# 1.Orders table to store details

**Order Details Table**

Order id(**int**)(identity column), item id(**int**)(foreign key), order date(datetime)

**Item Details Table**

Item id(**int**)(primary key),Item\_name(**Nvarchar(100)**), price(**decimal(16,2)**), attributes(**varchar(max**))

**For dynamic details I will provide two types:**

**First one:**

I will maintain separate table called Lov Details. I will display this on UI. If user want to use existing attributes, then user can pick up from the dropdown while ordering.

If he want a new attribute in the dropdown he should add in the attribute page.

**LOV Table**

ID(Identity column),Attribute\_value, Attribute\_description

**Second One:**

I will provide two text boxed with attribute value, attribute name. I will append this to the attributes column using the json format in the item details table.

# 2.For managing the product, attributes, and price I will display on UI and will provide edit and insert and delete operation.

I will use merge and will update the details in the table.

I will use openjson as below and update the data.

SELECT \* into #T FROM OPENJSON (@JSON)

WITH (

[item] int,

[item\_name] varchar(20),

[price] decimal(16,2),

[attributes] varchar(max) );

I will merge item details with #T and will populate the data.

To maintain the history, I will create the temporal table for item details as item details history which will log all the history information.

To view all the orders items reports with pagination

I will write a view to display all the details using the off set and fetch to get the pagination details