* **TRIGGER:**

***Trigger on Product Tables***

*/\* first trigger \*/*

*Create trigger tr\_Product*

*ON product*

*for Update*

*as*

*begin*

*/\* Update into audit table to maintain history \*/*

*Insert into audit\_history values*

*select getdate(),'Product','Update',ORIGINAL\_LOGIN(),d.attributevalue,i.attributevalue*

*from product p*

*inner join inserted i on i.product=p.product\_name*

*inner join deleted d on d.product=d.product\_name*

*end*

*/\* Second trigger \*/*

*Create trigger tr\_Product*

*ON product*

*for Insert*

*as*

*begin*

*/\* Insert into audit table to maintain history \*/*

*Insert into audit\_history values*

*select getdate(),'Insert',ORIGINAL\_LOGIN(),isnull(d.attributevalue,''),i.attributevalue*

*from product p*

*inner join inserted i on i.product=p.product\_name*

*inner join deleted d on d.product=d.product\_name*

*end*

SAME TRIGGER FOR other tables as well to keep track of events.

**Create Trigger to check whether the user who is editing the details of product and attributes is Administrator of not. If the user is Administrator, then they can edit the details, if not then error of not allowed.**

*CREATE TRIGGER tr\_UserCheck*

*ON DATABASE*

*FOR LOGON AS  
BEGIN  
  DECLARE @LoginName nvarchar(100)  
  SET @LoginName = original\_login()  
  IF(role <>'Administrator' FROM USER WHERE login\_name=  
  (  
         SELECT login\_name  
         FROM   sys.dm\_exec\_sessions  
         WHERE  login\_time=getdate() ))  
  BEGIN  
     Print 'Access Denied. Only Administrators are allowed to edit details'*

*ROLLBACK  
  END  
END*

* **Bulk Upload:**

Client will upload the csv file of product and its details as follows using SQL server Import/ Export Wizard. Excel tab2 contains sample csv.

Procedure **sp\_bulk\_insert\_update** will be used to insert/update the csv data

* **View Order Report:**

*CREATE PROCEDURE spGetProductsByPage*

*@PageNumber INT,*

*@PageSize INT*

*AS*

*BEGIN*

*SELECT p.Product\_Name, a.Price, o.Order\_Quantity,s.Size, c.Color, a.Other\_Attribtes,o.Order\_Created\_Time*

*FROM Product p*

*inner join attributes a on a.Product\_id\_fk=p.product\_id*

*left join size s on s.Product\_id\_fk=p.product\_id*

*left join color c on c.Product\_id\_fk=p.product\_id*

*inner join order o on o.Product\_id\_fk=p.product\_id*

*ORDER BY Id*

*OFFSET (@PageNumber - 1) \* @PageSize ROWS*

*FETCH NEXT @PageSize ROWS ONLY*

*END*

*EXECUTE spGetRowsByPageNumberAndSize 2, 5*

*/\* With PageNumber = 3 and PageSize = 10, the stored procedure returns the correct set of rows \*/*

* *Clustered index will automatically be created with the tables as we are creating primary key on tables. Creation of primary key results in creation of clustered index.*