023
1871,widget b,CLOTHING,Warehouse 2,Aisle 2,100,NaN,Supplier D,Out of Stock,25/04/2023
1308,gadget y,CLOTHING,Warehouse 1,Aisle 1,300,NaN,Supplier C,Out of Stock,25/04/2023
1459,gadget y,CLOTHING,Warehouse 3,Aisle 2,100,NaN,Supplier D,Out of Stock,20/12/2022
1058,gadget z,FURNITURE,Warehouse 1,Aisle 1,150,NaN,Supplier C,In Stock,15/01/2023
```

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ mean=$(awk -F',' 'NR>1 && $7!="" && $7!="NaN" {sum+=$7; count++} END {printf "%.2f", sum/count}' warehouse_messy_data.csv)

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' -v avg="$mean" 'NR==1 {print; next}
{
    if($7=="" || $7=="NaN") $7=avg
    print
}' OFS=',' warehouse_messy_data.csv > temp.csv && mv temp.csv warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F ',' 'NR>1 {print $7}' warehouse_messy_data.csv
9.99
19.99
19.99
49.99
9.99
28.09
```

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ head -1 warehouse_messy_data.csv
Product ID,Product Name,Category,Warehouse,Location,Quantity,Price,Supplier,Status,Last Restocked

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($8=="" || $8=="NaN") print }' warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($9=="" || $9=="NaN") print }' warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {if($10=="" || $10=="NaN") print }' warehouse_messy_data.csv
1102,gadget y,ELECTRONICS,Warehouse 2,Aisle 1,300,9.99,Supplier C,In Stock,NaN
1435,gadget y,ELECTRONICS,Warehouse 2,Aisle 4,200,19.99,Supplier C,Out of Stock,NaN
1385,gadget z,CLOTHING,Warehouse 1,Aisle 4,50,19.99,Supplier B,In Stock,NaN
1160,gadget y,FURNITURE,Warehouse 1,Aisle 5,161,29.99,Supplier B,Low Stock,NaN
```

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {
    if($10!="" && $10!="NaN") {
        split($10, d, "/")
        if(length(d)!=3 || d[1]<1 || d[1]>31 || d[2]<1 || d[2]>12 || d[3]<1970 || d[3]>2025)
            print NR, $10
    }
}' warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$
```

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ latest=$(awk -F',' 'NR>1 && $10!="" && $10!="NaN" {
    split($10,d,"/")
    date_val=d[3] d[2] d[1]
    if(date_val>max) max=date_val
} END {
    if(max!="") {
        printf "%s/%s/%s", substr(max,7,2), substr(max,5,2), substr(max,1,4)
    }
}' warehouse_messy_data.csv)

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ echo $latest
25/04/2023

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' -v fill="$latest" 'NR==1 {print; next}
{
    if($10=="" || $10=="NaN") $10=fill
    print
}' OFS=',' warehouse_messy_data.csv > temp.csv && mv temp.csv warehouse_messy_data.csv

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F ',' 'NR>1 {print $10}' warehouse_messy_data.csv
25/04/2023
25/04/2023
20/12/2022
20/12/2022
25/04/2023
```

Data Preparation and Cleaning

- **Step 6:** Check columns 8, 9, and 10 for empty/NaN values. Missing values are found in column 10 (date type). After verifying that all existing dates are correctly formatted, we replace the missing values with the latest date.

- **split($10,d,"/")** -> Split the date in column 10 into array d, where: d[1] = day, d[2] = month, d[3] = year
- **date_val=d[3] d[2] d[1]** -> Concatenate into YYYYMMDD format (string comparison works correctly here).
- **substr(max,7,2)** -> extracts day
  substr(max,5,2) -> extracts month
  substr(max,1,4) -> extracts year

Data Preparation and Cleaning

- **Step 7:** Check all the unique values in each column and their counts to see if there are any type/value mismatches left to fix.

- After reviewing all columns and their unique values, we can conclude that the dataset is now fully cleaned and prepared for further analysis and manipulation.

```
Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {count[$2]++} END {for(entry in count) print entry, count[entry]}' warehouse_messy_data.csv
gadget x 133
gadget y 177
gadget z 169
widget a 176
widget b 170
widget c 175

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {count[$3]++} END {for(entry in count) print entry, count[entry]}' warehouse_messy_data.csv
ELECTRONICS 248
TOYS 230
CLOTHING 257
FURNITURE 265

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {count[$4]++} END {for(entry in count) print entry, count[entry]}' warehouse_messy_data.csv
Warehouse 1 349
Warehouse 2 332
Warehouse 3 319

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {count[$5]++} END {for(entry in count) print entry, count[entry]}' warehouse_messy_data.csv
Aisle 1 210
Aisle 2 181
Aisle 3 211
Aisle 4 199
Aisle 5 199

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {count[$6]++} END {for(entry in count) print entry, count[entry]}' warehouse_messy_data.csv
50 169
100 161
150 175
161 158
200 160
300 177

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {count[$7]++} END {for(entry in count) print entry, count[entry]}' warehouse_messy_data.csv
9.99 181
29.99 211
28.09 207
49.99 204
19.99 197

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {count[$8]++} END {for(entry in count) print entry, count[entry]}' warehouse_messy_data.csv
Supplier A 244
Supplier B 288
Supplier C 232
Supplier D 236

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {count[$9]++} END {for(entry in count) print entry, count[entry]}' warehouse_messy_data.csv
In Stock 340
Out of Stock 332
Low Stock 328

Kamal Mustafayev@DESKTOP-TOUMGA5 MINGW64 ~/Downloads
$ awk -F',' 'NR>1 {count[$10]++} END {for(entry in count) print entry, count[entry]}' warehouse_messy_data.csv
15/01/2023 199
05/03/2023 191
25/04/2023 392
20/12/2022 218
```

Research Questions

- How can statistical analysis of warehouse inventory data provide insights into stock availability, category performance, and supplier efficiency?
- What patterns and correlations exist between item quantities, prices, and stock status in warehouse operations?
- How do stock statuses distribute within product categories and suppliers, and what does this signify for inventory health?
- Which suppliers and warehouses contribute most significantly to total quantity and average pricing, highlighting operational priorities?

Methods

- **Data Preparation and Cleaning:** Handled missing values, inconsistencies, and duplicates with Bash + AWK. Numerical gaps were filled with column means to preserve central tendency and avoid bias from arbitrary defaults; categorical/date gaps with modes or latest dates to keep completeness for time-sensitive insights. Textual entries in numeric fields were standardized for reliable computations. Lightweight tools ensured reproducibility without heavy dependencies.
- **Statistical Analysis:** Used AWK/shell for descriptive stats to detect shortages/overstocks (simple and transparent), group-wise aggregations to expose supplier/warehouse performance (directly tied to business questions), and correlations to show independent drivers. These methods were chosen for clarity and efficiency in exploratory analysis without overcomplicated models.
- **Visualization:** GNUplot produced scatter plots (showing weak quantity–price link), bar charts (ranking suppliers by quantity), and heatmaps (locating warehouse imbalances). It was selected for its speed, reproducibility, and ability to turn command-line outputs into clear visuals that non-technical staff can interpret for decisions.

Findings: Basic Statistics

Quantity & Price Range (min–max, avg)

Min–max quantities show products at risk of shortage or overstock in the warehouse. Price range highlights differences between bulk goods and premium items. This helps manage stock levels and supplier pricing strategies.

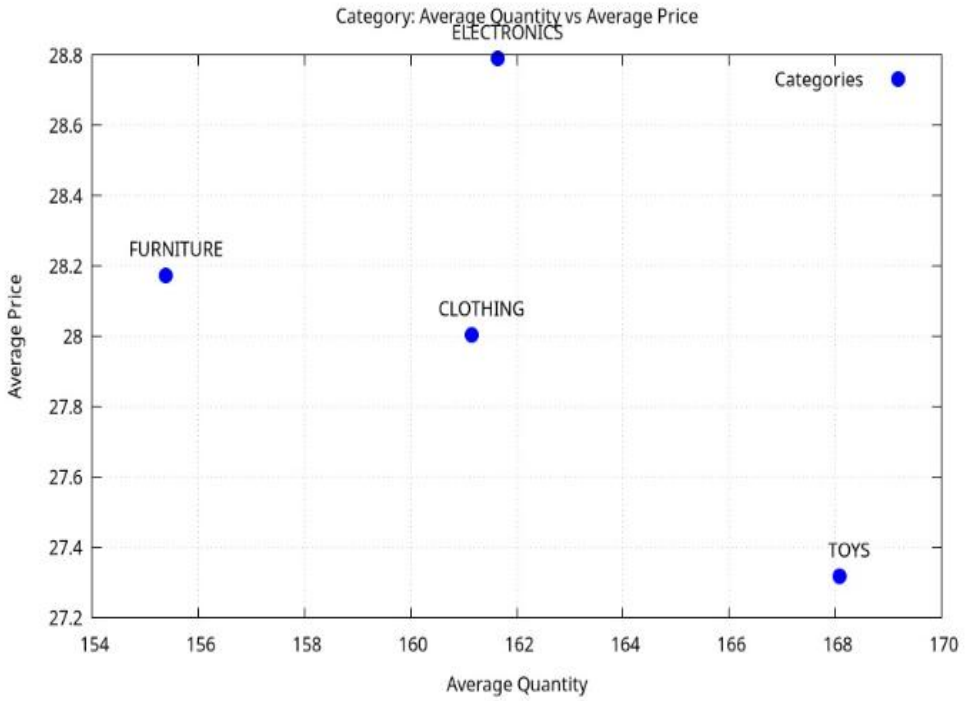
```
$ awk -F, 'NR>1 {
  if(minq==""){minq=maxq=$6; minp=maxp=$7}
  if($6<minq) minq=$6; if($6>maxq) maxq=$6
  if($7<minp) minp=$7; if($7>maxp) maxp=$7
  sumq+=$6; sump+=$7; count++
}
END {
  print "Quantity Min," minq > "quantity_price_stats.csv"
  print "Quantity Max," maxq >> "quantity_price_stats.csv"
  print "Quantity Avg," sumq/count >> "quantity_price_stats.csv"
  print "Price Min," minp >> "quantity_price_stats.csv"
  print "Price Max," maxp >> "quantity_price_stats.csv"
  print "Price Avg," sump/count >> "quantity_price_stats.csv"
}' warehouse_messy_data.csv
```

Aggregation by Category or Warehouse (sum & average of Quantity and Price)

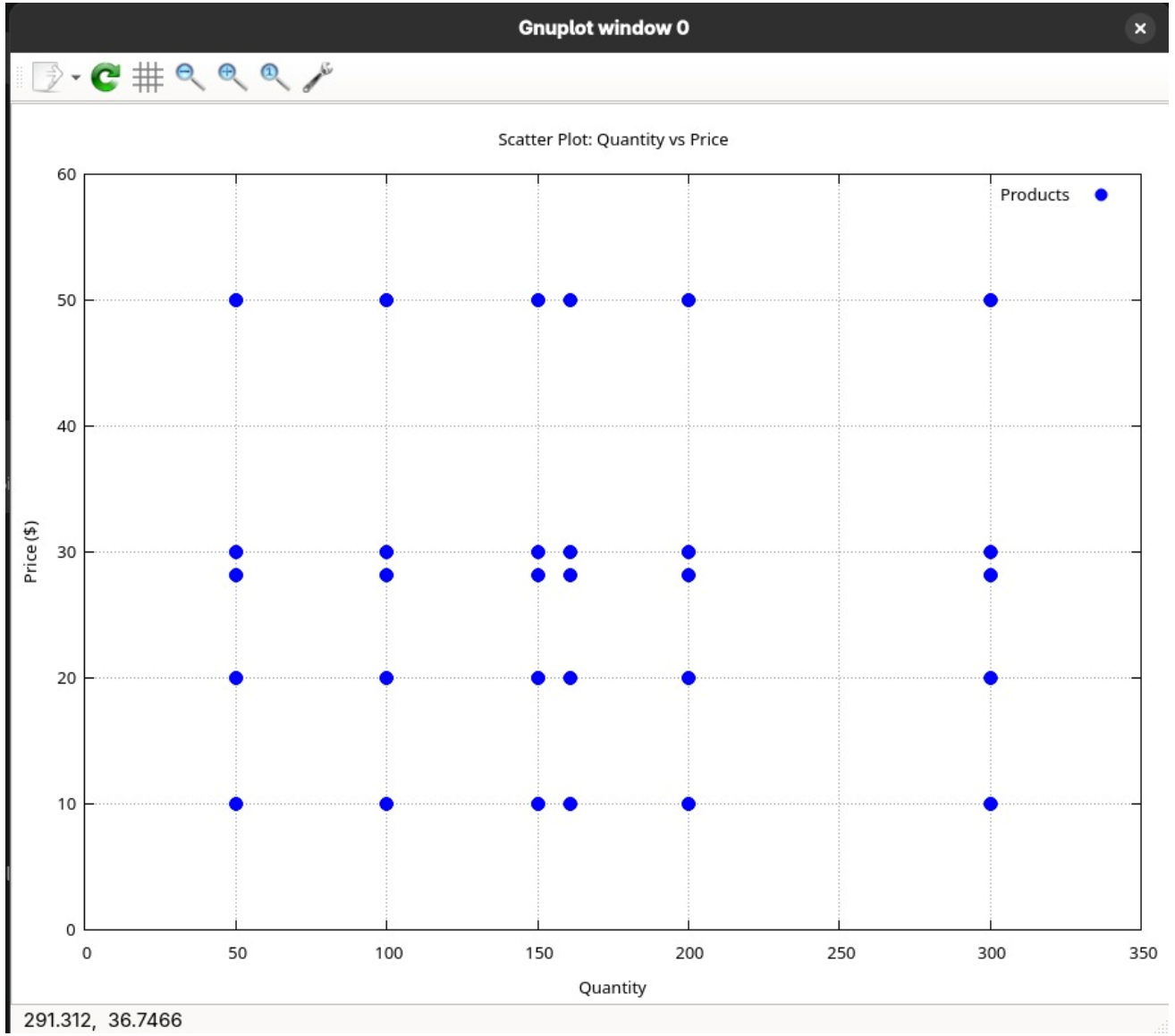

```
hbaba@DESKTOP-CGF38NE MINGW64 ~/Downloads (master)
$ awk -F, 'NR>1 {
  qty_sum_cat[$3]+=$6; price_sum_cat[$3]+=$7; count_cat[$3]++
  qty_sum_wh[$4]+=$6; price_sum_wh[$4]+=$7; count_wh[$4]++
}
END {
  print "Category,TotalQuantity,AvgQuantity,TotalPrice,AvgPrice" > "category_agg.csv"
  for (c in count_cat) print c "," qty_sum_cat[c] "," qty_sum_cat[c]/count_cat[c] "," price_su
m_cat[c] "," price_sum_cat[c]/count_cat[c] >> "category_agg.csv"

  print "Warehouse,TotalQuantity,AvgQuantity,TotalPrice,AvgPrice" > "warehouse_agg.csv"
  for (w in count_wh) print w "," qty_sum_wh[w] "," qty_sum_wh[w]/count_wh[w] "," price_sum_wh
[w] "," price_sum_wh[w]/count_wh[w] >> "warehouse_agg.csv"
}' warehouse_messy_data.csv
```

Sum & Average of Quantity and Price by Category



Here is the visualization of quantity vs price scatter plot.
As we can see the correlation between them is nearly zero

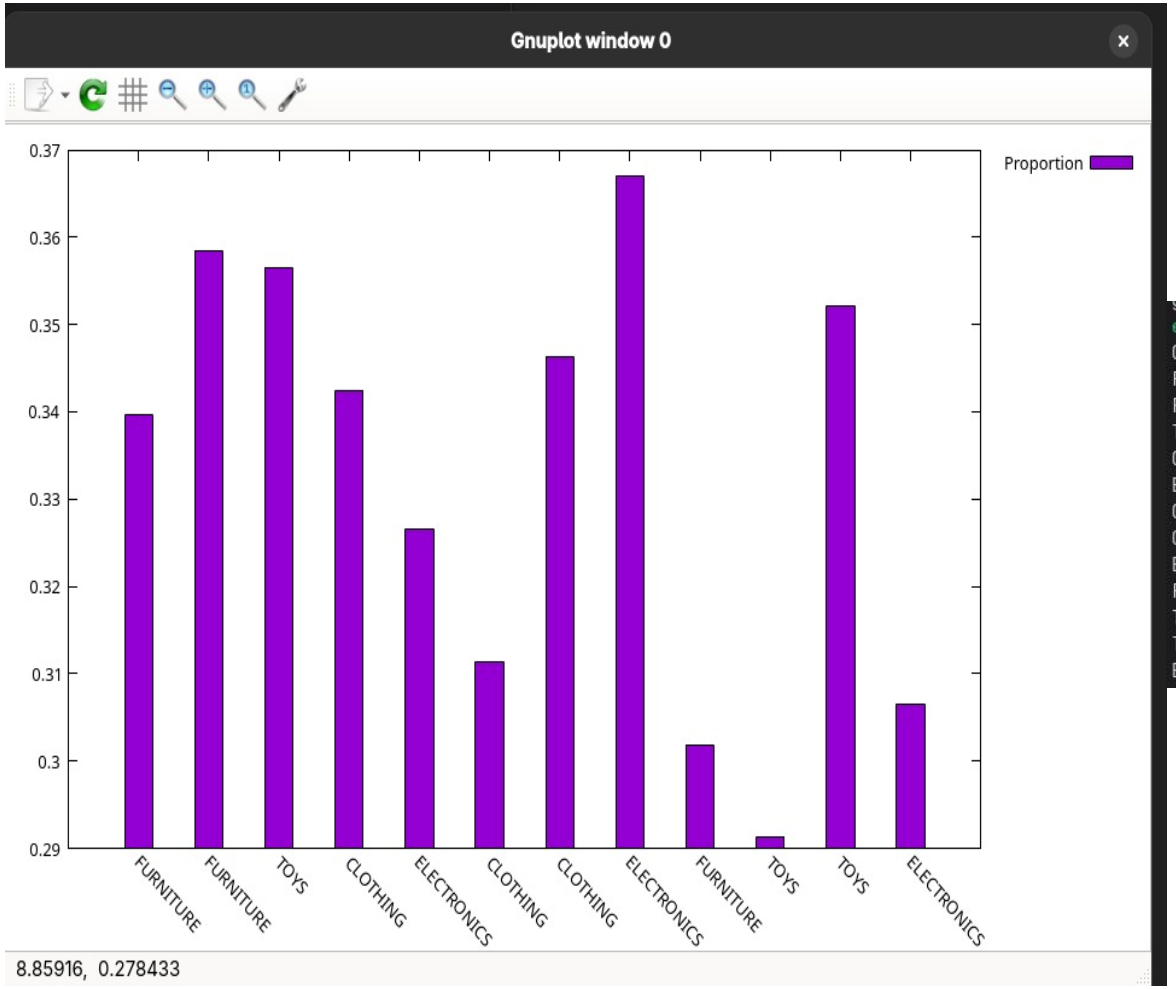



```
elvinmahmudzada@fedora:~/Downloads/warehouse-analysis$ cat correlation_qty_price.csv
Correlation_Quantity_Price,-0.0516874
elvinmahmudzada@fedora:~/Downloads/warehouse-analysis$
```

Correlation and Relationships:

- *Why it matters:* Shows the breakdown of inventory per category, highlighting which are well-stocked and which face shortages. Helps with **inventory health**, **demand planning**, and **warehouse management**, allowing managers to prioritize replenishment and improve supply chain efficiency. For example, if ELECTRONICS is 60% In Stock and 40% Out of Stock while TOYS is 90% In Stock, the warehouse can investigate supply issues or reallocate stock. This relative measure is more informative than total counts for strategic planning.

```
hbaba@DESKTOP-CGF38NE MINGW64 ~/Downloads (master)
$  awk -F, '
NR > 1 {
  status_cat[$3 FS $9]++
  total_cat[$3]++
}
END {
  # Print CSV header for gnuplot
  print "Category,Status,Proportion" > "status_proportion_per_category_v2.csv"
  for (k in status_cat) {
    split(k, arr, FS)
    cat = arr[1]
    stat = arr[2]
    prop = status_cat[k] / total_cat[cat]
    # Print category, status and proportion for v2 file
    print cat "," stat "," prop >> "status_proportion_per_category_v2.csv"
  }
}' warehouse_messy_data.csv
```



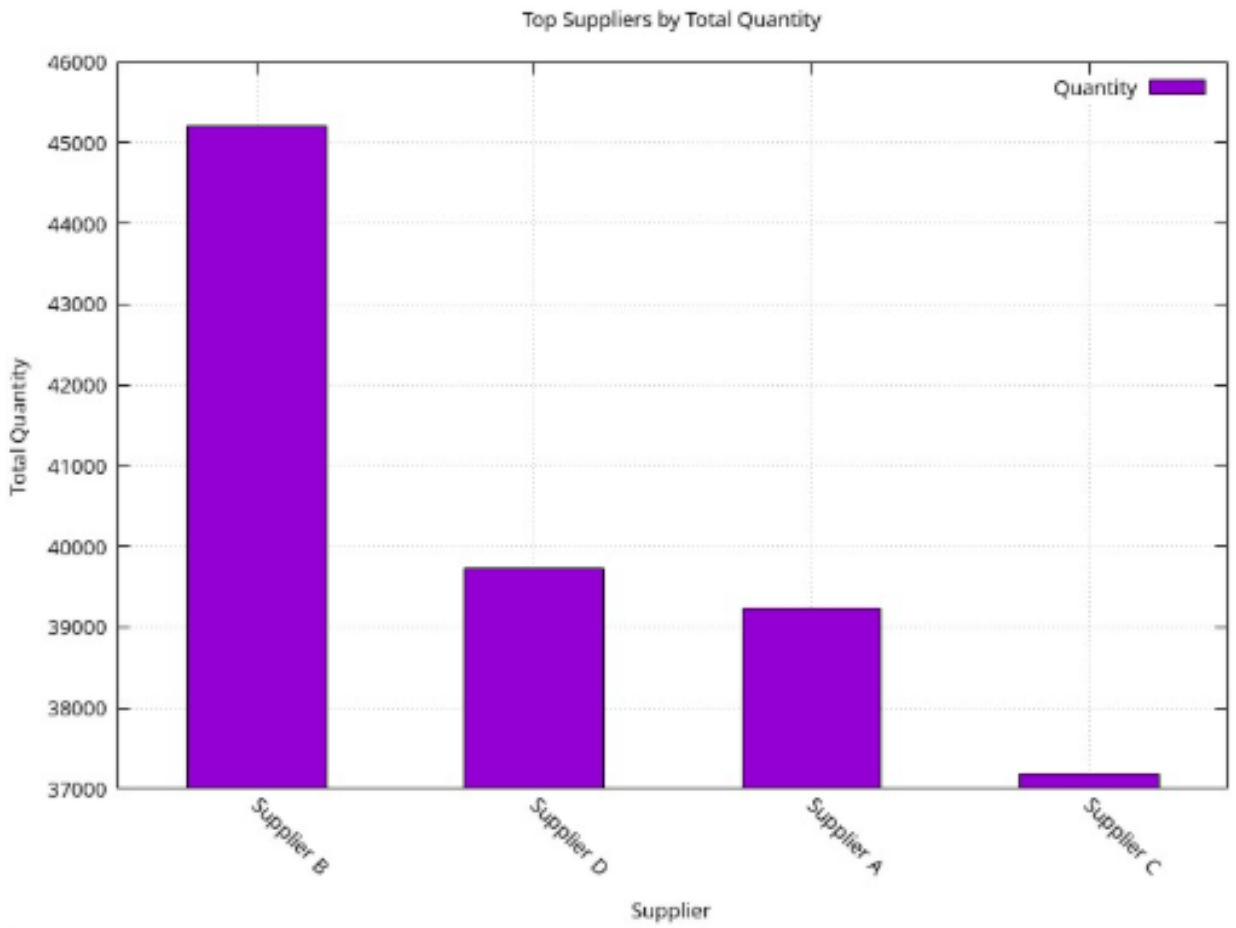

```
elvinmahmudzada@fedora:~/Downloads/warehouse-analysis$ cat status_proportion_per_category_v2.csv
Category,Status,Proportion
FURNITURE,Low Stock,0.339623
FURNITURE,Out of Stock,0.358491
TOYS,Low Stock,0.356522
CLOTHING,In Stock,0.342412
ELECTRONICS,Out of Stock,0.326613
CLOTHING,Low Stock,0.311284
CLOTHING,Out of Stock,0.346304
ELECTRONICS,In Stock,0.366935
FURNITURE,In Stock,0.301887
TOYS,Out of Stock,0.291304
TOYS,In Stock,0.352174
ELECTRONICS,Low Stock,0.306452
```

Top suppliers by total Quantity supplied

- *Why it matters:* Identifying the suppliers who provide the largest quantities helps understand which partners are **most critical to warehouse operations**. Visualizing this data highlights **supply concentration**—for example, if a few suppliers dominate, the warehouse may be vulnerable if one fails. It also guides **negotiations, stock planning, and risk management**, ensuring that high-volume suppliers are monitored and managed carefully.

```
hbaba@DESKTOP-CGF38NE MINGW64 ~/Downloads (master)
$ awk -F, 'NR>1 {
  qty_sup[$8] += $6
}
END {
  print "Supplier,TotalQuantity" > "top_suppliers.csv"
  for (sup in qty_sup) print sup "," qty_sup[sup] >> "top_suppliers.csv"
}' warehouse_messy_data.csv

# To get top suppliers sorted by quantity (descending) - do separately in Bash:
sort -t, -k2 -nr top_suppliers.csv > top_suppliers_sorted.csv
```



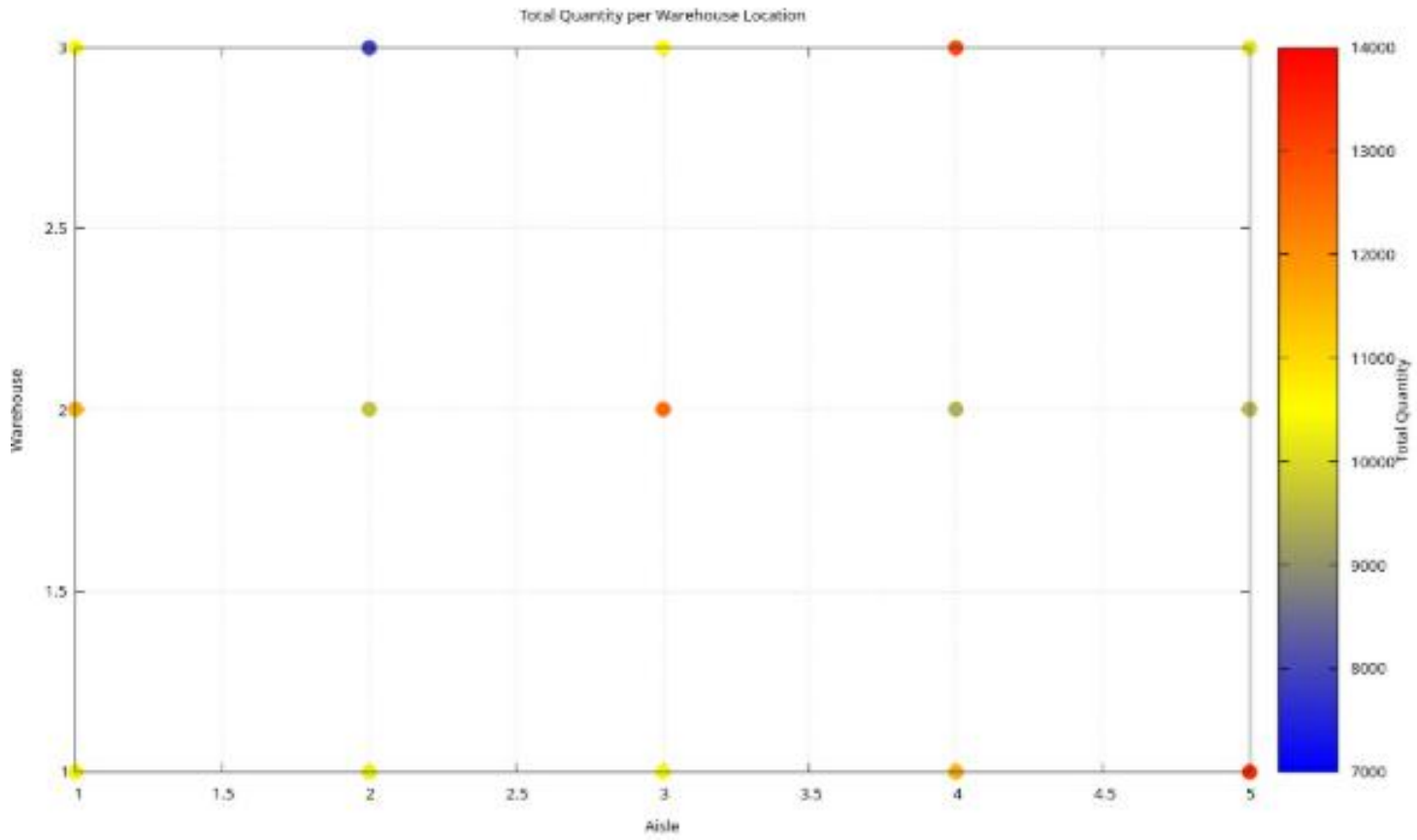
- *Outcome with visualization:* A bar chart of top suppliers makes it easy to spot **key contributors**, compare total quantities at a glance, and plan inventory or sourcing strategies accordingly.

```
hbaba@DESKTOP-CGF38NE MINGW64 ~/Downloads (master)
$ awk -F, 'NR>1 {
  sum_qty_wh_loc[$4 FS $5]+=$6;
}
END {
  print "Warehouse,Location,TotalQuantity" > "total_qty_wh_loc.csv";
  for (key in sum_qty_wh_loc) {
    split(key, arr, FS);
    wh=arr[1];
    loc=arr[2];
    print wh "," loc "," sum_qty_wh_loc[key] >> "total_qty_wh_loc.csv";
  }
}' warehouse_messy_data.csv
```

Total Quantity by Warehouse and Location

- Why it matters: Summing quantities by both warehouse and location shows how stock is distributed spatially across the warehouse system. A heatmap makes it easy to see which locations are heavily stocked and which are understocked, helping managers balance inventory, optimize storage, and plan replenishment.

- The heatmap clearly highlights areas of high and low stock at a glance, enabling quick decisions to **reallocate products** or **adjust supply flow** between warehouses and locations.



Limitations of Our Findings

- The analysis relies on the provided warehouse dataset, which may have incomplete or inaccurate data due to human error in inventory tracking or data entry limitations. This can affect the precision of statistics like counts and averages.
- The dataset only captures a snapshot of inventory status and quantities at a given time. It does not reflect real-time dynamics such as daily stock movements, order flows, or supply disruptions.
- Some variables such as supplier reliability, shipment delays, or product demand fluctuations are not included, which limits deeper causal analysis or forecasting.
- Proportions of stock status within categories or suppliers highlight relative availability but do not capture absolute sales or revenue impact, which are critical for business decisions.
- Correlation estimates between quantity and price assume linear relationships and do not account for other influencing factors like promotions or seasonality.
- Our work focuses on descriptive statistics and simple aggregations; advanced modeling or predictive analytics would require additional data and tools.

Conclusions

- The statistical analysis revealed detailed inventory patterns, including distribution summaries, unique counts, and correlations, essential for understanding warehouse operations.
- Stock status proportions within categories and suppliers highlight potential supply chain issues, aiding inventory prioritization and replenishment planning.
- Aggregated trends by category, supplier, warehouse, and location help identify high-performing and underperforming areas, informing resource allocation.
- Correlation between quantity and price informs pricing strategies and inventory valuation considerations.
- The clean and structured data allowed seamless integration with gnuplot visualizations for actionable presentations supporting decision-making.

References

- https://www.gnu.org/software/gawk/manual/
- https://linuxconfig.org/bash-scripting-tutorial-for-beginners
- https://earthly.dev/blog/awk-examples/
- https://stackoverflow.com/questions/8490500/redirect-command-output-with-awk
- https://ctcms-uq.github.io/data_tutorials/gnuplot.html