Isse Odawa

August, 27, 2025

IT FDN 130 A Su 25: Foundations of Databases & SQL Assignment07

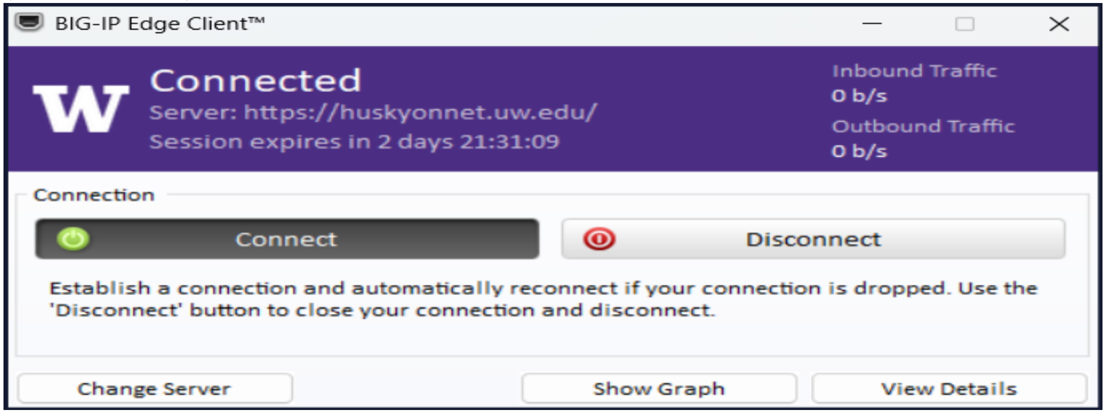
iodawa/IT-FDN-130-A-Su-25-Foundations-of-Databases-SQL-Assignment07: Assignment 07

Assignment07

Introduction
In this document I will describe assignment script code specification, requirements software tools, testing, and results in the below sections. SQL script will demonstrate creation of database, tables, inserting data in the table, testing table data with values, and generating results in table environment.

Software Application
The software application used in this assignment is SQL Server Management Studio, to run the software it is connected to remote University Server https://huskyonnet.uw.edu.

APPROX_COUNT_DISTINCT (Transact-SQL) - SQL Server | Microsoft Learn 23 August 2025)
(External Site)



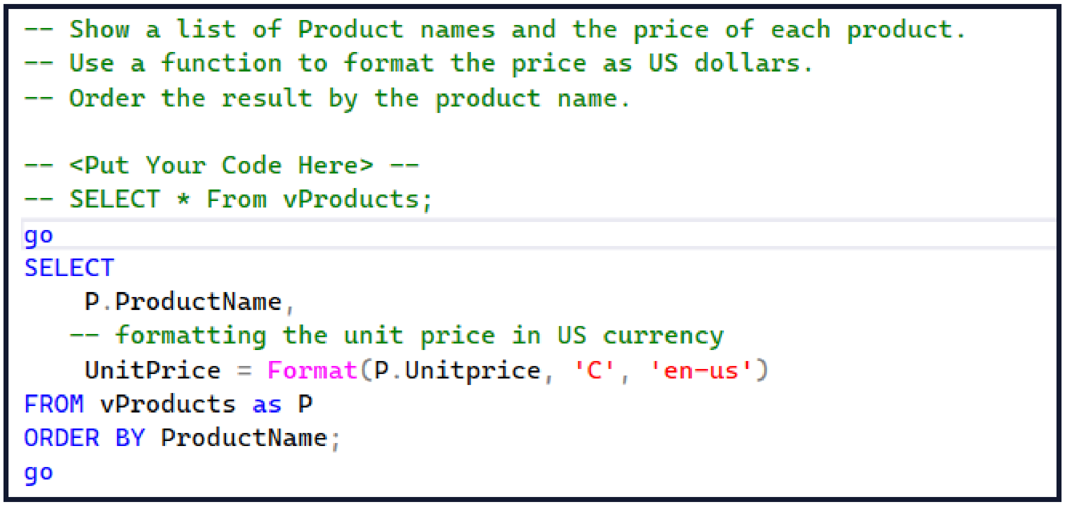
Establish a connection and automatically reconnect if your connection is dropped. Use the 'Disconnect' button to close your connection and disconnect.

Figure 0: Connection Remote Server

Requirements
In this assignment, I was given tasks for which I wrote scripts. The tasks required use functions, creating views, and creating User Defined Functions. Additionally this assignment required use functions for formatting dates for Month/year, and formatting for currency.

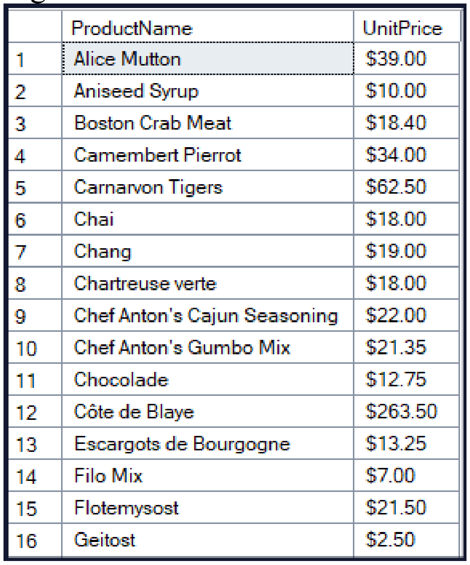
Scripts Written
Below are scripts, I have written to different tasks in this assignment.

Task 1:



```
-- Show a list of Product names and the price of each product.
-- Use a function to format the price as US dollars.
-- Order the result by the product name.

-- <Put Your Code Here> --
-- SELECT * From vProducts;
go
SELECT
    P.ProductName,
   -- formatting the unit price in US currency
    UnitPrice = Format(P.Unitprice, 'C', 'en-us')
FROM vProducts as P
ORDER BY ProductName;
go
```

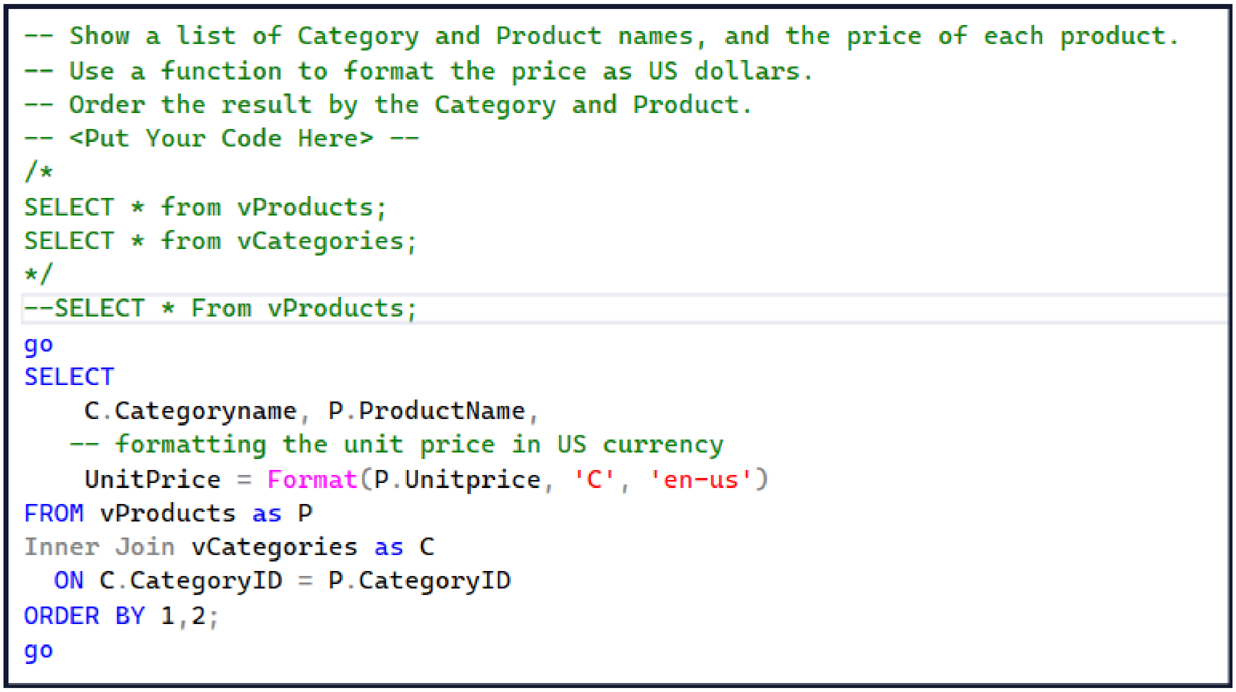
Figure 1: Task 1 Code



	ProductName	UnitPrice
1	Alice Mutton	$39.00
2	Aniseed Syrup	$10.00
3	Boston Crab Meat	$18.40
4	Camembert Pierrot	$34.00
5	Carnarvon Tigers	$62.50
6	Chai	$18.00
7	Chang	$19.00
8	Chartreuse verte	$18.00
9	Chef Anton's Cajun Seasoning	$22.00
10	Chef Anton's Gumbo Mix	$21.35
11	Chocolade	$12.75
12	Côte de Blaye	$263.50
13	Escargots de Bourgogne	$13.25
14	Filo Mix	$7.00
15	Flotemysost	$21.50
16	Geitost	$2.50

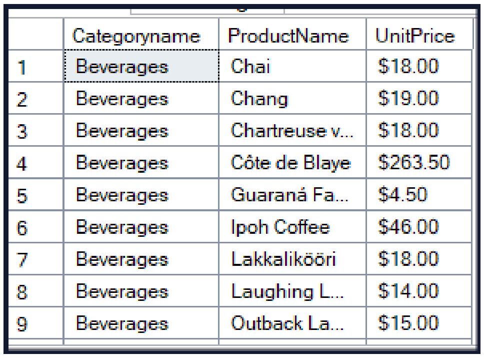
Figure 2: Task 1 Resulting table

Task 2:



```
-- Show a list of Category and Product names, and the price of each product.
-- Use a function to format the price as US dollars.
-- Order the result by the Category and Product.
-- <Put Your Code Here> --
/*
SELECT * from vProducts;
SELECT * from vCategories;
*/
--SELECT * From vProducts;
go
SELECT
    C.Categoryname, P.ProductName,
   -- formatting the unit price in US currency
    UnitPrice = Format(P.Unitprice, 'C', 'en-us')
FROM vProducts as P
Inner Join vCategories as C
  ON C.CategoryID = P.CategoryID
ORDER BY 1,2;
go
```

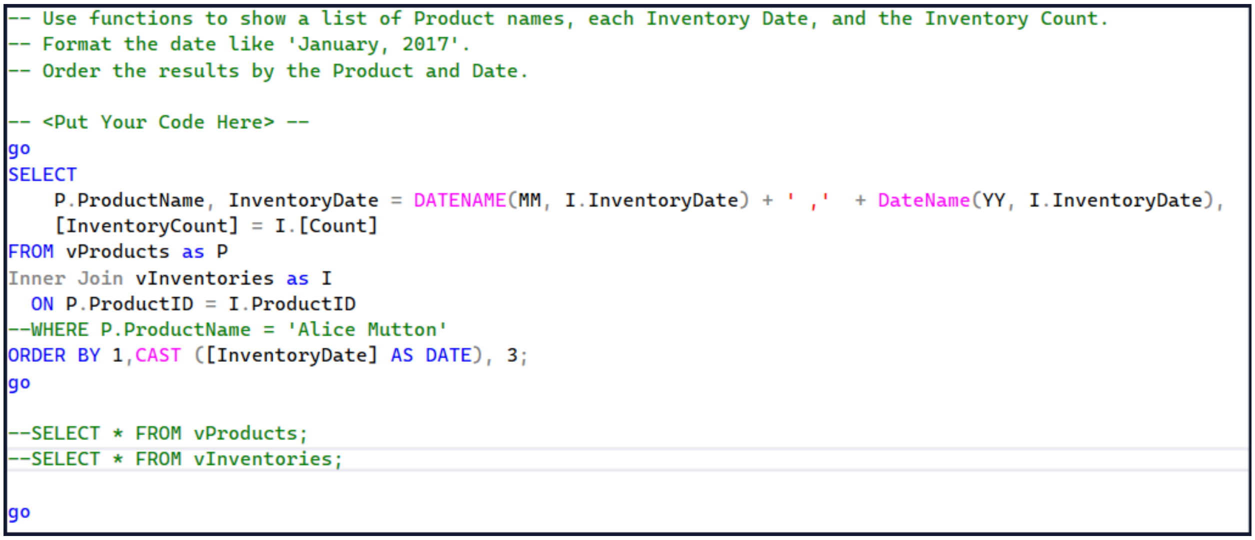
Figure 3: Task 2 Code



	Categoryname	ProductName	UnitPrice
1	Beverages	Chai	$18.00
2	Beverages	Chang	$19.00
3	Beverages	Chartreuse v...	$18.00
4	Beverages	Côte de Blaye	$263.50
5	Beverages	Guaraná Fa...	$4.50
6	Beverages	Ipoh Coffee	$46.00
7	Beverages	Lakkalikööri	$18.00
8	Beverages	Laughing L...	$14.00
9	Beverages	Outback La...	$15.00

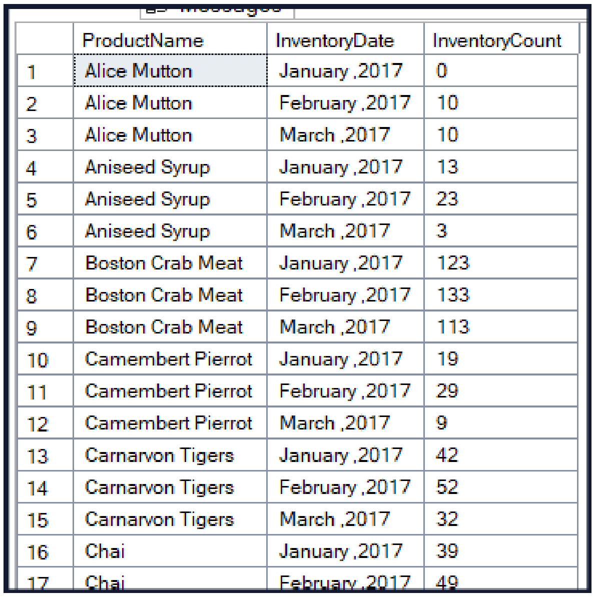
Figure 4: Task 2 Resulting table

Task 3:



```
-- Use functions to show a list of Product names, each Inventory Date, and the Inventory Count.
-- Format the date like 'January, 2017'.
-- Order the results by the Product and Date.

-- <Put Your Code Here> --
go
SELECT
    P.ProductName, InventoryDate = DATENAME(MM, I.InventoryDate) + ' ,'  + DateName(YY, I.InventoryDate),
    [InventoryCount] = I.[Count]
FROM vProducts as P
Inner Join vInventories as I
  ON P.ProductID = I.ProductID
--WHERE P.ProductName = 'Alice Mutton'
ORDER BY 1,CAST ([InventoryDate] AS DATE), 3;
go

--SELECT * FROM vProducts;
--SELECT * FROM vInventories;

go
```

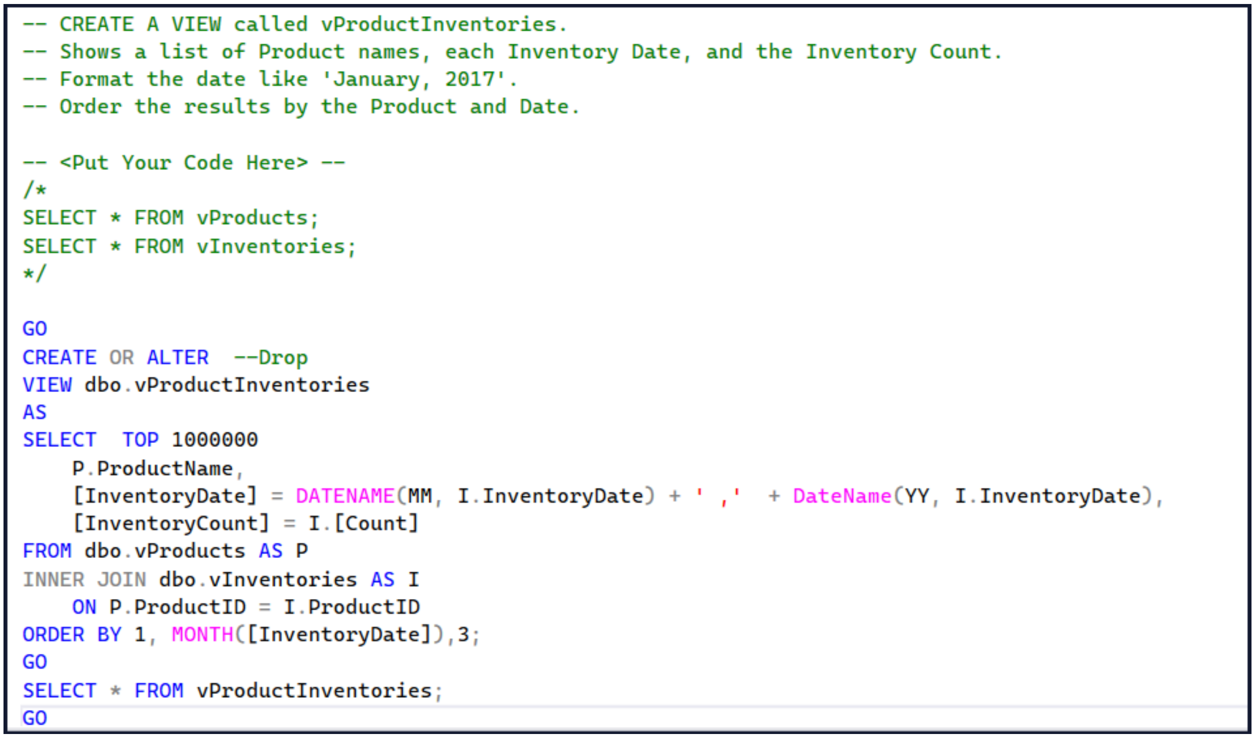
Figure 5: Task 3 Code



	ProductName	InventoryDate	InventoryCount
1	Alice Mutton	January ,2017	0
2	Alice Mutton	February ,2017	10
3	Alice Mutton	March ,2017	10
4	Aniseed Syrup	January ,2017	13
5	Aniseed Syrup	February ,2017	23
6	Aniseed Syrup	March ,2017	3
7	Boston Crab Meat	January ,2017	123
8	Boston Crab Meat	February ,2017	133
9	Boston Crab Meat	March ,2017	113
10	Camembert Pierrot	January ,2017	19
11	Camembert Pierrot	February ,2017	29
12	Camembert Pierrot	March ,2017	9
13	Carnarvon Tigers	January ,2017	42
14	Carnarvon Tigers	February ,2017	52
15	Carnarvon Tigers	March ,2017	32
16	Chai	January ,2017	39
17	Chai	February ,2017	49

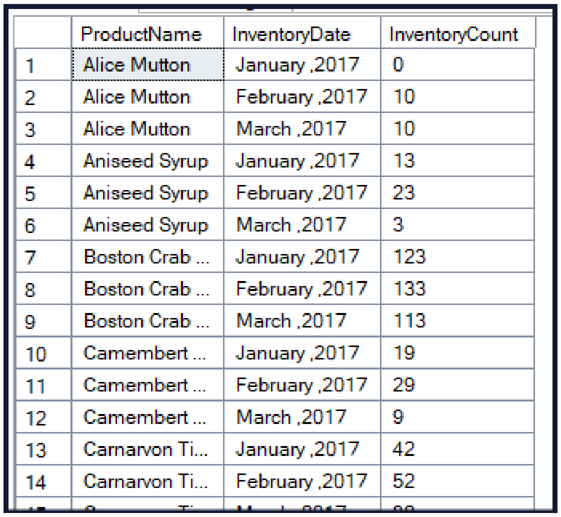
Figure 6: Task 3 Resulting table

Task 4:



```
-- CREATE A VIEW called vProductInventories.
-- Shows a list of Product names, each Inventory Date, and the Inventory Count.
-- Format the date like 'January, 2017'.
-- Order the results by the Product and Date.

-- <Put Your Code Here> --
/*
SELECT * FROM vProducts;
SELECT * FROM vInventories;
*/

GO
CREATE OR ALTER  --Drop
VIEW dbo.vProductInventories
AS
SELECT  TOP 1000000
    P.ProductName,
    [InventoryDate] = DATENAME(MM, I.InventoryDate) + ' ,'  + DateName(YY, I.InventoryDate),
    [InventoryCount] = I.[Count]
FROM dbo.vProducts AS P
INNER JOIN dbo.vInventories AS I
    ON P.ProductID = I.ProductID
ORDER BY 1, MONTH([InventoryDate]),3;
GO
SELECT * FROM vProductInventories;
GO
```

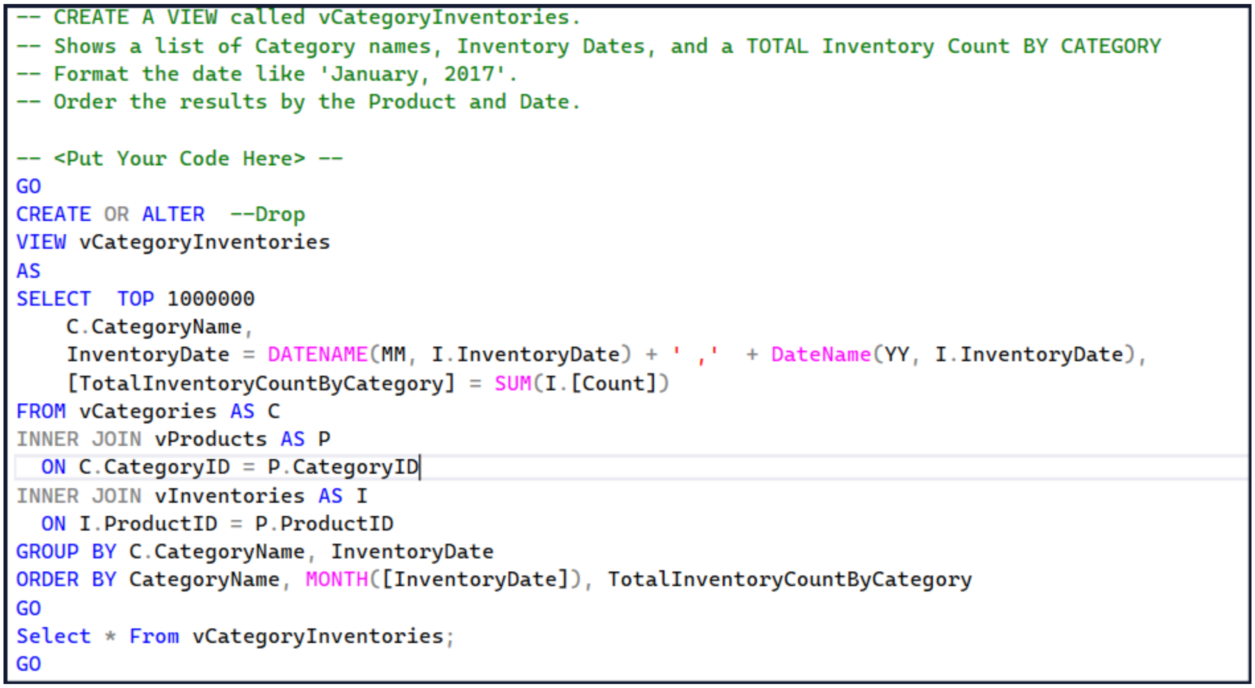
Figure 7: Task 4 Code



	ProductName	InventoryDate	InventoryCount
1	Alice Mutton	January ,2017	0
2	Alice Mutton	February ,2017	10
3	Alice Mutton	March ,2017	10
4	Aniseed Syrup	January ,2017	13
5	Aniseed Syrup	February ,2017	23
6	Aniseed Syrup	March ,2017	3
7	Boston Crab ...	January ,2017	123
8	Boston Crab ...	February ,2017	133
9	Boston Crab ...	March ,2017	113
10	Camembert ...	January ,2017	19
11	Camembert ...	February ,2017	29
12	Camembert ...	March ,2017	9
13	Carnarvon Ti...	January ,2017	42
14	Carnarvon Ti...	February ,2017	52

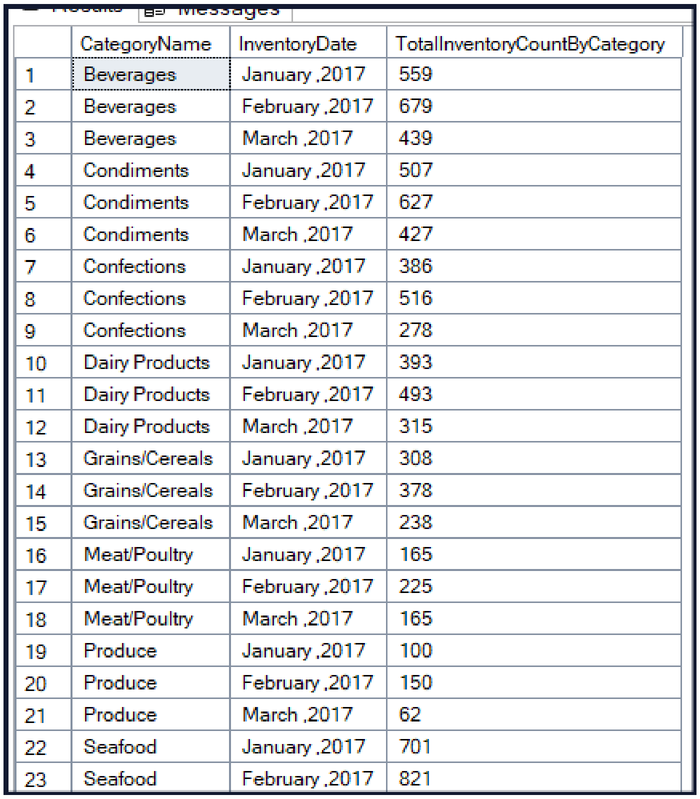
Figure 8: Task 4 Resulting table

Task 5:



```
-- CREATE A VIEW called vCategoryInventories.
-- Shows a list of Category names, Inventory Dates, and a TOTAL Inventory Count BY CATEGORY
-- Format the date like 'January, 2017'.
-- Order the results by the Product and Date.

-- <Put Your Code Here> --
GO
CREATE OR ALTER  --Drop
VIEW vCategoryInventories
AS
SELECT  TOP 1000000
    C.CategoryName,
    InventoryDate = DATENAME(MM, I.InventoryDate) + ' ,'  + DateName(YY, I.InventoryDate),
    [TotalInventoryCountByCategory] = SUM(I.[Count])
FROM vCategories AS C
INNER JOIN vProducts AS P
  ON C.CategoryID = P.CategoryID
INNER JOIN vInventories AS I
  ON I.ProductID = P.ProductID
GROUP BY C.CategoryName, InventoryDate
ORDER BY CategoryName, MONTH([InventoryDate]), TotalInventoryCountByCategory
GO
Select * From vCategoryInventories;
GO
```

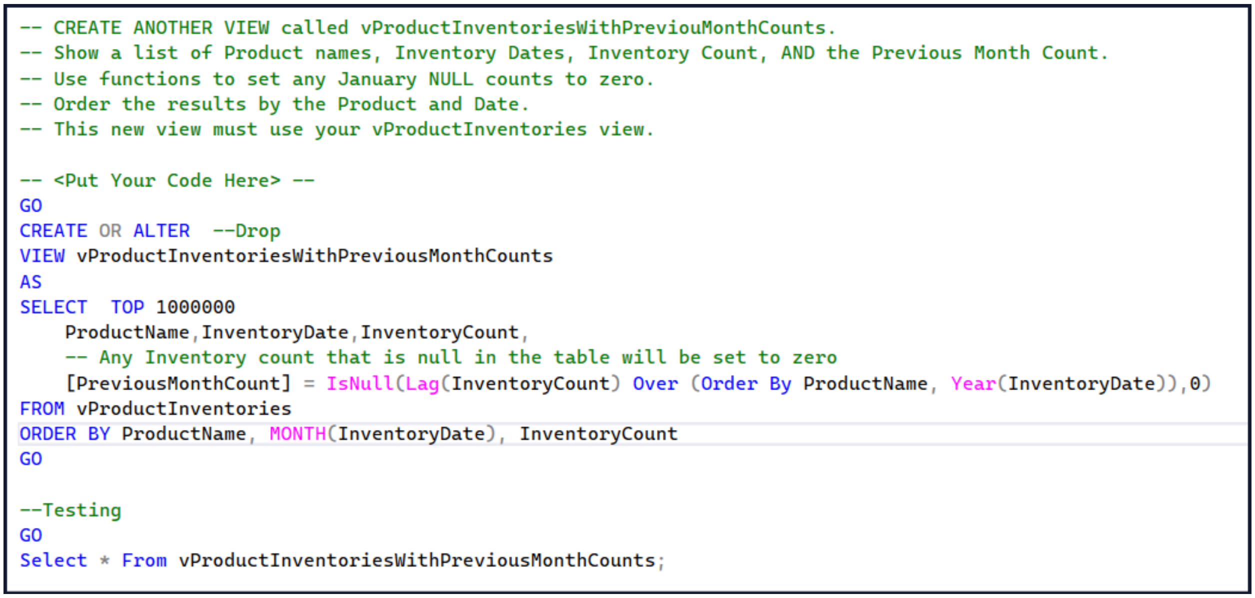
Figure 9: Task 5 Code



	CategoryName	InventoryDate	TotalInventoryCountByCategory
1	Beverages	January ,2017	559
2	Beverages	February ,2017	679
3	Beverages	March ,2017	439
4	Condiments	January ,2017	507
5	Condiments	February ,2017	627
6	Condiments	March ,2017	427
7	Confections	January ,2017	386
8	Confections	February ,2017	516
9	Confections	March ,2017	278
10	Dairy Products	January ,2017	393
11	Dairy Products	February ,2017	493
12	Dairy Products	March ,2017	315
13	Grains/Cereals	January ,2017	308
14	Grains/Cereals	February ,2017	378
15	Grains/Cereals	March ,2017	238
16	Meat/Poultry	January ,2017	165
17	Meat/Poultry	February ,2017	225
18	Meat/Poultry	March ,2017	165
19	Produce	January ,2017	100
20	Produce	February ,2017	150
21	Produce	March ,2017	62
22	Seafood	January ,2017	701
23	Seafood	February ,2017	821

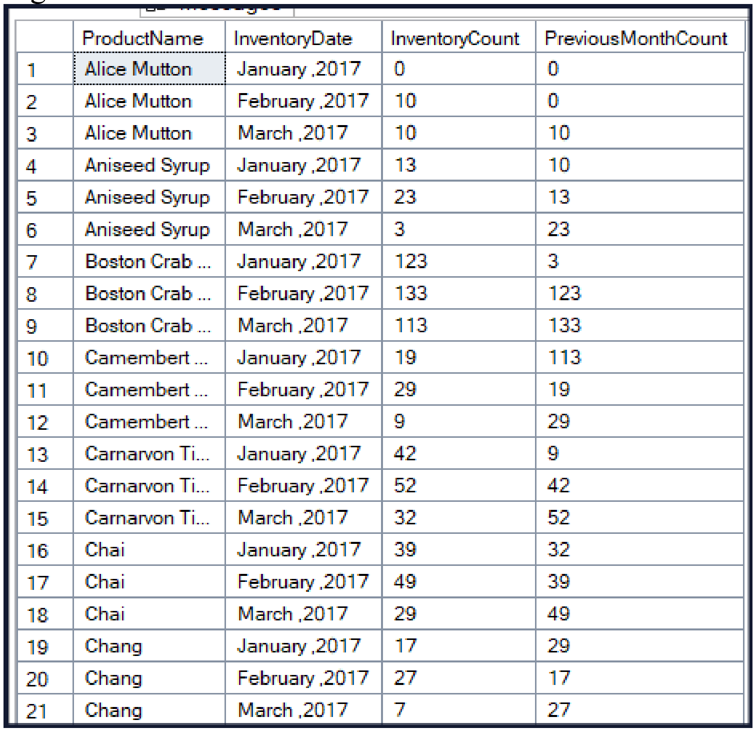
Figure 10: Task 5 Resulting table

Task 6:



```
-- CREATE ANOTHER VIEW called vProductInventoriesWithPreviouMonthCounts.
-- Show a list of Product names, Inventory Dates, Inventory Count, AND the Previous Month Count.
-- Use functions to set any January NULL counts to zero.
-- Order the results by the Product and Date.
-- This new view must use your vProductInventories view.

-- <Put Your Code Here> --
GO
CREATE OR ALTER  --Drop
VIEW vProductInventoriesWithPreviousMonthCounts
AS
SELECT  TOP 1000000
    ProductName,InventoryDate,InventoryCount,
    -- Any Inventory count that is null in the table will be set to zero
    [PreviousMonthCount] = IsNull(Lag(InventoryCount) Over (Order By ProductName, Year(InventoryDate)),0)
FROM vProductInventories
ORDER BY ProductName, MONTH(InventoryDate), InventoryCount
GO

--Testing
GO
Select * From vProductInventoriesWithPreviousMonthCounts;
```

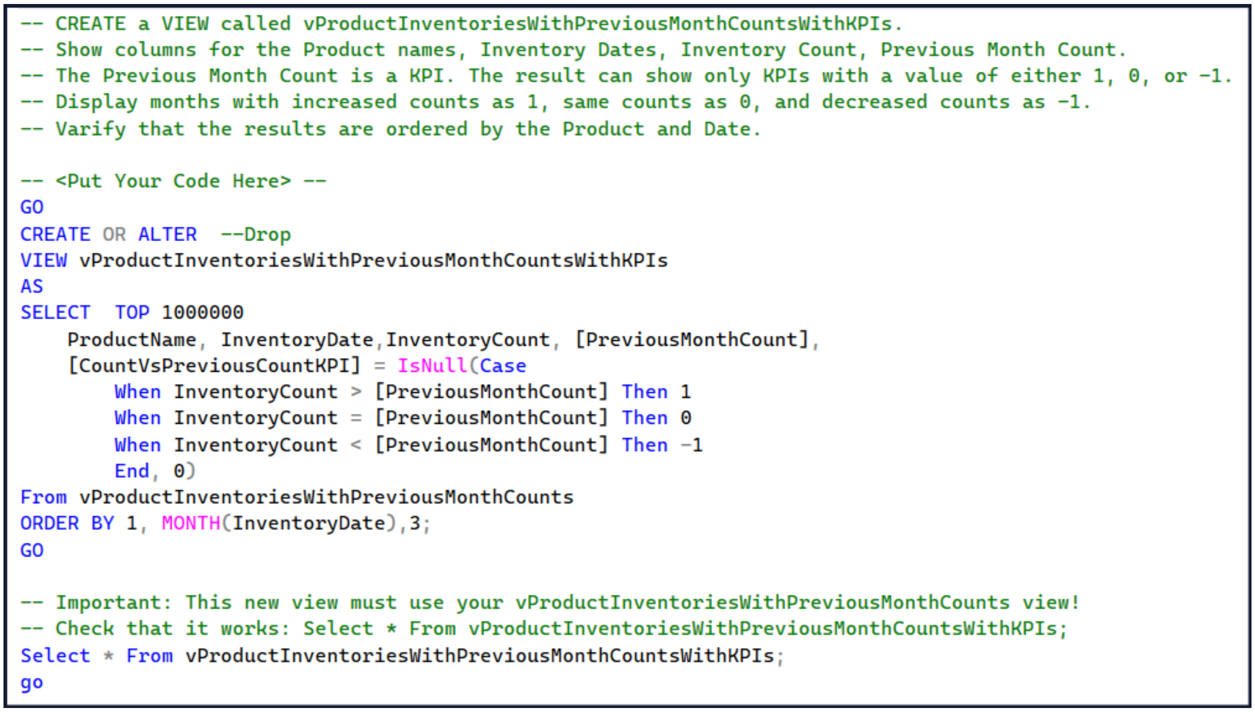
Figure 11: Task 6 Code



	ProductName	InventoryDate	InventoryCount	PreviousMonthCount
1	Alice Mutton	January ,2017	0	0
2	Alice Mutton	February ,2017	10	0
3	Alice Mutton	March ,2017	10	10
4	Aniseed Syrup	January ,2017	13	10
5	Aniseed Syrup	February ,2017	23	13
6	Aniseed Syrup	March ,2017	3	23
7	Boston Crab ...	January ,2017	123	3
8	Boston Crab ...	February ,2017	133	123
9	Boston Crab ...	March ,2017	113	133
10	Camembert ...	January ,2017	19	113
11	Camembert ...	February ,2017	29	19
12	Camembert ...	March ,2017	9	29
13	Carnarvon Ti...	January ,2017	42	9
14	Carnarvon Ti...	February ,2017	52	42
15	Carnarvon Ti...	March ,2017	32	52
16	Chai	January ,2017	39	32
17	Chai	February ,2017	49	39
18	Chai	March ,2017	29	49
19	Chang	January ,2017	17	29
20	Chang	February ,2017	27	17
21	Chang	March ,2017	7	27

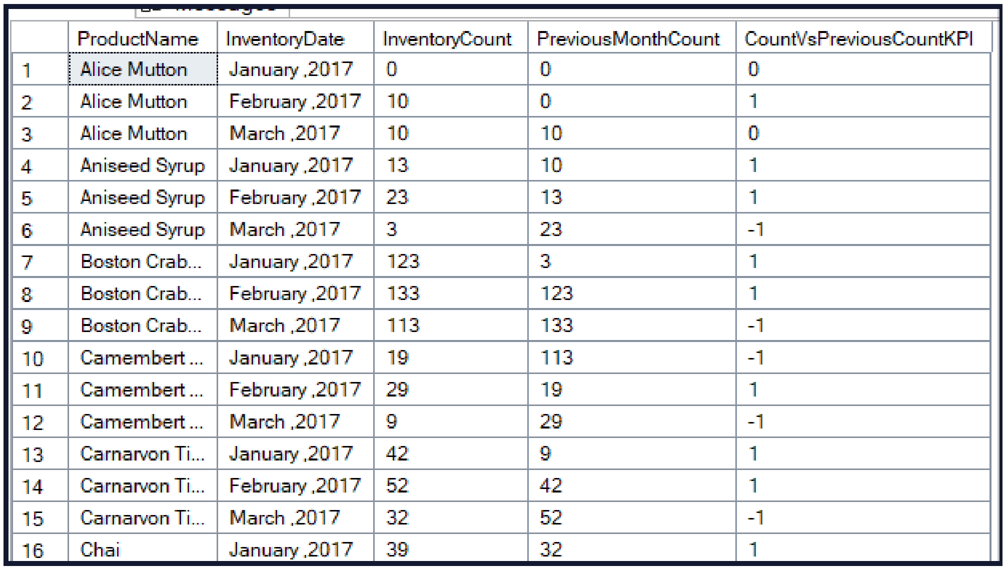
Figure 12: Task 6 Resulting table

Task 7:



```
-- CREATE a VIEW called vProductInventoriesWithPreviousMonthCountsWithKPIs.
-- Show columns for the Product names, Inventory Dates, Inventory Count, Previous Month Count.
-- The Previous Month Count is a KPI. The result can show only KPIs with a value of either 1, 0, or -1.
-- Display months with increased counts as 1, same counts as 0, and decreased counts as -1.
-- Varify that the results are ordered by the Product and Date.

-- <Put Your Code Here> --
GO
CREATE OR ALTER  --Drop
VIEW vProductInventoriesWithPreviousMonthCountsWithKPIs
AS
SELECT  TOP 1000000
    ProductName, InventoryDate,InventoryCount, [PreviousMonthCount],
    [CountVsPreviousCountKPI] = IsNull(Case
        When InventoryCount > [PreviousMonthCount] Then 1
        When InventoryCount = [PreviousMonthCount] Then 0
        When InventoryCount < [PreviousMonthCount] Then -1
        End, 0)
From vProductInventoriesWithPreviousMonthCounts
ORDER BY 1, MONTH(InventoryDate),3;
GO

-- Important: This new view must use your vProductInventoriesWithPreviousMonthCounts view!
-- Check that it works: Select * From vProductInventoriesWithPreviousMonthCountsWithKPIs;
Select * From vProductInventoriesWithPreviousMonthCountsWithKPIs;
go
```

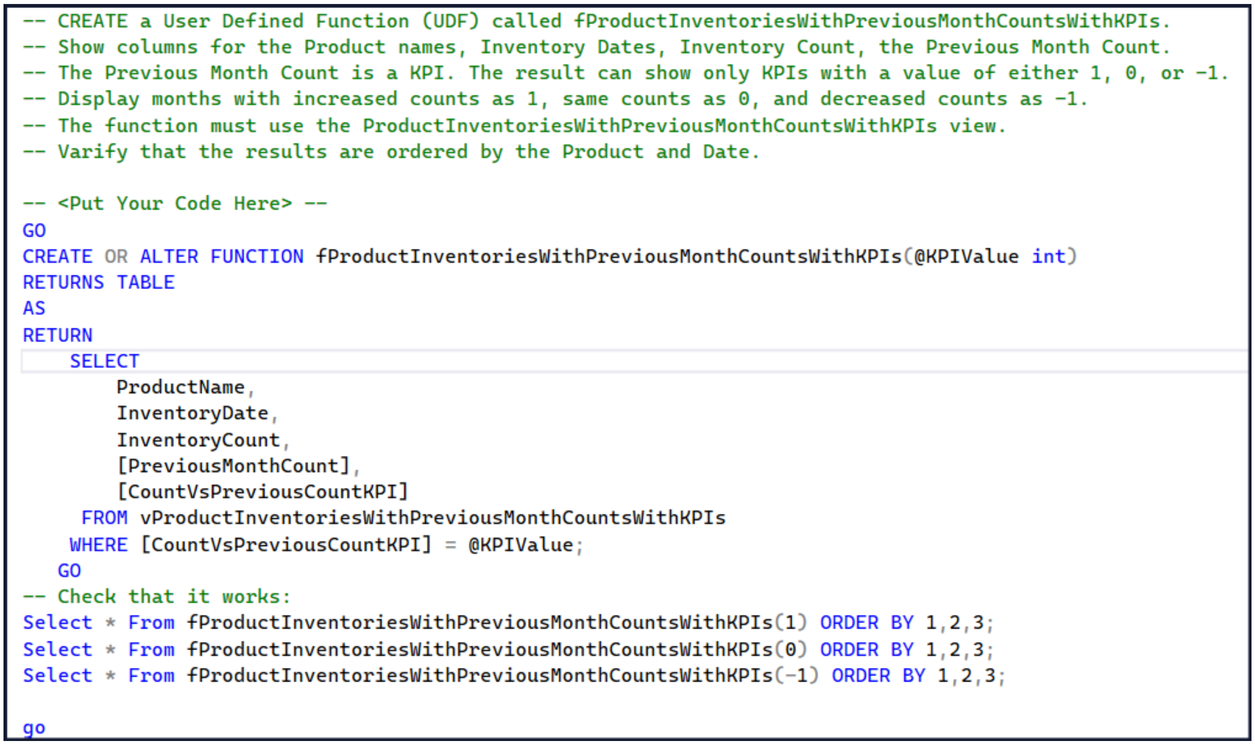
Figure 13: Task 7 Code



	ProductName	InventoryDate	InventoryCount	PreviousMonthCount	CountVsPreviousCountKPI
1	Alice Mutton	January ,2017	0	0	0
2	Alice Mutton	February ,2017	10	0	1
3	Alice Mutton	March ,2017	10	10	0
4	Aniseed Syrup	January ,2017	13	10	1
5	Aniseed Syrup	February ,2017	23	13	1
6	Aniseed Syrup	March ,2017	3	23	-1
7	Boston Crab...	January ,2017	123	3	1
8	Boston Crab...	February ,2017	133	123	1
9	Boston Crab...	March ,2017	113	133	-1
10	Camembert ...	January ,2017	19	113	-1
11	Camembert ...	February ,2017	29	19	1
12	Camembert ...	March ,2017	9	29	-1
13	Carnarvon Ti...	January ,2017	42	9	1
14	Carnarvon Ti...	February ,2017	52	42	1
15	Carnarvon Ti...	March ,2017	32	52	-1
16	Chai	January ,2017	39	32	1

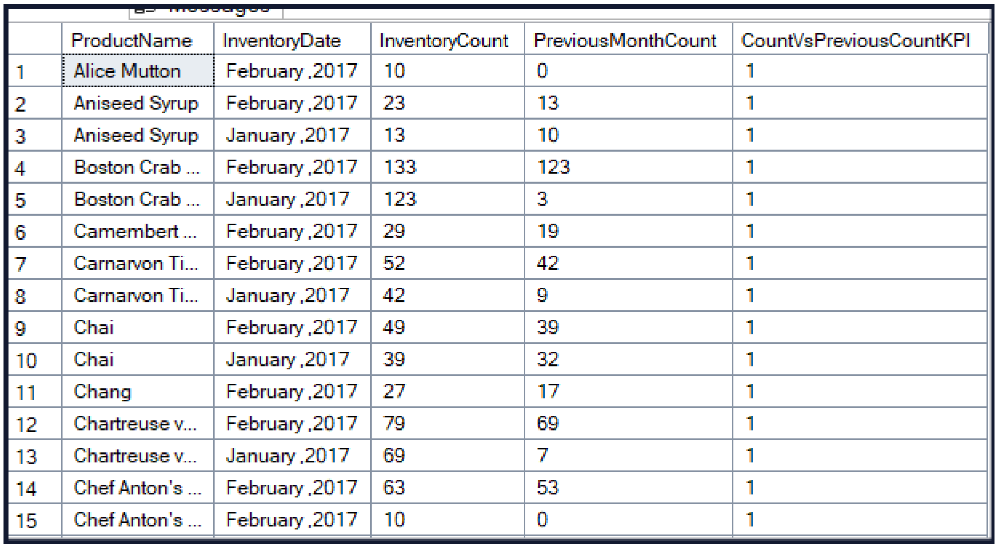
Figure 14: Task 7 Resulting table

Task 8:



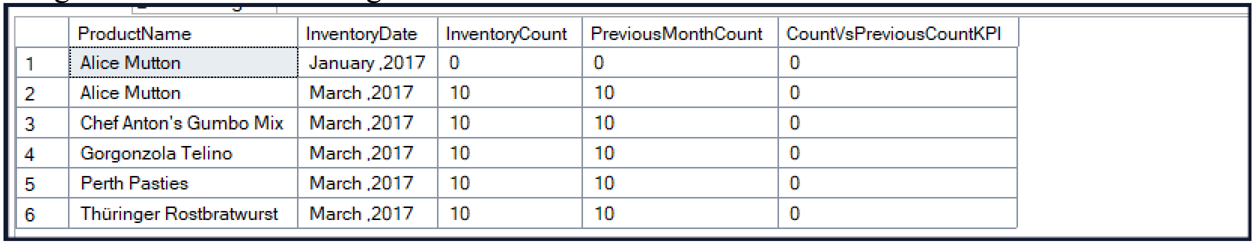
```
-- CREATE a User Defined Function (UDF) called fProductInventoriesWithPreviousMonthCountsWithKPIs.
-- Show columns for the Product names, Inventory Dates, Inventory Count, the Previous Month Count.
-- The Previous Month Count is a KPI. The result can show only KPIs with a value of either 1, 0, or -1.
-- Display months with increased counts as 1, same counts as 0, and decreased counts as -1.
-- The function must use the ProductInventoriesWithPreviousMonthCountsWithKPIs view.
-- Varify that the results are ordered by the Product and Date.

-- <Put Your Code Here> --
GO
CREATE OR ALTER FUNCTION fProductInventoriesWithPreviousMonthCountsWithKPIs(@KPIValue int)
RETURNS TABLE
AS
RETURN
    SELECT
        ProductName,
        InventoryDate,
        InventoryCount,
        [PreviousMonthCount],
        [CountVsPreviousCountKPI]
     FROM vProductInventoriesWithPreviousMonthCountsWithKPIs
    WHERE [CountVsPreviousCountKPI] = @KPIValue;
   GO
-- Check that it works:
Select * From fProductInventoriesWithPreviousMonthCountsWithKPIs(1) ORDER BY 1,2,3;
Select * From fProductInventoriesWithPreviousMonthCountsWithKPIs(0) ORDER BY 1,2,3;
Select * From fProductInventoriesWithPreviousMonthCountsWithKPIs(-1) ORDER BY 1,2,3;

go
```

Figure 15: Task 8 Code



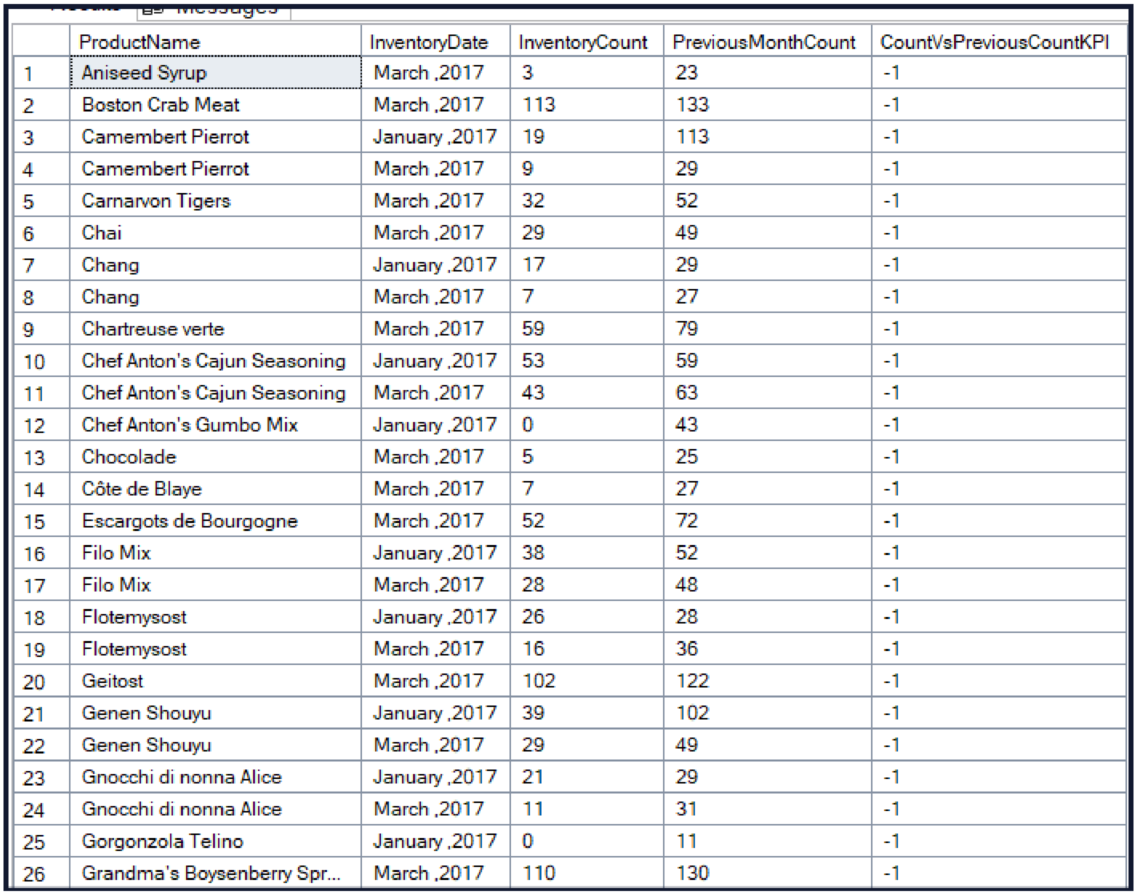
	ProductName	InventoryDate	InventoryCount	PreviousMonthCount	CountVsPreviousCountKPI
1	Alice Mutton	February ,2017	10	0	1
2	Aniseed Syrup	February ,2017	23	13	1
3	Aniseed Syrup	January ,2017	13	10	1
4	Boston Crab ...	February ,2017	133	123	1
5	Boston Crab ...	January ,2017	123	3	1
6	Camembert ...	February ,2017	29	19	1
7	Carnarvon Ti...	February ,2017	52	42	1
8	Carnarvon Ti...	January ,2017	42	9	1
9	Chai	February ,2017	49	39	1
10	Chai	January ,2017	39	32	1
11	Chang	February ,2017	27	17	1
12	Chartreuse v...	February ,2017	79	69	1
13	Chartreuse v...	January ,2017	69	7	1
14	Chef Anton's ...	February ,2017	63	53	1
15	Chef Anton's ...	February ,2017	10	0	1

Figure 16: Task 8 Resulting table



	ProductName	InventoryDate	InventoryCount	PreviousMonthCount	CountVsPreviousCountKPI
1	Alice Mutton	January ,2017	0	0	0
2	Alice Mutton	March ,2017	10	10	0
3	Chef Anton's Gumbo Mix	March ,2017	10	10	0
4	Gorgonzola Telino	March ,2017	10	10	0
5	Perth Pasties	March ,2017	10	10	0
6	Thüringer Rostbratwurst	March ,2017	10	10	0

Figure 17 : Task 8 KPIs with a value 0 result



	ProductName	InventoryDate	InventoryCount	PreviousMonthCount	CountVsPreviousCountKPI
1	Aniseed Syrup	March ,2017	3	23	-1
2	Boston Crab Meat	March ,2017	113	133	-1
3	Camembert Pierrot	January ,2017	19	113	-1
4	Camembert Pierrot	March ,2017	9	29	-1
5	Carnarvon Tigers	March ,2017	32	52	-1
6	Chai	March ,2017	29	49	-1
7	Chang	January ,2017	17	29	-1
8	Chang	March ,2017	7	27	-1
9	Chartreuse verte	March ,2017	59	79	-1
10	Chef Anton's Cajun Seasoning	January ,2017	53	59	-1
11	Chef Anton's Cajun Seasoning	March ,2017	43	63	-1
12	Chef Anton's Gumbo Mix	January ,2017	0	43	-1
13	Chocolade	March ,2017	5	25	-1
14	Côte de Blaye	March ,2017	7	27	-1
15	Escargots de Bourgogne	March ,2017	52	72	-1
16	Filo Mix	January ,2017	38	52	-1
17	Filo Mix	March ,2017	28	48	-1
18	Flotemysost	January ,2017	26	28	-1
19	Flotemysost	March ,2017	16	36	-1
20	Geitost	March ,2017	102	122	-1
21	Genen Shouyu	January ,2017	39	102	-1
22	Genen Shouyu	March ,2017	29	49	-1
23	Gnocchi di nonna Alice	January ,2017	21	29	-1
24	Gnocchi di nonna Alice	March ,2017	11	31	-1
25	Gorgonzola Telino	January ,2017	0	11	-1
26	Grandma's Boysenberry Spr...	March ,2017	110	130	-1

Figure   : Task 8 KPIs with a value -1 result

Summary

In this Assignment, I learned how to create SQL Views, secure the scripts, control access to views, and generate reusable modular SQL Vies. Encouraged to write scripts to join data from different tables and generate data in a table relative how the tables data correspond to each other. The understanding of fk and pk relationship among tables contributed to writing to SQL scripts for this assignment tasks. The tasks required use functions, creating views, and creating User Defined Functions. Additionally, this assignment required use functions for formatting dates for Month/year, and formatting for currency.

This was great exercise of creating modular reusable, secure SQL Views. Finaly I have uploaded this assign in Git Hub to provide classmates to comment on my assignment and use their feedback.