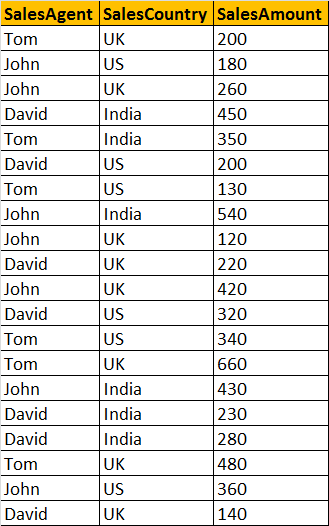
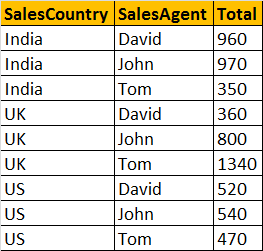
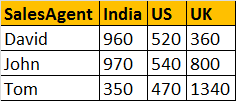
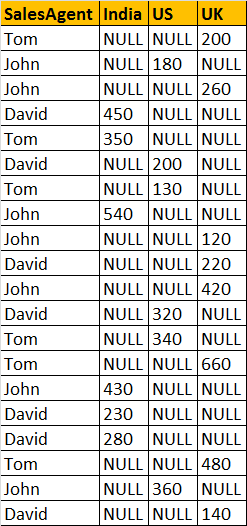
**Suggested SQL Server Videos:**  
[Part 11 - Group By](http://csharp-video-tutorials.blogspot.com/2012/08/group-by-part-11.html)  
[Part 48 - Derived table and CTE in sql server](http://csharp-video-tutorials.blogspot.com/2012/09/derived-table-and-cte-in-sql-server.html)  
  
One of my youtube channel subscribers, has asked me to make a video on **PIVOT**operator. So here we are with another sql server video.  
  
**Pivot is a sql server operator**that can be used to turn **unique values from one column**, into **multiple columns**in the output, there by effectively **rotating a table**.   
  
   
  
**Let's understand the power of PIVOT operator with an example**  
Create Table tblProductSales  
(  
 SalesAgent nvarchar(50),  
 SalesCountry nvarchar(50),  
 SalesAmount int   
)  
  
Insert into tblProductSales values('Tom', 'UK', 200)  
Insert into tblProductSales values('John', 'US', 180)  
Insert into tblProductSales values('John', 'UK', 260)  
Insert into tblProductSales values('David', 'India', 450)  
Insert into tblProductSales values('Tom', 'India', 350)  
Insert into tblProductSales values('David', 'US', 200)  
Insert into tblProductSales values('Tom', 'US', 130)  
Insert into tblProductSales values('John', 'India', 540)  
Insert into tblProductSales values('John', 'UK', 120)  
Insert into tblProductSales values('David', 'UK', 220)  
Insert into tblProductSales values('John', 'UK', 420)  
Insert into tblProductSales values('David', 'US', 320)  
Insert into tblProductSales values('Tom', 'US', 340)  
Insert into tblProductSales values('Tom', 'UK', 660)  
Insert into tblProductSales values('John', 'India', 430)  
Insert into tblProductSales values('David', 'India', 230)  
Insert into tblProductSales values('David', 'India', 280)  
Insert into tblProductSales values('Tom', 'UK', 480)  
Insert into tblProductSales values('John', 'US', 360)  
Insert into tblProductSales values('David', 'UK', 140)   
  
   
  
**Select \* from tblProductSales**: As you can see, we have 3 sales agents selling in 3 countries  
   
  
**Now, let's write a query which returns TOTAL SALES**, grouped by **SALESCOUNTRY**and **SALESAGENT**. The output should be as shown below.   
   
  
**A simple GROUP BY query can produce this output.**  
Select SalesCountry, SalesAgent, SUM(SalesAmount) as Total   
from tblProductSales  
group by SalesCountry, SalesAgent  
order by SalesCountry, SalesAgent  
 **At, this point, let's try to present the same data in different format** using PIVOT operator.   
   
  
**Query using PIVOT operator:**  
Select SalesAgent, India, US, UK   
from tblProductSales  
Pivot   
(  
   Sum(SalesAmount) for SalesCountry in ([India],[US],[UK])  
) as PivotTable

**This PIVOT query is converting the unique column values**(India, US, UK) in **SALESCOUNTRY** column, **into Columns**in the output, along with performing aggregations on the **SALESAMOUNT**column. The Outer query, simply, selects **SALESAGENT**column from **tblProductSales**table, along with pivoted columns from the PivotTable.  
  
**Having understood the basics of PIVOT**, let's look at another example. Let's create **tblProductsSale**, a slight variation of **tblProductSales**, that we have already created. The table, that we are creating now, has got an additional**Id** column.  
Create Table tblProductsSale  
(  
   Id int primary key,  
   SalesAgent nvarchar(50),  
   SalesCountry nvarchar(50),  
   SalesAmount int   
)  
  
Insert into tblProductsSale values(1, 'Tom', 'UK', 200)  
Insert into tblProductsSale values(2, 'John', 'US', 180)  
Insert into tblProductsSale values(3, 'John', 'UK', 260)  
Insert into tblProductsSale values(4, 'David', 'India', 450)  
Insert into tblProductsSale values(5, 'Tom', 'India', 350)  
Insert into tblProductsSale values(6, 'David', 'US', 200)  
Insert into tblProductsSale values(7, 'Tom', 'US', 130)  
Insert into tblProductsSale values(8, 'John', 'India', 540)  
Insert into tblProductsSale values(9, 'John', 'UK', 120)  
Insert into tblProductsSale values(10, 'David', 'UK', 220)  
Insert into tblProductsSale values(11, 'John', 'UK', 420)  
Insert into tblProductsSale values(12, 'David', 'US', 320)  
Insert into tblProductsSale values(13, 'Tom', 'US', 340)  
Insert into tblProductsSale values(14, 'Tom', 'UK', 660)  
Insert into tblProductsSale values(15, 'John', 'India', 430)  
Insert into tblProductsSale values(16, 'David', 'India', 230)  
Insert into tblProductsSale values(17, 'David', 'India', 280)  
Insert into tblProductsSale values(18, 'Tom', 'UK', 480)  
Insert into tblProductsSale values(19, 'John', 'US', 360)  
Insert into tblProductsSale values(20, 'David', 'UK', 140)

**Now, run the same PIVOT query**that we have already created, just by changing the name of the table to **tblProductsSale**instead of **tblProductSales**  
Select SalesAgent, India, US, UK   
from tblProductsSale  
Pivot   
(  
   Sum(SalesAmount) for SalesCountry in ([India],[US],[UK])  
)   
as PivotTable  
  
**This output is not what we have expected.**  
   
  
**This is because**of the presence of **Id**column in **tblProductsSale**, which is also considered when performing pivoting and group by. To eliminate this from the calculations, we have used derived table, which only selects, **SALESAGENT, SALESCOUNTRY**, and **SALESAMOUNT**. The rest of the query is very similar to what we have already seen.  
Select SalesAgent, India, US, UK  
from   
(  
   Select SalesAgent, SalesCountry, SalesAmount from tblProductsSale  
) as SourceTable  
Pivot  
(  
 Sum(SalesAmount) for SalesCountry in (India, US, UK)  
) as PivotTable  
  
**UNPIVOT**performs the opposite operation to **PIVOT**by rotating columns of a table-valued expression into column values.   
  
**The syntax of PIVOT operator from MSDN**  
SELECT <non-pivoted column>,  
    [first pivoted column] AS <column name>,  
    [second pivoted column] AS <column name>,  
    ...  
    [last pivoted column] AS <column name>  
FROM  
    (<SELECT query that produces the data>)   
    AS <alias for the source query>  
PIVOT  
(  
    <aggregation function>(<column being aggregated>)  
FOR   
    [<column that contains the values that will become column headers>]   
    IN ( [first pivoted column], [second pivoted column], ... [last pivoted column])  
)   
AS <alias for the pivot table>  
<optional ORDER BY clause>;