# **Database and Data Warehouse Design**

## **Artificial Intelligence task 2**

# **Requirements**

## **Business Description**

This is a model of a company producing conveyor equipment for companies across the world. It is based in US focusing exclusively on conveyor manufacturing and engineering. The assembly is done in work centers; each product is being produced in its own area. This has made it necessary to use level loading tools and capacity planning. The data warehouse will be set up to provide data to external sources for security purposes between the productive system and external users. There is need to provide for process improvement. The conveyor sector is very competitive with numerous organizations, and such, firms must evolve very fast and efficiently, including new table and data warehousing tooling

## **Business Problem**

The problem here is the level loading of the newly created work centers. The most pressing problem is that of level loading the newly created work centers. The new work centers must have known available capacities, flexibility to change them on the fly, to schedule work to each work center, and to understand sales projections into the future for work center level loading and business commitments. Although there are hundreds of tables in the operation ERP, this will only view the tables needed for this specific work function.

## **Business Requirements**

### Operational System

* Ability to create new productions and new items
* Accessible through the Microsoft Dynamics Navision client
* Ability to add and modify work centers
* Real-time production claiming
* Ability to reroute productions to new area
* Real-time production scheduling and loading
* Ability to predict production volume based solely on sale value
* Ability to change work center parameters such as capacity

### Data Warehouse

* The only method by which visualizations and external access can access data
* Additional calculations for averages and estimates
* Exclusively used for analytics, non-productive
* As close to real-time data as possible

## **Technical Requirements**

* Two Microsoft SQL servers
* Microsoft Dynamics Navision
* Power BI
* SQL Server Management Studio

## **Data Requirements**

### Operational System

* Production orders must be retained in history and unique
* Ability to change the work center capacities on a granular level
* Work centers must be schedulable but are not used to determine production dates
* Item number may drive routings, but routings must be modifiable
* Ability to modify production dates by a level loading operator

### Data Warehouse

* The data warehouse cannot be accessed directly by users
* All aggregations must be done to standard time scales
* There is no data entry, all data changes are driven by operational

## Business Rules

* Every work center group may contain many work centers
* Booked sales lines that do not contain production order information must be converted to specialty production orders for capacity planning
* The production orders drive the routing lines and may have many for a single production
* Every work center can only be part of one group
* The routing lines for a production order may only have one Assembly routing
* The sales orders drive the creation of production orders and only one can be made per line
* The dates must be based on fiscal year and quarter, with calendar months, weeks, and dates coming after

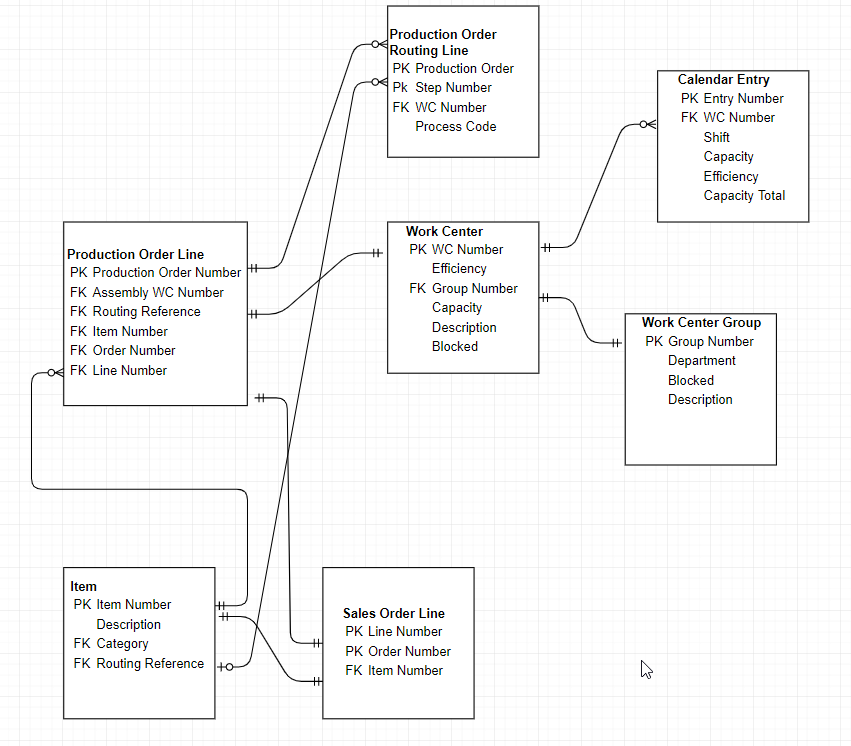
# **Data Models**

## Conceptual Data Model

### **Attribute and Model Descriptions**

* Sales Order Line
  + Contains the customer, shipping address, shipment dates, quantities and pricing data
  + Contains many foreign keys to other normalized tables for things like standard costs and item categories
  + Certain lines are converted to production orders while some items are directly sold to customer after picking operations
* Item
  + All items, including produced and purchased items
  + Unique item number as the primary key
  + Contains a routing reference number which is used for the default routing line creation later
* Production Order Line
  + Contains information on the production type, date, quantities and painting
  + Uses the assembly routing to derive a field for ease of access
  + Production order number is a unique identifier always beginning with the letter “M”
* Production Order Routing Line
  + Many lines for each production, including picking operations, paint operations, inbound warehouse times, and assembly
  + Must only contain one assembly step for each production order
  + Defaults are taken from the item number that was used to generate the production order
  + Contains work center numbers for every process step
* Work Center
  + Each work center that is in the company
  + Ability to block and obsolete work centers on the high level
  + Contains the default worker number, efficiencies, as well as the physical location of the work center
  + Must only be part of a single group (such as assembly, fabrication, shipping, etc.)
* Work Center Group
  + The one half of a one to many relationships with Work Center
  + Contains cost center information for budgeting
  + Used for aggregation level and blocking of large areas of production
* Calendar Entries
  + Single entry for every work center and work period (breaks and closing times determine the start and end periods from the company calendar)
  + Ability to change each entry for new efficiencies, absence, etc.
  + Modifiable in productive system to dynamically change the capacities

## **Logical Data Model**



### **Attribute and Model Descriptions**

* Production Order Line
  + Production Order Number – PK – unique number starting with M then 7 digits
  + Assembly WC Number – FK – the work center where the item will be assembled
  + Routing Reference Number – FK – unique number typically the same as the item number
  + Item Number – FK – used to determine the actual item produced and the type of production
  + Order Number – FK – the sales order number that generated the demand for production
  + Line Number – FK – the sales order line that specifically generated the demand
* Production Order Routing Line
  + Production Order Number – FK and PK – may have several lines for each production order
  + Step Number – PK – the subline of the production order number
  + WC Number – FK – work center relation
  + Process code – information such as PICK, PAINT, or ASSEMBLE
  + Run Time – amount of time the step will take
* Item
  + Item Number – PK – distinct item variants
  + Description – item information
  + Category – FK – an item category code, relation not defined here
  + Routing Reference – FK – the default routings for the item, relation not defined here
* Sales Order Line
  + Order Number – PK – purchase order
  + Line Number – PK – subline of the order number
  + Item Number – FK – the item that is being sold to customer
* Work Center
  + WC Number – PK – identifier of the work center
  + Group Number – FK – relation to the work center group table
  + Description – work center name and info
  + Efficiency – expected production rate
  + Capacity – number of workers
  + Blocked – whether the work center can be used or not
* Work Center Group
  + Group Number – PK – information like ASSY, FAB, etc.
  + Department – FK – department that bears the cost, relation not defined here
  + Description – detailed info
  + Blocked – whether to block all associated work centers in this group – pushed to work center table through validation in application
* Calendar Entry
  + Entry Number – PK – unique entry number generated when the calendar plan is recalculated for a date range
  + Date – no defined relation in productive, will have relation in data warehouse
  + WC Number – FK – associated work center
  + Shift – shift number
  + Capacity – entered when the line is generated, default is from the work center
  + Efficiency - entered when the line is generated, default is from the work center
  + Capacity total – Capacity \* Efficiency/100

# **Physical Database Design**

## Normalized Forms

All tables are normalized by default due to design of the original database. Steps undertaken by original designers to ensure that each transaction and fact table was normalized has thusly enabled users to focus on the analytics and design rather than the ERD.

## Sample Tables

* Production Order Line

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Production Order Number | Ending Date (Due Date) | Assembly WC Number | Routing Reference Number | Item Number | Order Number | Line Number |
| M0000001 | 05/31/2019 | ASY-ITR-130 | E001 | E001 | 1111111 | 1 |
| M0000002 | 05/31/2019 | ASY-SPECIALS | E002 | E002 | 1111112 | 1 |
| M0000003 | 05/31/2019 | ASY-ITR-130 | E001 | E001 | 1111113 | 1 |
| M0000004 | 05/31/2019 | ASY-NBC-165 | E003 | E003 | 1111112 | 2 |
| M0000005 | 05/30/2019 | ASY-CZB-150 | E010 | E004 | 1111114 | 1 |

* Production Order Routing Line

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Production Order Number | Step Number | WC Number | Process Code | Run Time |
| M0000001 | 1 | PICK | PICK | 1 |
| M0000001 | 2 | ASY-ITR-130 | ASSY | 1 |
| M0000002 | 1 | PICK | PICK | 1 |
| M0000002 | 2 | ASY-SPECIALS | ASSY | 5 |
| M0000005 | 1 | PICK | PICK | 1 |

* Item

|  |  |  |  |
| --- | --- | --- | --- |
| Item Number | Description | Category | Routing Reference Number |
| E001 | ITR bed | PRODUCT | E001 |
| E002 | Customer bed | PRODUCT | E002 |
| E003 | NBC bed | PRODUCT | E003 |
| E004 | Roller | PURCHASE ITEM | E010 |
| E005 | O-ring | PURCHASE ITEM | E005 |

* Sales Order Line

|  |  |  |
| --- | --- | --- |
| Order Number | Line Number | Item Number |
| 1111111 | 1 | E001 |
| 1111112 | 1 | E002 |
| 1111112 | 2 | E003 |
| 1111113 | 1 | E001 |
| 1111114 | 1 | E004 |

* Work Center

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| WC Number | Group Number | Description | Efficiency | Capacity | Blocked |
| ASY-ITR-130 | ASSY | ITR line | 100 | 10 | 0 |
| ASY-SPECIALS | ASSY | Specials area | 90 | 5 | 0 |
| PICK | STOCK | Picking | 90 | 10 | 0 |
| ASY-NBC-165 | ASSY | NBC line | 100 | 5 | 0 |
| ASY-CZB-150 | ASSY | CZB line | 100 | 5 | 0 |
| WELD | FAB | Welding station | 70 | 2 | 0 |
| BENCH1 | ASSY | Bench number 1 | 80 | 1 | 1 |

* Work Center Group

|  |  |  |  |
| --- | --- | --- | --- |
| Group Number | Department | Description | Blocked |
| ASSY | D001 | Assembly group | 0 |
| STOCK | D002 | Logistics group | 0 |
| FAB | D001 | Assembly group | 0 |
| MAINT | D003 | Maintenance group | 1 |

* Calendar Entry

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Entry Number | Date | WC Number | Shift | Capacity | Efficiency | Capacity Total |
| 00001 | 05/31/2019 | ASY-ITR-130 | 1 | 10 | 100 | 10 |
| 00002 | 05/31/2019 | ASY-ITR-130 | 1 | 10 | 100 | 10 |
| 00003 | 05/31/2019 | ASY-ITR-130 | 2 | 10 | 100 | 10 |
| 00004 | 05/31/2019 | ASY-SPECIALS | 1 | 90 | 10 | 9 |
| 00005 | 05/31/2019 | ASY-NBC-165 | 1 | 100 | 5 | 5 |

## **Table Creation Code**

CREATE TABLE [Production Order Line] (

[Production Order Number] VARCHAR(255) NOT NULL,

[Ending Date] DATETIME NOT NULL,

[Assembly WC Number] VARCHAR(255) NOT NULL,

[Routing Reference Number] VARCHAR(255) NOT NULL,

[Item Number] VARCHAR(255) NOT NULL,

[Order Number] VARCHAR(255) NOT NULL,

[Line Number] INT NOT NULL,

);

CREATE TABLE [Production Order Routing Line] (

[Production Order Number] VARCHAR(255) NOT NULL,

[Step Number] INT NOT NULL,

[WC Number] VARCHAR(255) NOT NULL,

[Process Code] VARCHAR(255) NOT NULL,

[Run Time] INT NOT NULL,

);

CREATE TABLE [Item] (

[Item Number] VARCHAR(255) NOT NULL,

[Description] VARCHAR(255) NOT NULL,

[Category] VARCHAR(255) NOT NULL,

[Routing Reference Number] VARCHAR(255) NOT NULL,

);

CREATE TABLE [Sales Order Line] (

[Order Number] VARCHAR(255) NOT NULL,

[Line Number] VARCHAR(255) NOT NULL,

[Item Number] VARCHAR(255) NOT NULL,

);

CREATE TABLE [Work Center] (

[WC Number] VARCHAR(255) NOT NULL,

[Group Number] VARCHAR(255) NOT NULL,

[Description] VARCHAR(255) NOT NULL,

[Efficiency] INT NOT NULL,

[Capacity] INT NOT NULL,

[Blocked] BIT NOT NULL,

);

CREATE TABLE [Work Center Group] (

[Group Number] VARCHAR(255) NOT NULL,

[Department] VARCHAR(255) NOT NULL,

[Description] VARCHAR(255) NOT NULL,

[Blocked] BIT NOT NULL,

);

CREATE TABLE [Calendar Entry] (

[Entry Number] INT NOT NULL,

[Date] DATETIME NOT NULL,

[WC Number] VARCHAR(255) NOT NULL,

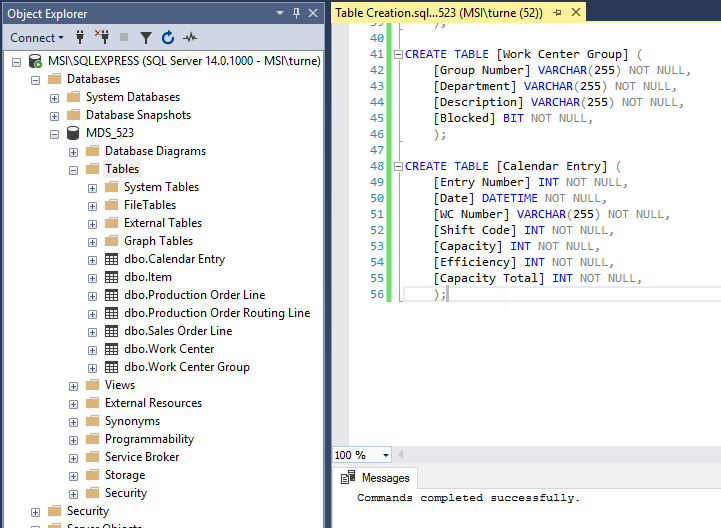
[Shift Code] INT NOT NULL,

[Capacity] INT NOT NULL,

[Efficiency] INT NOT NULL,

[Capacity Total] INT NOT NULL,

);



## **Table Insertion Code**

INSERT INTO [Production Order Line]

VALUES

('M0000001','05/31/2019','ASY-ITR-130','E001','E001','1111111','1'),

('M0000002','05/31/2019','ASY-SPECIALS','E002','E002','1111112','1'),

('M0000003','05/31/2019','ASY-ITR-130','E001','E001','1111113','1'),

('M0000004','05/31/2019','ASY-NBC-165','E003','E003','1111112','2'),

('M0000005','05/30/2019','ASY-CZB-150','E010','E004','1111114','1')

;

INSERT INTO [Production Order Routing Line]

VALUES

('M0000001','1','PICK','PICK','1'),

('M0000001','2','ASY-ITR-130','ASSY','1'),

('M0000002','1','PICK','PICK','1'),

('M0000002','2','ASY-SPECIALS','ASSY','5'),

('M0000005','1','PICK','PICK','1')

;

INSERT INTO [Item]

VALUES

('E001','ITR bed','PRODUCT','E001'),

('E002','Customer bed','PRODUCT','E002'),

('E003','NBC bed','PRODUCT','E003'),

('E004','Roller','PURCHASE ITEM','E010'),

('E005','O-ring','PURCHASE ITEM','E005')

;

INSERT INTO [Sales Order Line]

VALUES

('1111111','1','E001'),

('1111112','1','E002'),

('1111112','2','E003'),

('1111113','1','E001'),

('1111114','1','E004')

;

INSERT INTO [Work Center]

VALUES

('ASY-ITR-130','ASSY','ITR line','100','10','0'),

('ASY-SPECIALS','ASSY','Specials area','90','5','0'),

('PICK','STOCK','Picking','90','10','0'),

('ASY-NBC-165','ASSY','NBC line','100','5','0'),

('ASY-CZB-150','ASSY','CZB line','100','5','0'),

('WELD','FAB','Welding station','70','2','0'),

('BENCH1','ASSY','Bench number 1','80','1','1')

;

INSERT INTO [Calendar Entry]

VALUES

('0000','05/31/2019','ASY-ITR-130','1','10','100','10'),

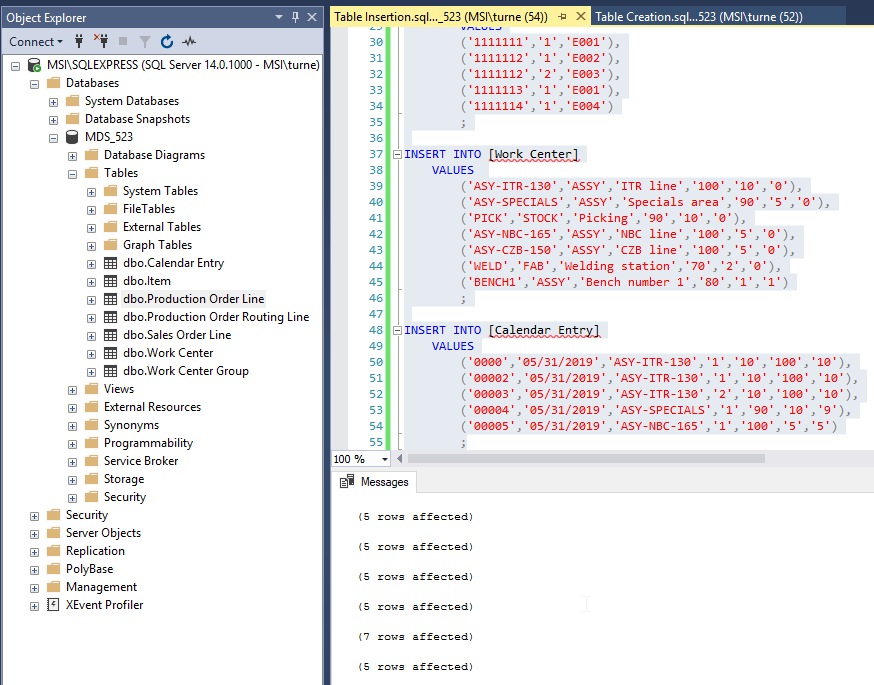
('00002','05/31/2019','ASY-ITR-130','1','10','100','10'),

('00003','05/31/2019','ASY-ITR-130','2','10','100','10'),

('00004','05/31/2019','ASY-SPECIALS','1','90','10','9'),

('00005','05/31/2019','ASY-NBC-165','1','100','5','5')

;

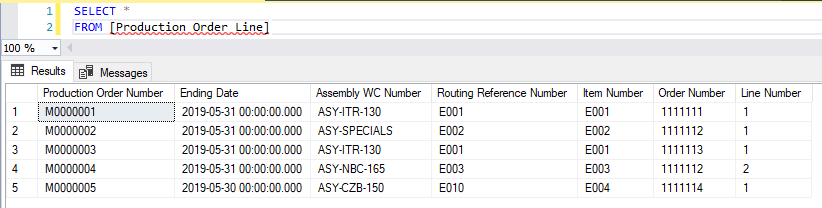


# **Queries**

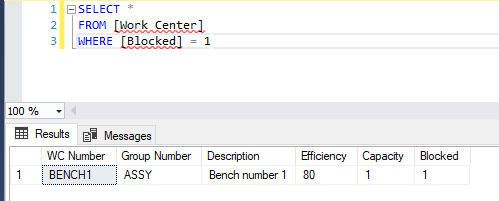
The queries shown below only return a small number of entries, while the productive system may return many thousands or millions. An example of the report generated form the real data can be seen below as an example of the more complex and large scale dataset.

## **Sample Queries**

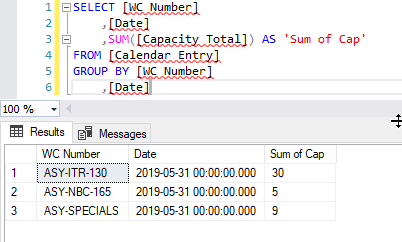
* All production order lines:



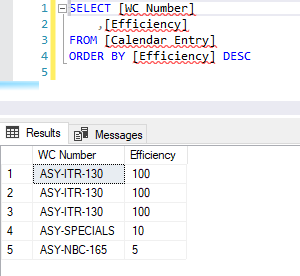
* All blocked work centers:



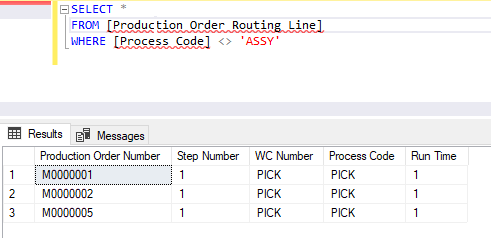
* Capacity by work center and date:



* Work centers order by efficiency:



* Routing lines that are not assembly:



# Data Warehouse Design and Coding

## Data Warehouse Design

In this existing business, the data warehouse is being created using batch updates at set intervals of once hourly, where the data from the productive database above is being loaded into the data warehouse using merge statements for each desired table. The existing infrastructure is reliable enough for this to work for the majority of users, however, as we begin to expand it is becoming increasingly more useful to include more tables than previously defined in the refresh and more real-time calculations.

In addition to the operational database, there is CRM and HRM services that run outside of out ERP, as well as a newly designed warehouse management system that runs separately from the operational database and uses shared views to communicate with each other.

The modeling of the data warehouse has perfectly matched the productive in its base form. This includes all the related fields, schema changes, and informational changes.

We are also batching certain views and tables for quicker access. For example, a modified production order view that includes all the hours that have been claimed by employees like welding and assembly time and the count of production order components that are still short

## Data Marts

All data in databases is currently controlled by IT across the international units. For security and compliance concerns, all data is accessed either through the ERP, CRM, or HRM systems with their own validation and certain reporting capabilities, or through data warehouse image queries. This does make the lead time for ad hoc data and reporting to be a bit longer, but it ensures that all data is kept secure and standardized.

## Data Warehouse Queries

Below are examples of report and the query that is being used to create a large portion of the data, and a few sample queries for different users

### Level Loading Example

SELECT pol.[Prod\_ Order No\_]

,CASE pol.[Status]

WHEN 0 THEN 'Simulated'

WHEN 1 THEN 'Planned'

WHEN 2 THEN 'Firm Planned'

WHEN 3 THEN 'Released'

WHEN 4 THEN 'Finished'

END AS 'Status'

,pol.[Line No\_]

,pol.[Item No\_]

,pol.[Sales Order No\_]

,pol.[Segment]

,pol.[Sales Order No\_] + ' - ' + pol.[Segment] AS 'Sales Order Key'

,pol.[Variant Code]

,pol.[Description]

,pol.[Shortcut Dimension 2 Code]

,pol.[Remaining Quantity]

,pol.[Finished Quantity]

,pol.[Ending Date Modifier]

,pol.[Modified Ending Date]

,pol.[Accuracy]

,pol.[Pick Operation Header No\_] AS 'PH-A'

,(SELECT CASE poh.[Status]

WHEN 0 THEN 'Open'

WHEN 1 THEN 'Rel'

WHEN 2 THEN 'IP'

WHEN 3 THEN 'PC'

WHEN 4 THEN 'Comp'

END AS 'Status'

FROM [TGW-ERMANCO$Pick Operation Header] AS poh

WHERE poh.[Pick Operation Header No\_] = pol.[Pick Operation Header No\_]

) AS 'PH-A Status'

,pol.[Pick Operation Hdr\_ No\_ B-Side] AS 'PH-B'

,(SELECT CASE poh.[Status]

WHEN 0 THEN 'Open'

WHEN 1 THEN 'Rel'

WHEN 2 THEN 'IP'

WHEN 3 THEN 'PC'

WHEN 4 THEN 'Comp'

END AS 'Status'

FROM [TGW-ERMANCO$Pick Operation Header] AS poh

WHERE poh.[Pick Operation Header No\_] = pol.[Pick Operation Hdr\_ No\_ B-Side]

) AS 'PH-B Status'

,porl.[Work Center No\_]

,porl.[Run Time] AS 'Run Time/ Standard Hours Per Production'

,porl.[Run Time] \* pol.[Quantity] AS 'Total Planned Run Time/ Standard Hours'

,porl.[Run Time] \* pol.[Finished Quantity] AS 'Total Produced Run Time/ Standard Hours'

,cle2.[Sum of Reported Hours]

,pol.[Remaining Quantity] \* porl.[Run Time] AS 'Remaining Run Time'

FROM [TGW-ERMANCO$Prod\_ Order Line] AS pol

LEFT JOIN [TGW-ERMANCO$Prod\_ Order Routing Line] AS porl

ON pol.[Prod\_ Order No\_] = porl.[Prod\_ Order No\_]

LEFT JOIN (SELECT cle.[Document No\_]

,cle.[Work Center No\_]

,SUM(cle.[Run Time]) AS 'Sum of Reported Hours'

FROM [TGW-ERMANCO$Capacity Ledger Entry] AS cle

GROUP BY cle.[Document No\_]

,cle.[Work Center No\_]

) AS cle2

ON cle2.[Document No\_] = pol.[Prod\_ Order No\_]

AND cle2.[Work Center No\_] = porl.[Work Center No\_]

WHERE porl.[Process Structure Code] = 'ASS'

AND pol.[Due Date] BETWEEN DATEADD(month, -1, GETDATE()) AND DATEADD(year, 1, GETDATE())

AND pol.[Item Category Code] = 'PRODUCT'

UNION ALL

SELECT CONVERT(VARCHAR(30), 'M-' + sl.[Document No\_] + '-' + sl.[Segment] + '-' + CONVERT(VARCHAR(30),sl.[Line No\_])) AS 'Prod\_ Order No\_'

,CONVERT(VARCHAR(12), 'Quote') AS 'Status'

,CONVERT(INT, sl.[Line No\_]) AS 'Line No\_'

,CONVERT(VARCHAR(20), sl.[No\_]) AS 'Item No\_'

,CONVERT(VARCHAR(20), sl.[Document No\_]) AS 'Sales Order No\_'

,CONVERT(VARCHAR(30), sl.[Segment]) AS 'Segment'

,CONVERT(VARCHAR(53), sl.[Document No\_] + ' - ' + sl.[Segment]) AS 'Sales Order Key'

,CONVERT(VARCHAR(10), sl.[Variant Code]) AS 'Variant Code'

,CONVERT(VARCHAR(50), sl.[Description] + ' for unit price of ' + CONVERT(VARCHAR, CONVERT(DECIMAL(10,2),sl.[Unit Price]))) AS 'Description'

,CONVERT(VARCHAR(20), sl.[Shortcut Dimension 2 Code]) AS 'Shortcut Dimension 2 Code'

,CONVERT(DECIMAL(38,20), sl.[Outstanding Quantity]) AS 'Remaining Quantity'

,CONVERT(DECIMAL(38,20), 0) AS 'Finished Quantity'

,CONVERT(INT, 0) AS 'Ending Date Modifier'

,CONVERT(DATETIME, sl.[Shipment Date]) AS 'Modified Ending Date'

,CONVERT(INT, sl.[Accuracy]) AS 'Accuracy'

,CONVERT(VARCHAR(20), 'PHBOOKORDER') AS 'PH-A'

,CONVERT(VARCHAR(4), 'Open') AS 'PH-A Status'

,CONVERT(VARCHAR(20), 'PHBOOKORDER') AS 'PH-B'

,CONVERT(VARCHAR(4), 'Open') AS 'PH-B Status'

,CONVERT(VARCHAR(20), sl.[Description 2]) AS 'Work Center No\_'

,CONVERT(DECIMAL(38,20), 0) AS 'Run Time/ Standard Hours Per Production'

,CONVERT(DECIMAL(38,6), sl.[Unit Price] /

wcm.[Mean Unit Price per Run Time]

) AS 'Total Planned Run Time/ Standard Hours' --Expected run time based on average by work center

,CONVERT(DECIMAL(38,6), 0) AS 'Total Produced Run Time/ Standard Hours'

,CONVERT(DECIMAL(38,20), 0) AS 'Sum of Reported Hours'

,CONVERT(DECIMAL(38,6), sl.[Unit Price] /

wcm.[Mean Unit Price per Run Time]

) AS 'Remaining Run Time'

FROM [TGW-ERMANCO$Sales Line] AS sl

LEFT JOIN [US\_DataWarehouse].[dbo].[Work Center Means] AS wcm

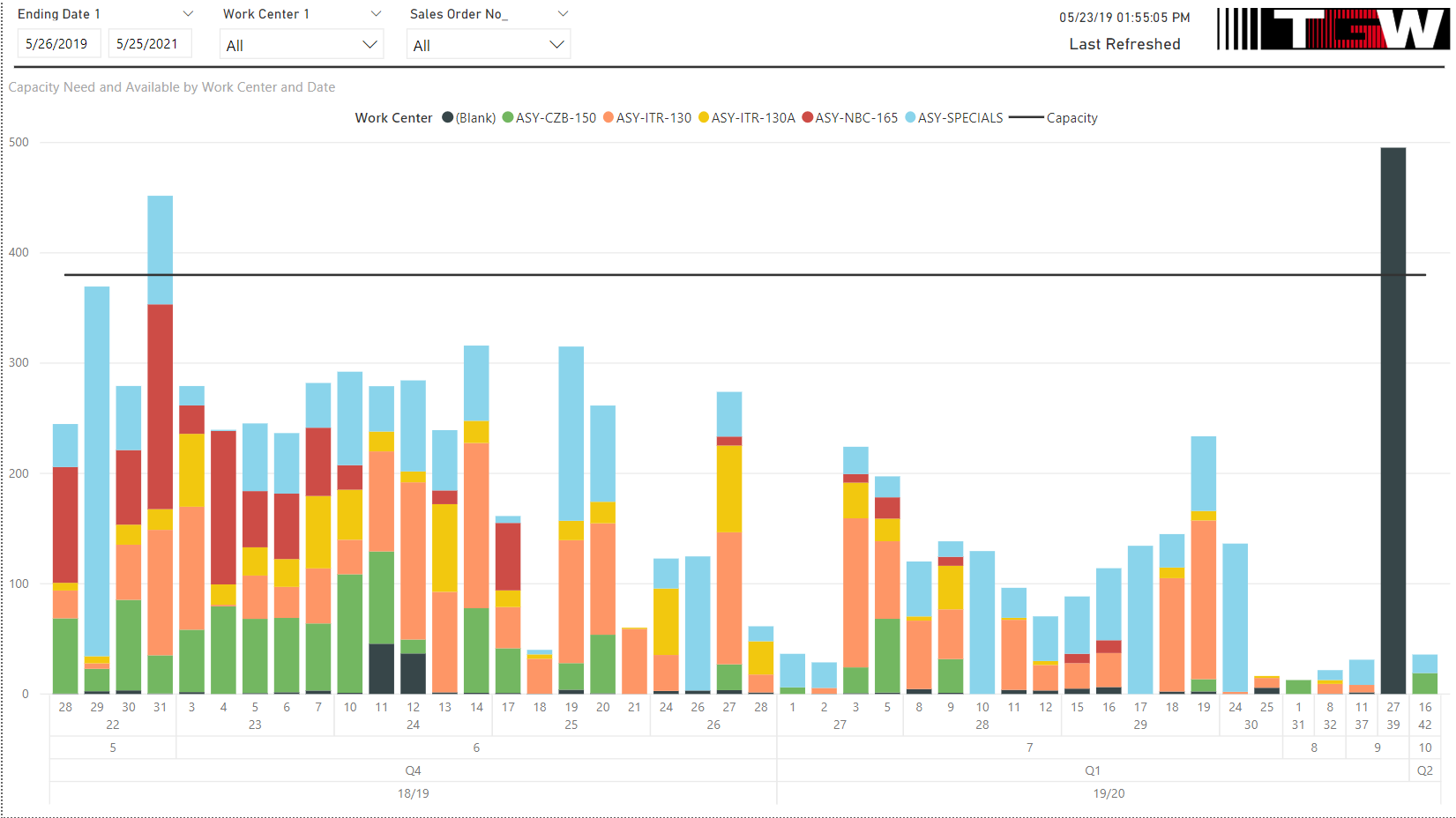
ON sl.[Description 2] = wcm.[No\_]

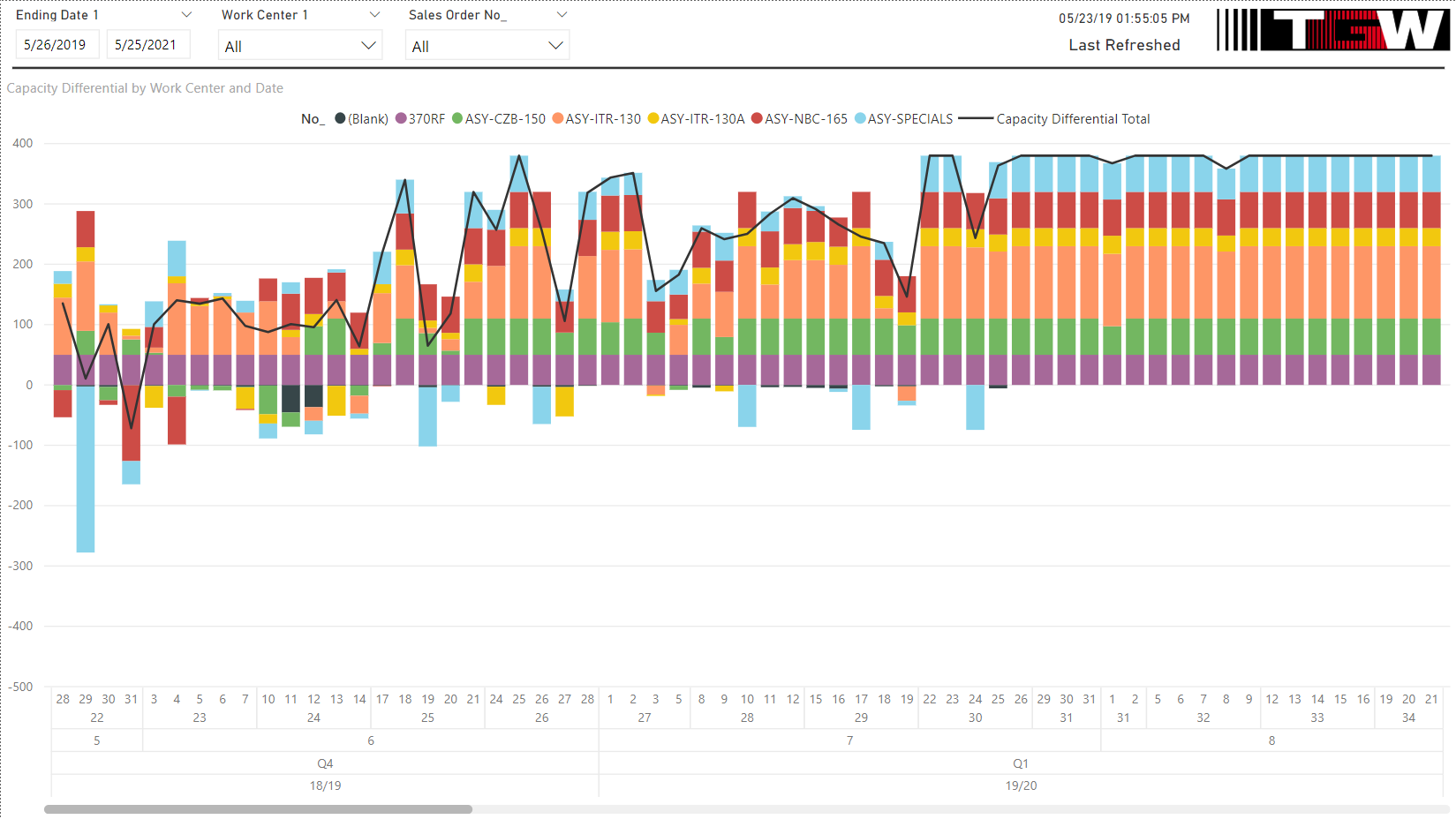
WHERE sl.[No\_] = 'BOOKORDER'

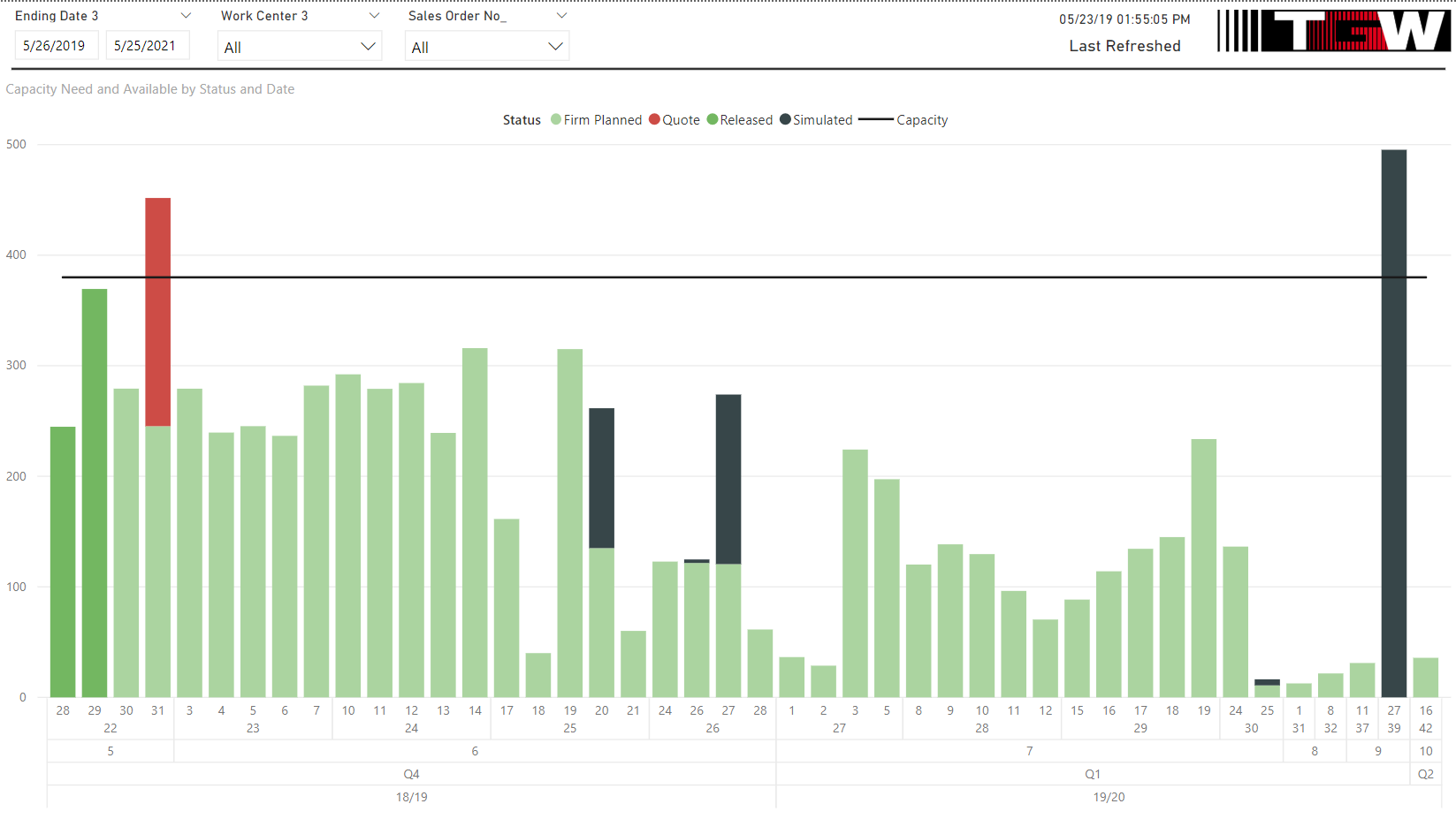
AND sl.[Document No\_] NOT LIKE '%Q%'

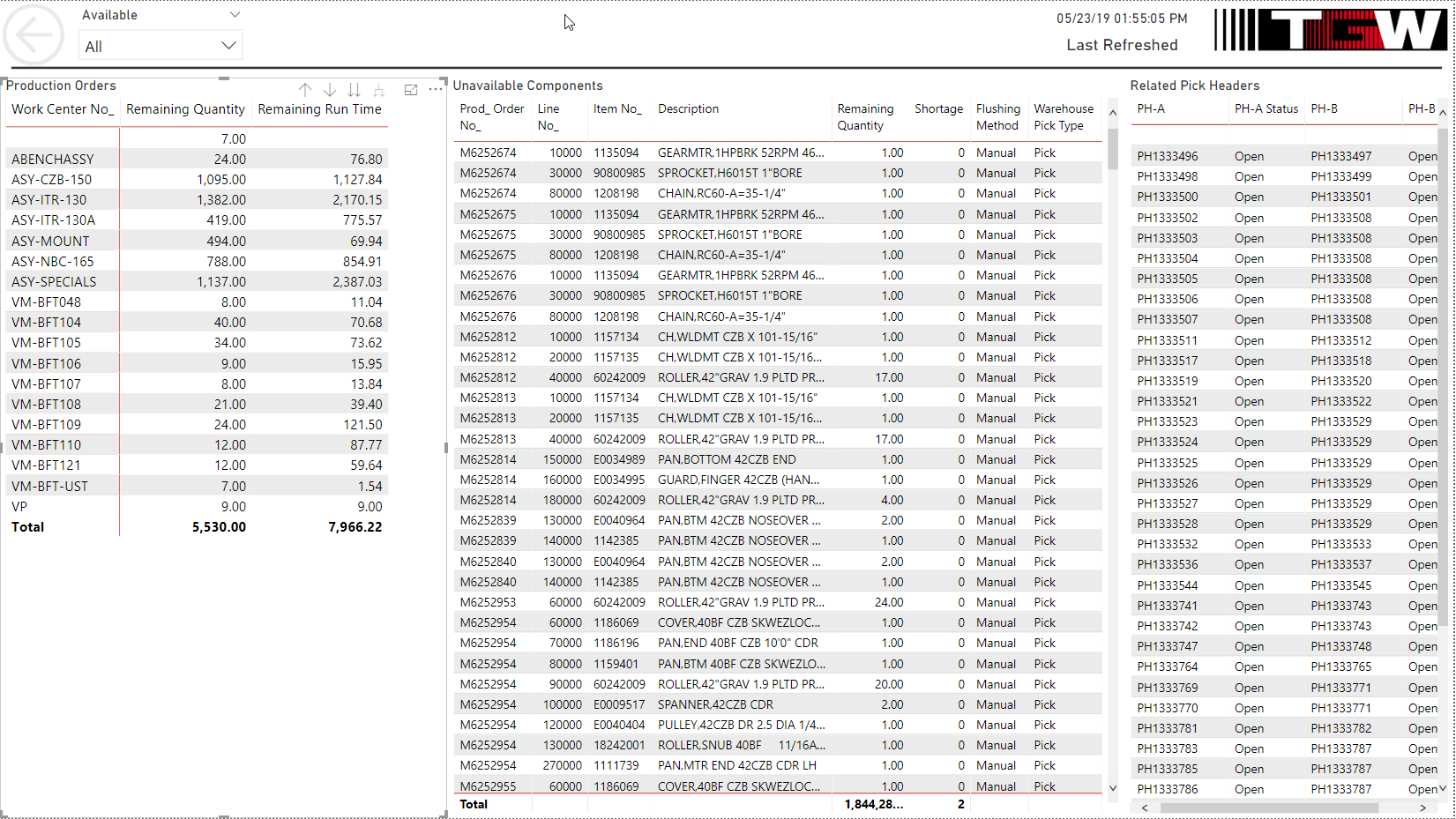
AND sl.[Shipment Date] > DATEADD(month, -1, GETDATE())

The above is the basis for the following user interactive visuals:









Additionally, data warehouse has been used to create standardized tables like the following date table and the subsequent production order line view that is used for several reports used in daily production

### Date Table Example

--Datebase

USE US\_DataWarehouse;

--Variable declaration

DECLARE @Start\_date DATE = '01/01/2021', @End\_date DATE = '12/31/2050';

SET DATEFIRST 1;

SET DATEFORMAT mdy;

--Create temporary table to give values to permanent table

CREATE TABLE #temp

([Date] DATE PRIMARY KEY

,[Day] AS DATEPART(DAY, [date])

,[Month] AS DATEPART(MONTH, [date])

,[First of Month] AS CONVERT(DATE, DATEADD(MONTH, DATEDIFF(MONTH, 0, [date]), 0))

,[Month Name] AS DATENAME(MONTH, [date])

,[Week] AS DATEPART(WEEK, [date])

,[ISOweek] AS DATEPART(ISO\_WEEK, [date])

,[Day of Week] AS DATEPART(WEEKDAY, [date])

,[Quarter] AS DATEPART(QUARTER, [date])

,[Year] AS DATEPART(YEAR, [date])

,[First of Year] AS CONVERT(DATE, DATEADD(YEAR, DATEDIFF(YEAR, 0, [date]), 0))

,[Style112] AS CONVERT(CHAR(8), [date], 112)

,[Style101] AS CONVERT(CHAR(10), [date], 101)

);

--Generate and insert values into temporary table

INSERT #temp([Date])

SELECT vals

FROM (SELECT vals = DATEADD(DAY,row\_num - 1, @Start\_date)

FROM (SELECT TOP (DATEDIFF(DAY, @Start\_date, @End\_date))

row\_num = ROW\_NUMBER() OVER (ORDER BY sao.[object\_id])

FROM sys.all\_objects AS sao

CROSS JOIN sys.all\_objects AS sao2

ORDER BY sao.[object\_id]

) AS x

) AS y;

--Create permanent table

/\*

CREATE TABLE [dbo].[TGW-ERMANCO$Date]

(

[Date] DATE NOT NULL PRIMARY KEY

,[Day] TINYINT NOT NULL

,[Month] TINYINT NOT NULL

,[Year] INT NOT NULL

,[Week] TINYINT NOT NULL

,[Calendar Quarter] TINYINT NOT NULL

,[Day of Week Number] INT NOT NULL

,[Day of Week Name] VARCHAR(3) NOT NULL

,[Week in Month] INT NOT NULL

,[Month Name] VARCHAR(10) NOT NULL

,[Fiscal Year] VARCHAR(5) NOT NULL

,[Fiscal Quarter Number] TINYINT NOT NULL

,[Fiscal Quarter Name] VARCHAR(2) NOT NULL

,[Weekend] BIT NOT NULL

,[First of Month] DATE NOT NULL

,[Last of Month] DATE NOT NULL

);

\*/

--Load temporary values into table along with calculated values

INSERT INTO [dbo].[TGW-ERMANCO$Date]

SELECT [Date] = [Date]

,[Day] = [Day]

,[Month] = [Month]

,[Year] = [Year]

,[Week] = [Week]

,[Calendar Quarter] = [Quarter]

,[Day of Week Number] = [Day of Week]

,[Day of Week Name] = CONVERT(VARCHAR(3), DATENAME(WEEKDAY, [Day of Week] - 1))

,[Week in Month] = CONVERT(TINYINT, DENSE\_RANK() OVER

(PARTITION BY [year], [month] ORDER BY [week]))

,[Month Name] = CONVERT(VARCHAR(10), [Month Name])

,[Fiscal Year] = CONVERT(VARCHAR(5), CASE

WHEN [Month] >= 7 THEN RIGHT(YEAR([Date]),2) + '/' + RIGHT((YEAR([Date]) + 1),2)

WHEN [Month] < 7 THEN RIGHT((YEAR([Date]) - 1),2) + '/' + RIGHT(YEAR([Date]),2)

END)

,[Fiscal Quarter Number] = CONVERT(TINYINT, CASE

WHEN [Quarter] = 1 THEN 3

WHEN [Quarter] = 2 THEN 4

WHEN [Quarter] = 3 THEN 1

WHEN [Quarter] = 4 THEN 2

END)

,[Fiscal Quarter Name] = CONVERT(VARCHAR(2), CASE

WHEN [Quarter] = 1 THEN 'Q3'

WHEN [Quarter] = 2 THEN 'Q4'

WHEN [Quarter] = 3 THEN 'Q1'

WHEN [Quarter] = 4 THEN 'Q2'

END)

,[Weekend] = (CASE

WHEN [Day of Week] >= 6 THEN 1

ELSE 0

END)

,[First of Month] = [First of Month]

,[Last of Month] = DATEADD(DAY, -(DAY(DATEADD(MONTH, 1, [Date]))), DATEADD(MONTH, 1, [Date]))

FROM #temp

### Production Order Modified Example

SELECT pl.[Prod\_ Order No\_]

,pl.[Sales Order No\_]

,pl.[Segment]

,(pl.[Sales Order No\_]+'-'+pl.[Segment]) AS 'Sales Order Key'

,i.[No\_]

,i.[Description]

,i.[Global Dimension 2 Code]

,d.[Name]

,(i.[Global Dimension 2 Code] + ' - ' + d.[Name]) AS 'Item Class - Name'

,pl.[Item Category Code]

,pl.[Due Date] AS 'Assembly due date'

,pl.[Ending Date] AS 'Stockroom Due Date'

,pl.[Pick Operation Header No\_] AS 'A Type Header'

,pl.[Pick Operation Hdr\_ No\_ B-Side] AS 'B Type Header'

,ISNULL((SELECT SUM(poc.[Shortage])

FROM [TGW-ERMANCO$Prod\_ Order Component] AS poc

WHERE poc.[Prod\_ Order No\_] = pl.[Prod\_ Order No\_]

AND poc.[Status] IN (3,4)

AND poc.[Due Date] BETWEEN (GETDATE() - 30) AND (GETDATE() + 30)

),0) AS 'Sum of Shortage(s)'

,ISNULL((STUFF(

(SELECT ',' + poc.[Item No\_]

FROM [TGW-ERMANCO$Prod\_ Order Component] AS poc

WHERE poc.[Prod\_ Order No\_] = pl.[Prod\_ Order No\_]

AND poc.[Status] IN (3,4)

AND poc.[Due Date] BETWEEN (GETDATE() - 30) AND (GETDATE() + 30)

AND poc.[Shortage] = 1

FOR XML PATH ('')

)

, 1, 1, ''

)

)

,'') AS 'Shortage(s)'

,ISNULL((STUFF(

(SELECT ',' + ncrd.[No\_]

FROM [TGW-ERMANCO$NCR\_Deviation] AS ncrd

WHERE ncrd.[Prod\_ Order No\_] = pl.[Prod\_ Order No\_]

FOR XML PATH ('')

)

, 1, 1, ''

)

)

,'') AS 'Related NCRD(s)'

,pl.[Remaining Quantity]

,pr.[Run Time]

,pl.[Remaining Quantity] \* pr.[Run Time] AS 'Remaining Run Time'

,(SELECT pr2.[Work Center No\_]

FROM [TGW-ERMANCO$Prod\_ Order Routing Line] AS pr2

WHERE pl.[Prod\_ Order No\_] = pr2.[Prod\_ Order No\_]

AND pr2.[Work Center Group Code] = 'ASSY'

) AS 'Work Center'

FROM [TGW-ERMANCO$Prod\_ Order Line] AS pl

LEFT JOIN [TGW-ERMANCO$Prod\_ Order Routing Line] AS pr ON pl.[Prod\_ Order No\_] = pr.[Prod\_ Order No\_]

LEFT JOIN [TGW-ERMANCO$Item] AS i on pl.[Item No\_] = i.[No\_]

LEFT JOIN [TGW-ERMANCO$Dimension Value] AS d ON i.[Global Dimension 2 Code] = d.[Code]

WHERE pl.[Remaining Quantity] > 0

AND pr.[Work Center Group Code] = 'ASSY'

AND pl.[Due Date] BETWEEN (GETDATE() - 35) AND (GETDATE() + 35)

AND pl.[Item Category Code] = 'PRODUCT'