USERACCOUNT TABLE:

CREATE TABLE [dbo].[UserAccount](

[CustomerId] [bigint] IDENTITY(1,1) NOT NULL,

[FirstName] [varchar](64) NULL,

[LastName] [varchar](64) NULL,

[EmailAddress] [varchar](64) NULL,

[MobileNumber] [varchar](16) NULL,

[IsActive] [bit] NOT NULL,

[Password] [varchar](128) NULL,

[CreatedDate] [datetime] NOT NULL,

[LastLoginDate] [datetime] NULL,

[LastUpdate] [datetime] NULL,

CONSTRAINT [PK\_Account] PRIMARY KEY CLUSTERED

(

[CustomerId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[UserAccount] ADD CONSTRAINT [DF\_Account\_CustomerStatus] DEFAULT ((1)) FOR [IsActive]

GO

ALTER TABLE [dbo].[UserAccount] ADD CONSTRAINT [DF\_Account\_CreatedDate] DEFAULT (getdate()) FOR [CreatedDate]

GO

PRODUCT TABLE

CREATE TABLE [dbo].[Product](

[ProductId] [bigint] IDENTITY(1,1) NOT NULL,

[ProductName] [nvarchar](255) NOT NULL,

ProductPrice Money NOT NULL,

ProductSize nvarchar(5),

ProductColor nVarchar(64),

[IsActive] [bit] NOT NULL,

[CreatedDate] [datetime] NOT NULL,

[LastUpdate] [datetime] NULL,

[UpsertedBy] [bigint] NOT NULL,

CONSTRAINT [PK\_Product] PRIMARY KEY CLUSTERED

(

[ProductId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

ALTER TABLE [dbo].[Product] WITH NOCHECK ADD CONSTRAINT [FK\_Product\_UserAccount] FOREIGN KEY([UpsertedBy])

REFERENCES [dbo].[UserAccount] ([CustomerId])

GO

ALTER TABLE [dbo].[Product] CHECK CONSTRAINT [FK\_Product\_UserAccount]

GO

ORDER TABLE:

CREATE TABLE [dbo].[PurchaseOrderDetail](

[OrderId] [bigint] NOT NULL,

[OrderDetailID] [bigint] NOT NULL,

[OrderQty] Int NOT NULL,

[ProductID] Bigint NOT NULL,

[OrderDate] [datetime] NOT NULL,

[AdminId] [bigint] NULL,

[CreatedDate] [datetime] NULL,

CONSTRAINT [PK\_Order] PRIMARY KEY CLUSTERED

(

[OrderId] ASC,[OrderDetailID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[PurchaseOrderDetail] WITH CHECK ADD CONSTRAINT [FK\_PurchaseOrderDetail\_Product\_ProductID] FOREIGN KEY([ProductID])

REFERENCES [dbo].[Product] ([ProductID])

GO

ALTER TABLE [dbo].[PurchaseOrderDetail] CHECK CONSTRAINT [FK\_PurchaseOrderDetail\_Product\_ProductID]

GO

ALTER TABLE [dbo].[PurchaseOrderDetail] WITH CHECK ADD CONSTRAINT [CK\_PurchaseOrderDetail\_OrderQty] CHECK (([OrderQty]>(0)))

GO

ALTER TABLE [dbo].[PurchaseOrderDetail] CHECK CONSTRAINT [CK\_PurchaseOrderDetail\_OrderQty]

GO

TRIGGER :

CREATE TABLE ProductAudit

(

AuditID INT NOT NULL IDENTITY(1, 1) ,

ProductID BIGINT ,

ProductName NVARCHAR(255) ,

ProductPrice Money,

[ProductSize] [nvarchar](5) NULL,

[ProductColor] [nvarchar](64) NULL,

ModifiedBy VARCHAR(128) ,

ModifiedDate DATETIME ,

Operation CHAR(1)

PRIMARY KEY CLUSTERED ( AuditID )

)

CREATE TRIGGER TR\_Audit\_Product ON dbo.Product

FOR INSERT, UPDATE, DELETE

AS

DECLARE @login\_name VARCHAR(128)

SELECT @login\_name = login\_name

FROM sys.dm\_exec\_sessions

WHERE session\_id = @@SPID

IF EXISTS ( SELECT 0 FROM Deleted )

BEGIN

IF EXISTS ( SELECT 0 FROM Inserted )

BEGIN

INSERT INTO ProductAudit

( ProductID,

ProductName,

ProductPrice,

ProductSize,

ProductColor,

ModifiedBy ,

ModifiedDate ,

Operation

)

SELECT D.ProductID ,

D.ProductName ,

D.ProductPrice ,

D.ProductSize ,

D.ProductColor ,

@login\_name ,

GETDATE() ,

'U'

FROM Deleted D

END

ELSE

BEGIN

INSERT INTO ProductAudit

( ProductID,

ProductName,

ProductPrice,

ProductSize,

ProductColor,

ModifiedBy ,

ModifiedDate ,

Operation

)

SELECT D.ProductID ,

D.ProductName ,

D.ProductPrice ,

D.ProductSize ,

D.ProductColor ,

@login\_name ,

GETDATE() ,

'D'

FROM Deleted D

END

END

ELSE

BEGIN

INSERT INTO ProductAudit

( ProductID,

ProductName,

ProductPrice,

ProductSize,

ProductColor,

ModifiedBy ,

ModifiedDate ,

Operation

)

SELECT I.ProductID ,

I.ProductName ,

I.ProductPrice ,

I.ProductSize ,

I.ProductColor ,

@login\_name ,

GETDATE() ,

'I'

FROM Inserted I

END

GO

CREATE TABLE [dbo].[ProductList](

[ProductId] [bigint] IDENTITY(1,1) NOT NULL,

[ProductName] [varchar](max) NOT NULL,

ProductPrice Money NOT NULL,

ProductSize nvarchar(5),

ProductColor nVarchar(64),

[IsActive] [bit] NOT NULL,

[CreatedDate] [datetime] NOT NULL,

[LastUpdate] [datetime] NULL,

[UpsertedBy] [bigint] NOT NULL,

CONSTRAINT [PK\_ProductList] PRIMARY KEY CLUSTERED

(

[ProductId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

GO

PRODUCTLIST UPLOAD:

MERGE Product AS TARGET

USING ProductList AS SOURCE

ON (TARGET.ProductID = SOURCE.ProductID)

WHEN MATCHED AND TARGET.ProductName <> SOURCE.ProductName OR TARGET.ProductPrice <> SOURCE.ProductPrice or

TARGET.ProductSize <> SOURCE.ProductSize or TARGET.ProductColor <> SOURCE.ProductColor

THEN UPDATE SET TARGET.ProductName = SOURCE.ProductName, TARGET.ProductPrice = SOURCE.ProductPrice,

TARGET.ProductSize = SOURCE.ProductSize ,TARGET.ProductColor = SOURCE.ProductColor

WHEN NOT MATCHED BY TARGET

THEN INSERT (ProductID, ProductName, ProductPrice,ProductSize,ProductColor) VALUES (SOURCE.ProductID, SOURCE.ProductName, SOURCE.ProductPrice,SOURCE.ProductSize,SOURCE.ProductColor)

WHEN NOT MATCHED BY SOURCE

THEN DELETE

OUTPUT $action,

DELETED.ProductID AS TargetProductID,

DELETED.ProductName AS TargetProductName,

DELETED.ProductPrice AS TargetProductPrice,

DELETED.ProductSize AS TargetProductSize,

DELETED.ProductColor AS TargetProductColor,

INSERTED.ProductID AS SourceProductID,

INSERTED.ProductName AS SourceProductName,

INSERTED.ProductPrice AS SourceProductPrice,

INSERTED.ProductSize AS SourceProductSize,

INSERTED.ProductColor AS SourceProductColor;

ORDERITEM DETAIL REPORT:

DECLARE @Size AS NVARCHAR(MAX),

@Color AS NVARCHAR(MAX),

@query AS NVARCHAR(MAX);

SET @Size = STUFF((SELECT distinct ',' + QUOTENAME(p.ProductSize)

FROM dbo.product p inner join [PurchaseOrderDetail] pod

on pod.ProductID=p.productid

FOR XML PATH(''), TYPE

).value('.', 'NVARCHAR(MAX)')

,1,1,'')

SET @Color = STUFF((SELECT distinct ',' + QUOTENAME(p.ProductColor)

FROM dbo.product p inner join [PurchaseOrderDetail] pod

on pod.ProductID=p.productid

FOR XML PATH(''), TYPE

).value('.', 'NVARCHAR(MAX)')

,1,1,'')

set @query = 'SELECT \*, ' + @Size + ',' + @Color + ' from

(

SELECT

P.ProductName,

p.ProductPrice,

pod.OrderQty,

pod.OrderQty as OrderQty1,

ProductSize,

ProductColor,

pod.OrderDate

list\_price

FROM dbo.product p inner join [PurchaseOrderDetail] pod

on pod.ProductID=p.productid

) x

pivot

(

sum(OrderQty)

for category in (' + @Size + ')

) p

pivot

(

sum(OrderQty1)

for category in (' + @Color + ')

ORDER BY

ProductPrice,

ProductName

OFFSET 10 ROWS

FETCH NEXT 10 ROWS ONLY;'