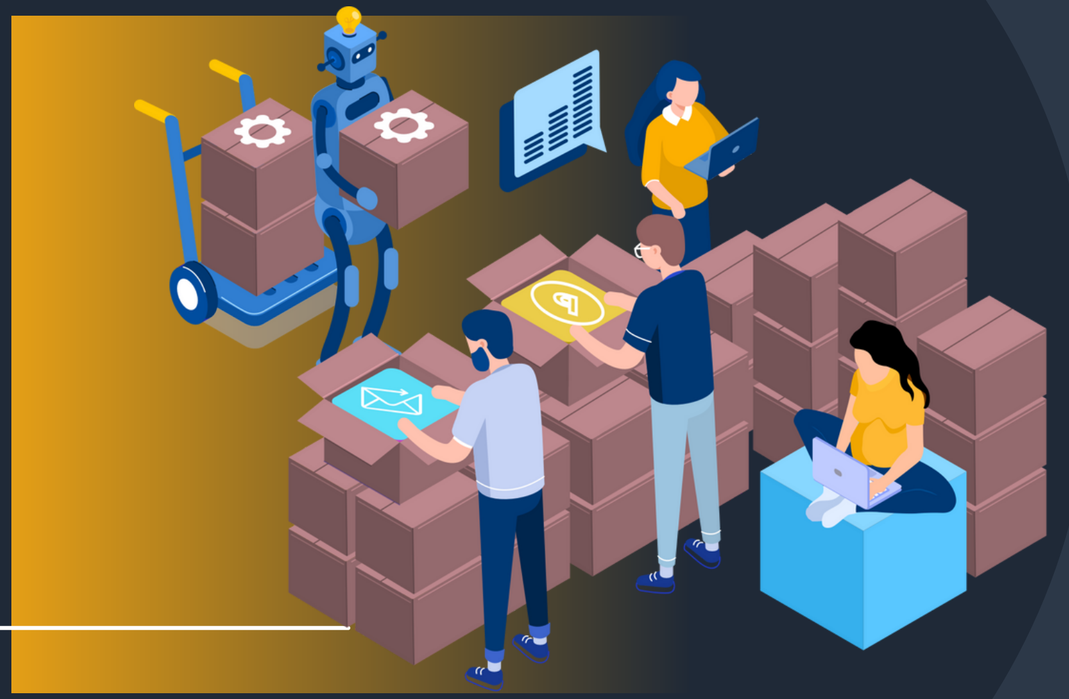
INVENTORY MANAGEMENT

USING POWER BI



STEPS FOR CREATING DASHBOARD USING POWER BI

- In power BI-> Upload Excel File -->GET DATA
- Performing ETL
- Performing DAX Operations
- Visualizations

VISUALIZATIONS

Stack Average Chart --> shows the Average, Cummulative of ABC Corporation

Cards, Gauge -> shows the requirement according to particular areas

Tables, Matrix and Line Charts for Distributions, Stocks Reports and Turnover Ratio.

Different Types of DAX Functions to analyse the actual data

With the help of DAX , create calculated columns and rows (combining dates, etc)

INVENTORY MANAGEMENT

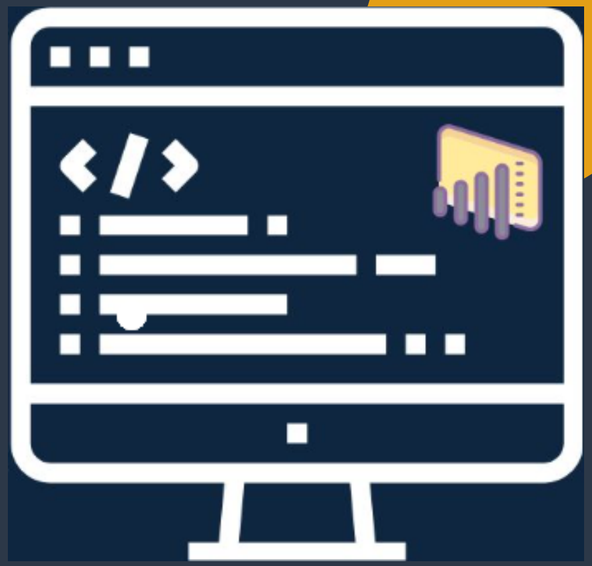
For any Warehouse In Inventory Management

---> 2 Main Tables are Important,
What is **Stock Quantity**
What is **Past Orders**

In Stocks Table, Adding New Column

Annual Sale Quantity -> Represents Annual Sales for Last 1 Year

```
Annual Sale Quantity = CALCULATE(
    SUM('Past Orders'[Order Quantity]),
    FILTER('Past Orders',
    'Past Orders'[SKU ID]='Stock'[SKU ID] &&
    'Past Orders'[Order Date] >= DATE(2020, 6, 14) - 365
))
```

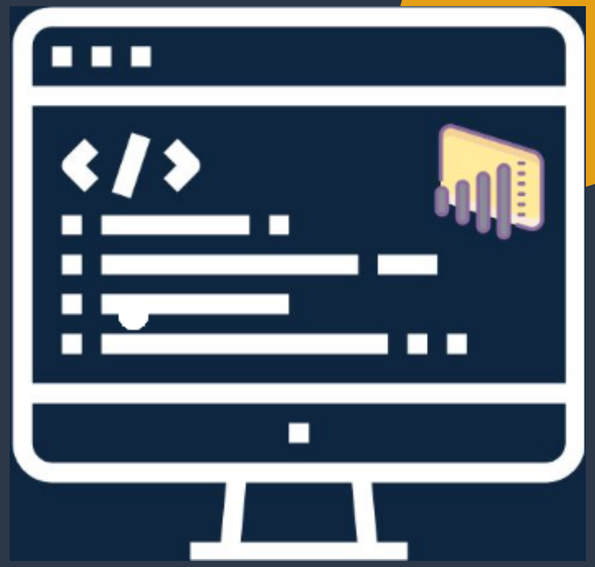


filter --> Used to filter the past order from 1 Year only.

In Stocks Table, Adding New Column

Annual Revenue -> Represents the Annual Revenue

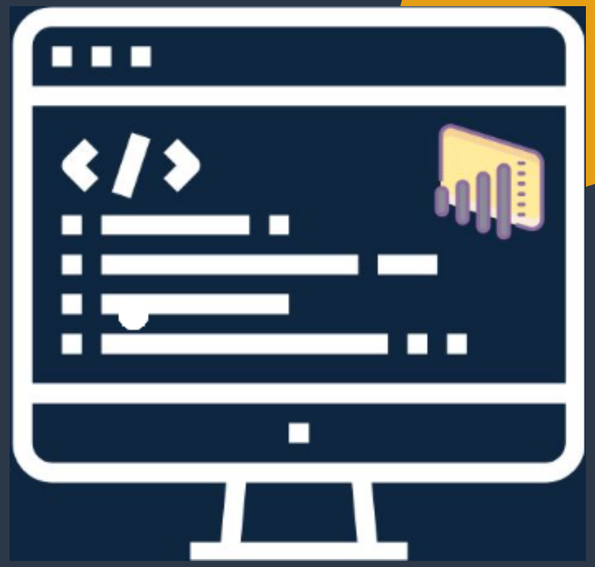
```
Annual Revenue = Stock[Annual Sale Quantity]*Stock[Unit Price]
```



In Stocks Table, Adding New Column

Revenue Share %-> Represents what is % share of each SKU ID wrt all Total Revenue

```
Revenue Share % = (100*Stock[Annual Revenue])/SUM(Stock[Annual Revenue])+0
```

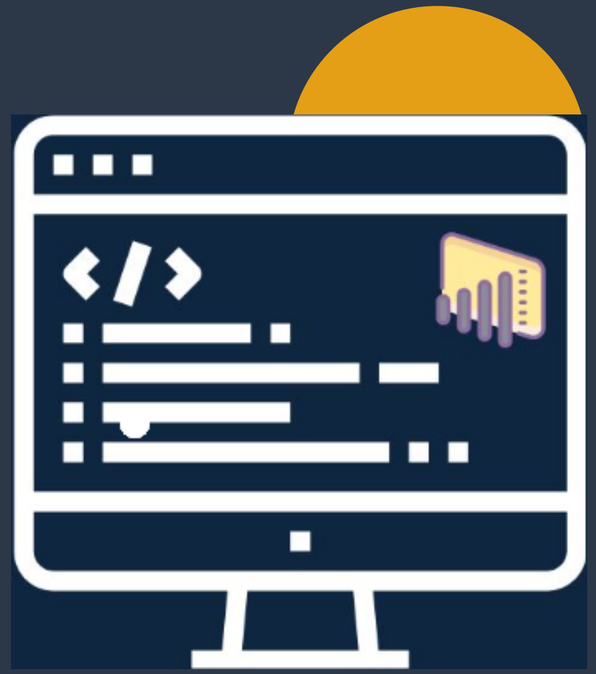


Observed some Blank Cells that means there is NO Order Received form Last 1 Year for Particular SKU ID --> Replaced to Zero

In Stocks Table, Adding New Column

Cummulative Share --> Represents the Cummulative Share

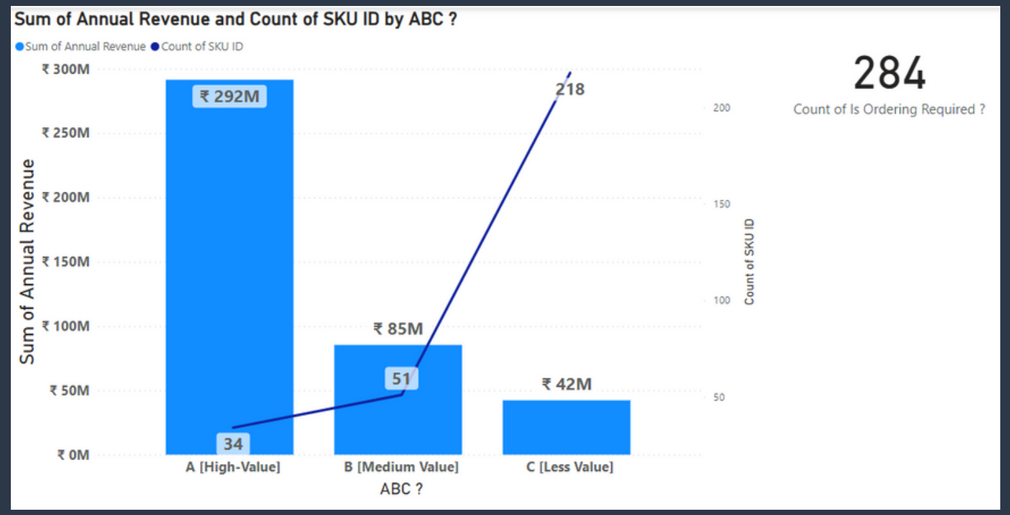
```
Cummulative Share = CALCULATE(
    SUM(Stock[Revenue Share %]),
    FILTER(Stock,
    Stock[Revenue Share %] >= EARLIER(Stock[Revenue Share %])
))
```

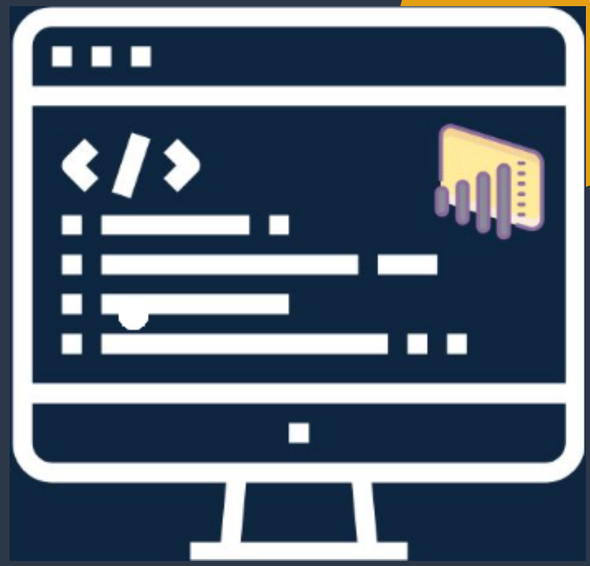


if the SKU ID is 1116CA it will ADD the Revenue share % Till 1116CA SKU ID

In Stocks Table, Adding New Column

ABC Analysis -> follows a principle called **Parito Principle**





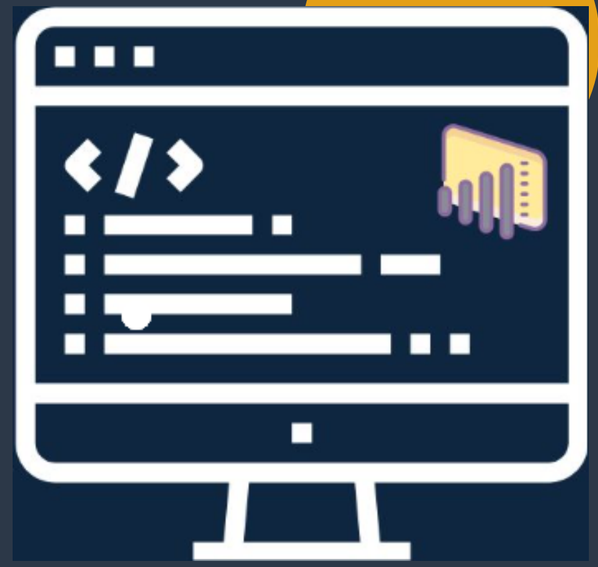
The Less Number of Revenue is generated by More Number of Items.

The More Number of Revenue iis generated by Less Number of Items

Creating **New Table** --> Week Table

Week Date --> Last 1 Year generate all **Weekly Dates**

```
Week Table = GENERATESERIES(DATE(2020,06,14)-365,DATE(2020,06,14),7)
```



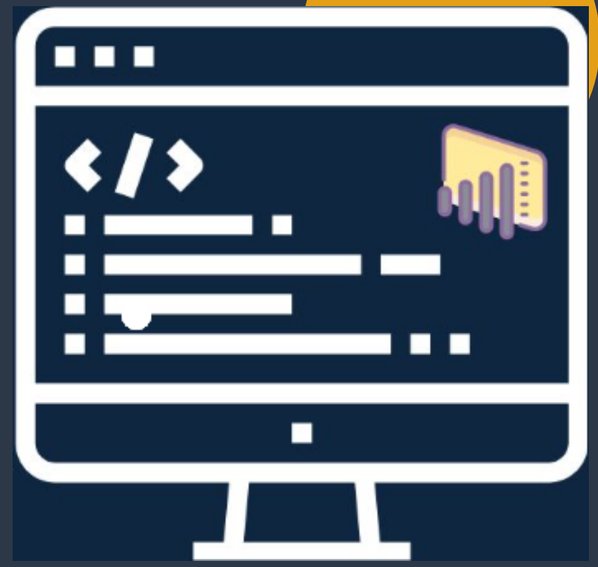
Generate Series --> Function that Generates Dates

7 --> per week we have 7 days

Creating **New Table** --> SKU Table

SKU Table --> From Stocks Table Taking SKU ID

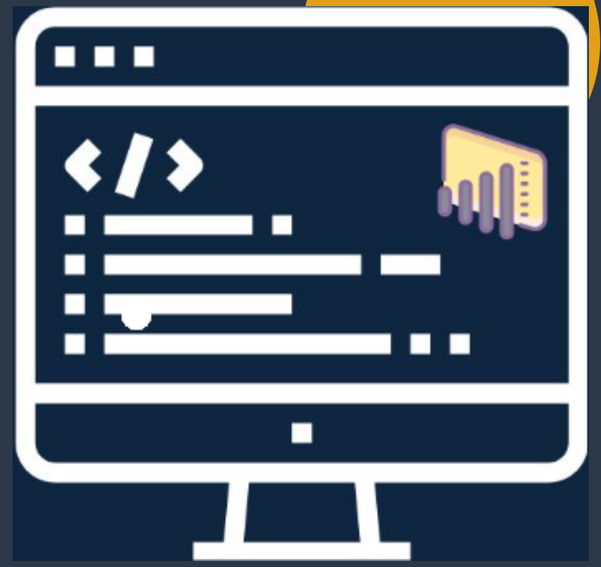
```
SKU Table = SUMMARIZE(Stock,
Stock[SKU ID]
)
```



Creating **New Table** --> Weekly Demand Sheet

Weekly Demand Sheet --> Adding Week Table & SKU ID. For Each Week we have to monitor Every SKU ID and what is the demand.

```
Weekly Demand Sheet = GENERATE('Week Table','SKU Table')
```

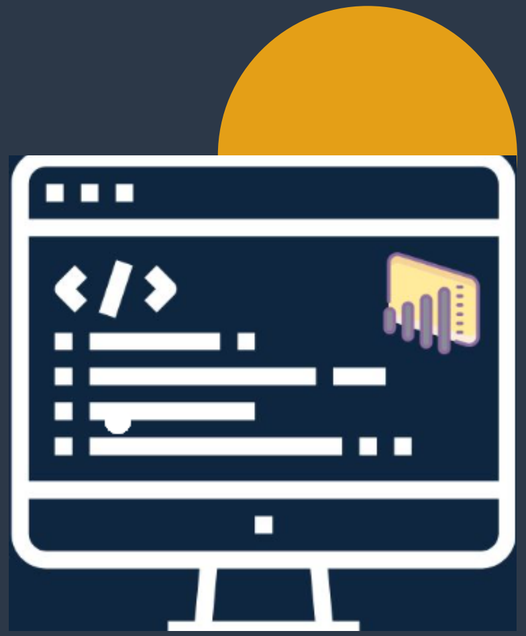


we got for Each WeekDate, what is SKU ID

In Weekely Demand Sheet Table, Adding New Column

Weeks Demand--> For each week what is the demand

```
Weeks Demand = CALCULATE(
    SUM('Past Orders'[Order Quantity]),
    FILTER('Past Orders',
        'Past Orders'[SKU ID]='Weekly Demand Sheet'[SKU ID] &&
        'Past Orders'[Order Date] >= 'Weekly Demand Sheet'[Week Date]-6 &&
        'Past Orders'[Order Date] <= 'Weekly Demand Sheet'[Week Date]
    )
)+0
```



Last 2 Conditions indicates Capturing all orders from that day - 6 days & that particular week Date

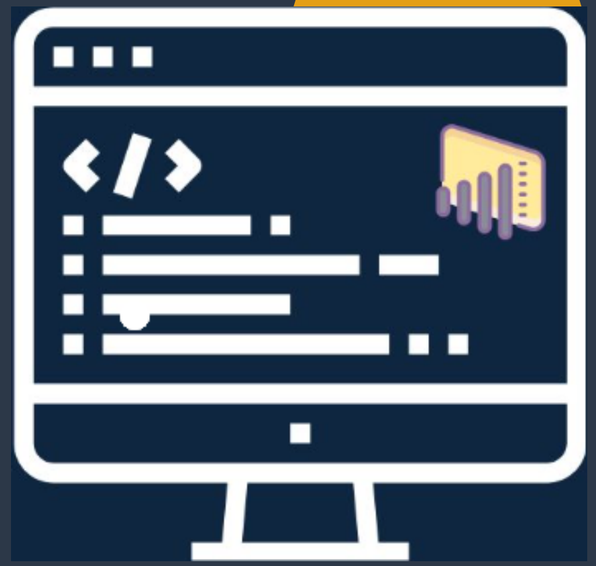
Obeserved some Blanks --> Putting Blanks to Zero because Blank will Not Take Average

Get Weekly Demand For Each Week Date

In Stocks Table, Adding New Column

Average Weekly Demand --> represents the Average of Weekly Demand

```
Average Weekly Demand = CALCULATE(
    AVERAGE('Weekly Demand Sheet'[Weeks Demand]),
    FILTER('Weekly Demand Sheet',
        'Weekly Demand Sheet'[SKU ID]=Stock[SKU ID]
    )
)
```

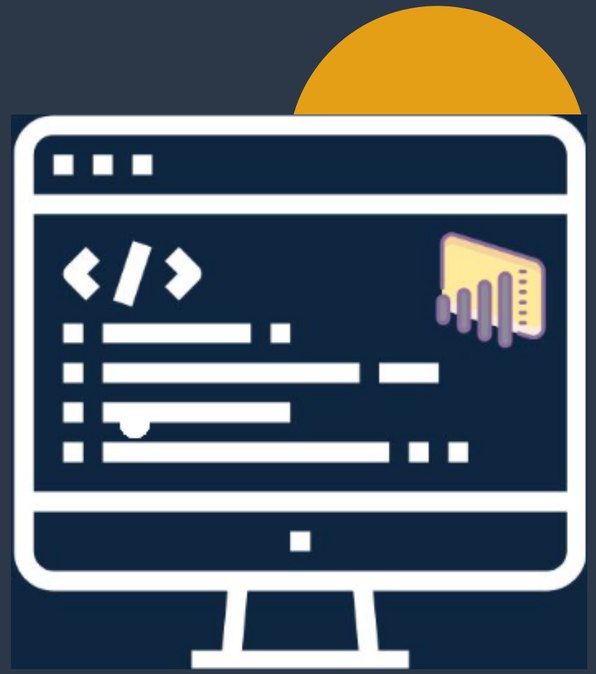


Generated the Average Weekly Demand

In Stocks Table, Adding New Column

STD DEV of Weekly Demand --> represents the Standard Deviation of Weekly Demand

```
Stdev of Weekly Demand = CALCULATE(
    STDEV.P('Weekly Demand Sheet'[Weeks Demand]),
    FILTER('Weekly Demand Sheet',
        'Weekly Demand Sheet'[SKU ID]=Stock[SKU ID]
    )
)
```



Generated the Standard Deviation for Each SKU ID

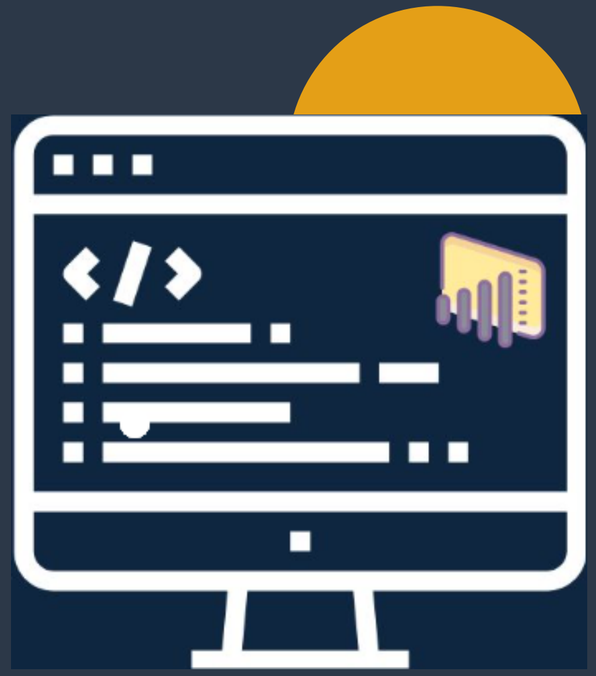
In Stocks Table, Adding New Column

Coefficient of Variation---> Std dev of Weekly Demand by Average Weekly Demand

```
Coeff. Of Variation = IF(Stock[Stdev of Weekly Demand]>0,Stock[Stdev of Weekly Demand]/Stock[Average Weekly Demand],1000)
```

In CV, NAN is there --> Avg Weekly Demand & SD of Weekly Demand is Zero

For Avioding this, Adding Condition.



Generated the CV for Each SKU ID

CV is Classified :- Smaller CV Value --> Under **X Category**

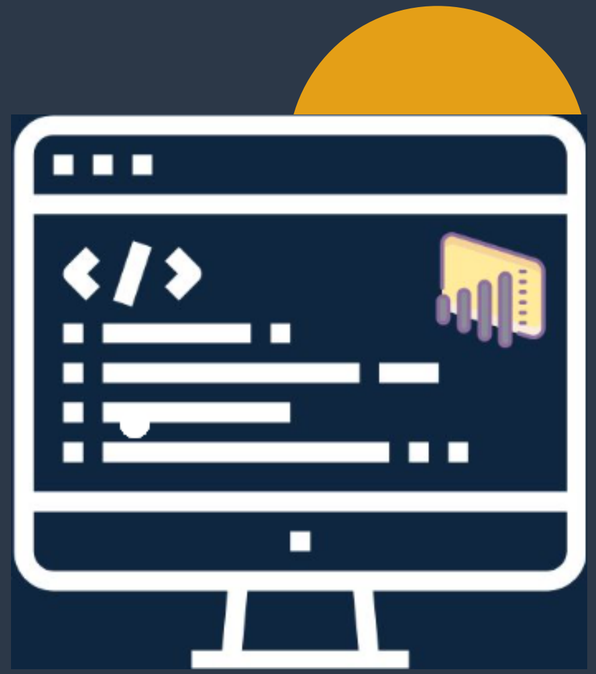
In between Value --> will be **Y Category**

High CV Value -> will be in **Z Category**

In Stocks Table, Adding New Column

CV Rank --> represents the Coefficient of variance Rank

```
Coeff. Var Rank = RANK.EQ(Stock[Coeff. Of Variation],Stock[Coeff. Of Variation],ASC)
```

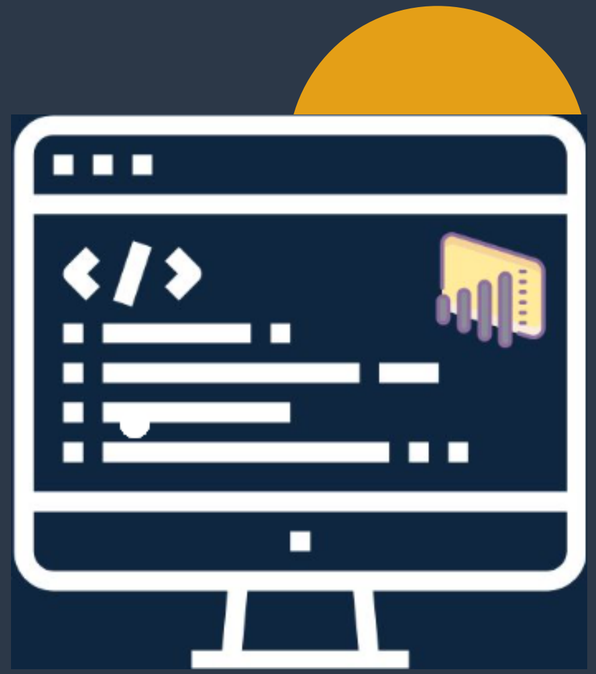


We got Rank wise Based on CV Value in Ascending Order.

In Stocks Table, Adding New Column

XYZ ? --> Categorizing items based on their Demand and Certainity

```
XYZ ? = IF(Stock[Coeff. Var Rank]<=0.2*max(Stock[Coeff. Var Rank]),"X [Uniform Demand]",
    IF(Stock[Coeff. Var Rank]<=0.5*MAX(Stock[Coeff. Var Rank]),"Y [Variable Demand]",
    "Z [Uncertain Demand]"
    )
)
```



The CV Rank <= 20% of MAX CV Rank --> X Uniform Demand

The CV Rank <= 50% of MAX CV Rank --> YVariable Demand,
else Z UncertainDemand

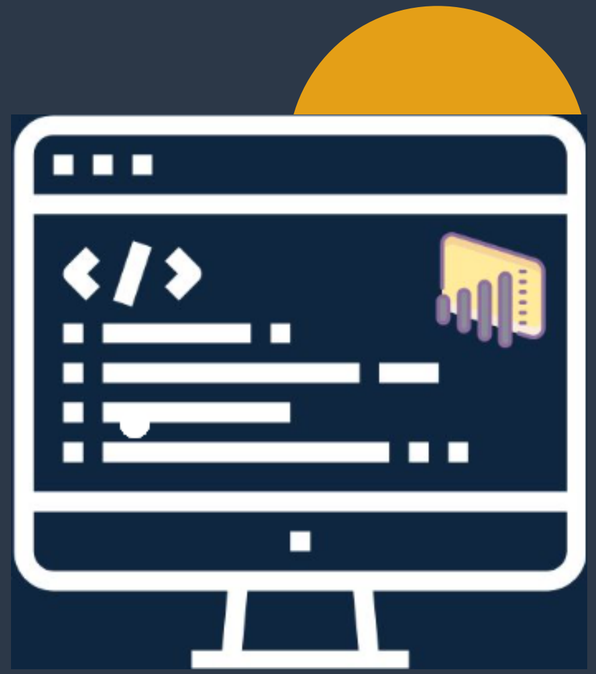
Shows the XYZ Classification for Each SKU ID

We have **Current Stock Quantity** but we want to represent the **Value** as well of the Current Stock Quantity

In Stocks Table, Adding New Column

Value in WH --> Represents the Value in Warehouse of Current Stock Quantity

```
XYZ ? = IF(Stock[Coeff. Var Rank]<=0.2*max(Stock[Coeff. Var Rank]),"X [Uniform Demand]",
    IF(Stock[Coeff. Var Rank]<=0.5*MAX(Stock[Coeff. Var Rank]),"Y [Variable Demand]",
    "Z [Uncertain Demand]"
    )
)
```



The CV Rank <= 20% of MAX CV Rank --> X Uniform Demand

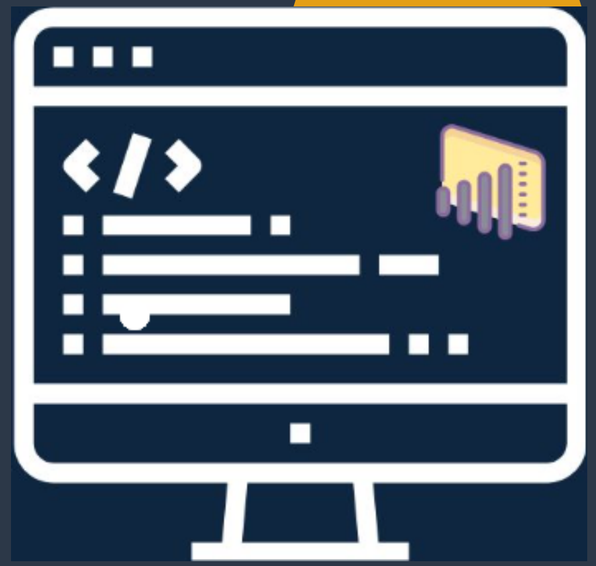
The CV Rank <= 50% of MAX CV Rank --> YVariable Demand,
else Z UncertainDemand

Shows the XYZ Classification for Each SKU ID

In Stocks Table, Adding New Column

ABC Rank --> For Generating ABC Parito Chart

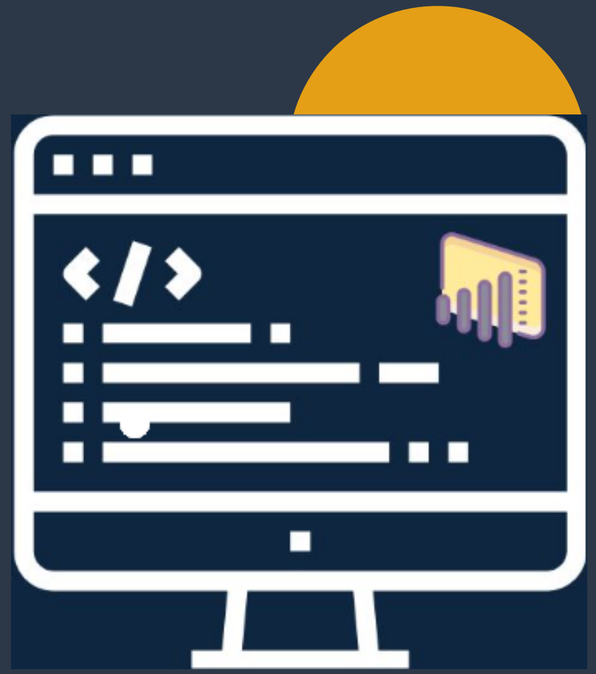
```
ABC Rank = RANK.EQ(Stock[Cummulative Share],Stock[Cummulative Share],ASC)
```



In Stocks Table, Adding New Measure

Inventory Turnover Ratio --> Represents how efficient is Warehouse or Inventory Management is.

```
Coeff. Var Rank = RANK.EQ(Stock[Coeff. Of Variation],Stock[Coeff. Of Variation],ASC)
```

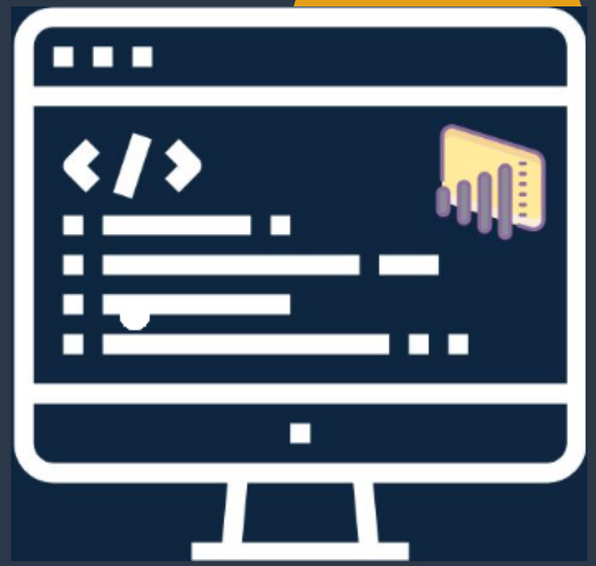


We got Rank wise Based on CV Value in Ascending Order.

In Stocks Table, Adding New Column

Peak Weekly Demand --> represents Peak Weekly Demand for each SKU ID for **Safety Stock**

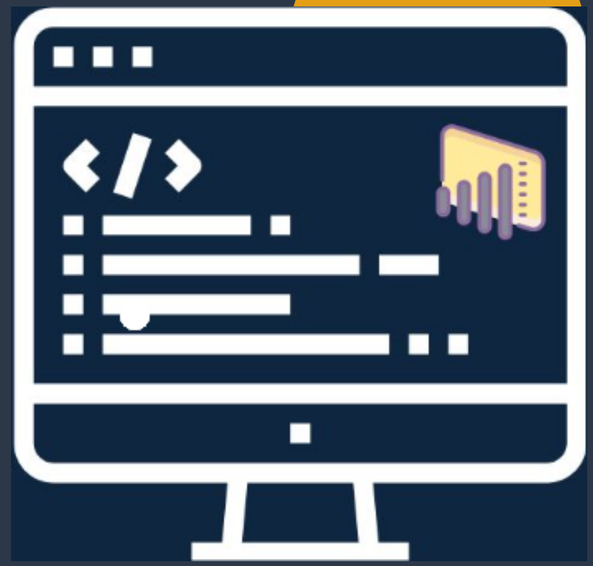
```
Peak Weekly Demand = CALCULATE(
    MAX('Weekly Demand Sheet'[Weeks Demand]),
    FILTER('Weekly Demand Sheet',
    'Weekly Demand Sheet'[SKU ID]=Stock[SKU ID]
    )
)
```



In Stocks Table, Adding New Column

Safety Stocks --> represents Safety Stocks for each SKU ID

```
Safety Stock = ( Stock[Peak Weekly Demand] * Stock[Maximum Lead Time (days)]/7 ) - ( Stock[Average Weekly Demand] * Stock
[Average Lead Time (days)]/7 )
```

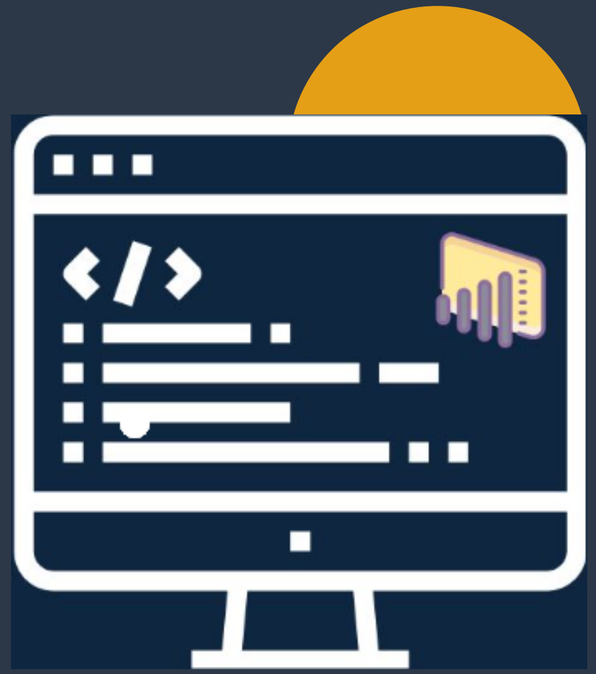


Dividing to 7 --> To Convert into Weeks

In Stocks Table, Adding New Column

Re-order point -->Depending on Safety Stock.

```
Re-Order Point = Stock[Safety Stock] + (Stock[Average Weekly Demand] * Stock[Average Lead Time (days)]/7 )
```

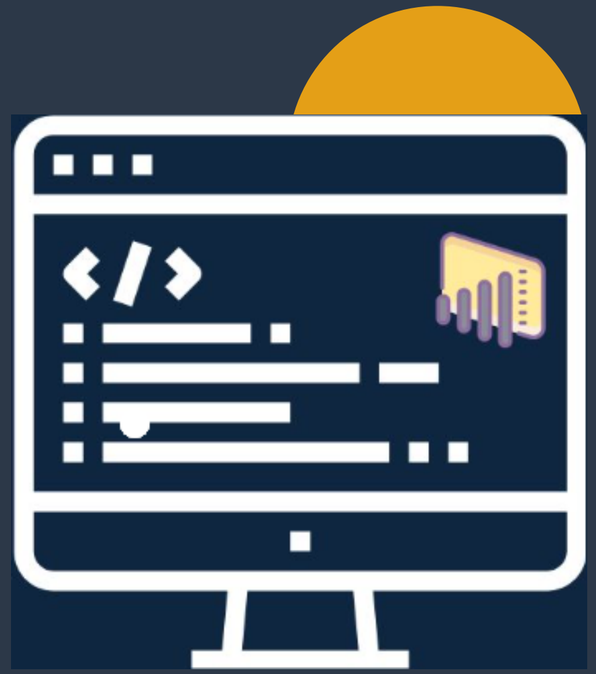


Dividing to 7 --> To Convert into Weeks

In Stocks Table, Adding New Column

Is Ordering Required --> whenever the **Current Stock Value < Re-order** point, that means we need to order

```
Re-Order Point = Stock[Safety Stock] + (Stock[Average Weekly Demand] * Stock[Average Lead Time (days)]/7 )
```

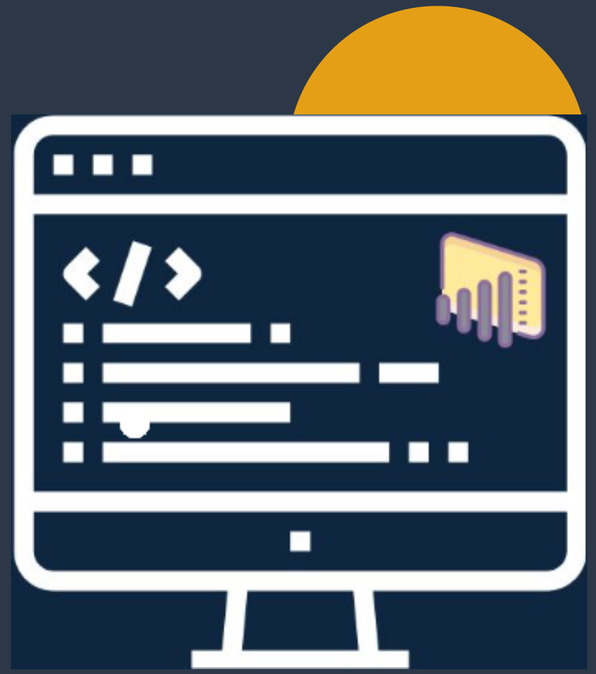


In Our Column, Mostly need to Re-order

In Stocks Table, Adding New Column

Stock Status --> Categorizing **Current Stock Quantity** --> In Stock or Out of Stock

```
Stock Status = IF(Stock[Current Stock Quantity] = 0, "Out of Stock",
    IF(Stock[Current Stock Quantity] < Stock[Safety Stock], "Below Safety Stock",
        IF(Stock[Is Ordering Required ?]="Yes","Below ROP, above Safety Stock", "In Stock"
    )
))
```

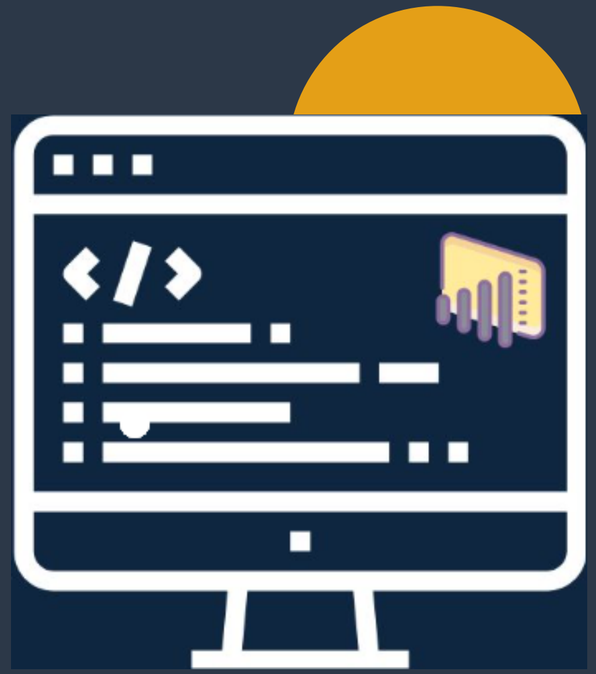


Whenever it is Below Re-order point or Out of Stock, we need to Order

In Stocks Table, Adding New Column

Stock Status --> Categorizing **Current Stock Quantity** --> In Stock or Out of Stock

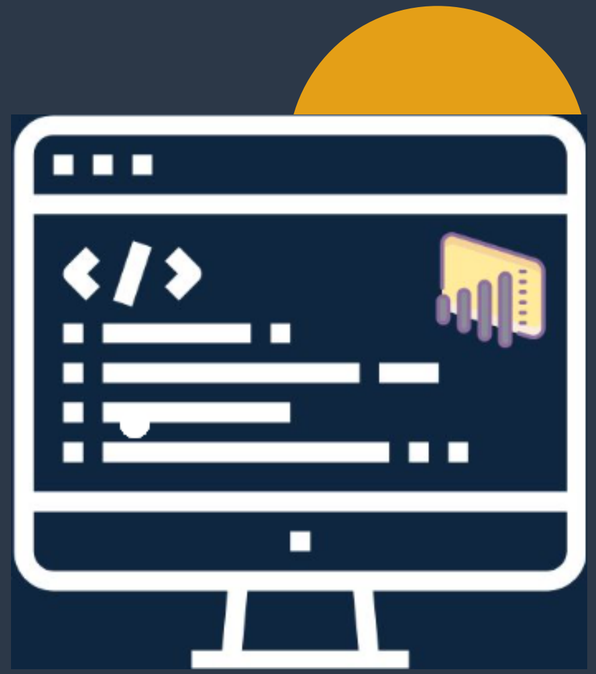
```
Stock Status = IF(Stock[Current Stock Quantity] = 0, "Out of Stock",
    IF(Stock[Current Stock Quantity] < Stock[Safety Stock], "Below Safety Stock",
        IF(Stock[Is Ordering Required ?]="Yes","Below ROP, above Safety Stock", "In Stock"
    )
))
```



Whenever it is Below Re-order point or Out of Stock, we need to Order

Creating **New Measure** for **SKU to Re-order**

```
SKUs to Re-order = CALCULATE(
    COUNT(Stock[SKU ID]),
    FILTER(Stock,
        Stock[Is Ordering Required ?]="Yes"
    )
)
```



Stock as per SKU ID to Reorder

SUMMARY

Using this Dashboard, Inventory Managers have effectively Control in which we are Storing & maintain the Service levels by ordering Right Re-order points.