**بسمه تعالی**



برنامه نویسی بانک اطلاعاتی SQL Server

**نویسنده : محمد حسین فخرآوری**

Pivot table ها در واقع همانند سایر Table های دیگر می باشند فقط با این تفاوت که در نمایش ردیف و ستون های آنها مقداری تفاوت وجود دارد.

دستور pivot جزو دستوراتی از Sql server می باشد که ممکن است در بیشتر پروژه هایمان برای نمایش گزارشات مختلف   به استفاده از آن نیاز داشته باشیم. کاربرد دستور pivot تقریبا تبدیل نمایش خروجی یک کوئری از حالت ردیفی به ستونی می باشد . به صورت خیلی خلاصه بخواهیم نحوه کارکرد این دستور را بیان کنیم، کار آن یک چرخش 90 درجه جداول  می باشد ، یعنی جای ستون و ردیف ها را با هم جابه جا کنیم .

1) انجام گروهبندی روی یک ستون (Grouping)

2) توزیع مقادیر یک ستون در ستون های تعیین شده (Spreading)

3) اعمال تابع ستونی روی مقادیر موجود در ستون ها (Aggregation)

با یک مثال بهتر متوجه کارکرد این مثال خواهید شد. فرض کنید  در دیتابیس خود یک جدول دارید که کل فروش محصولات خود را در داخل آن جدول نگه داری میکنیم ، حال قصد داریم یک کوئری بنویسیم که نمایش دهیم که در هر ماه ، از هر محصول چه تعداد فروش داشته ایم. که خروجی آن تقریبا شکلی شبیه به جدول زیر را خواهد داشت :

  
ساختار دستور Pivot به صورت زیر می باشد :

|  |
| --- |
| SELECT <non-pivoted column>,  [first 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],...)  ) AS <alias for the pivot table>  <optional ORDER BY clause>; |

بعد از نوشتن Pivot و داخل پرانتز مقابل آن خواهیم داشت :

1- یک aggregation function

2- نوشتن کلمه کلیدی For و  تعیین نام فیلدی که قصد تبدیل آن به چنید ستون را خواهیم داشت

همانطور که درsyntax این دستور ملاحضه می کنید ،   دستور pivot حتما باید داری یک aggregation function باشد

فیلد مقابل کلمه For با توجه به مقادیری که بر میگرداند هر کدام به ستون هایی مجزا تبدیل می شوند . مثلا اگر در مقابل for اسم ستونی را داریم که محتوای آن فیلد دارای سه مقدار A,B,C می باشد ، در نتیجه خروجی دستور شاهد سه ستون با عنوان های A,B,C خواهیم بود .

ساختار دستور آن به صورت خلاصه تر به صورت زیر می باشد :

|  |
| --- |
| Select Field1,Field2,...,Field n  From  (  Table\_Source  )  Pivot ( Aggregate Function(Value\_Column)  For Pivot Column In(مقادیر مورد نظر)  ) Table Alias |

بررسی یک مثال عملی :

جدولی به اسم Invoice داریم که شامل سه فیلد به عنوان های زیر می باشد :

InvoiceNumber , InvoiceDate , InvoiceAmmount

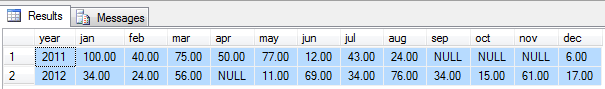
ابتدا با استفاده از کوئری زیر جدول مورد نظر را ایجاد میکنیم :

|  |
| --- |
| CREATE TABLE Invoice  (  InvoiceNumber INT PRIMARY KEY IDENTITY,  InvoiceDate DATE,  InvoiceAmount INT  ); |

و چندین رکورد را در آن درج میکنیم . قصد داریم میزان فروش محصولات را در هرماه از سال نمایش دهیم . دستور مورد نظر را به صورت زیر می نویسیم :

|  |
| --- |
| SELECT \* FROM  (SELECT YEAR(invoiceDate) AS [year], LEFT(DATENAME(MONTH, invoicedate), 3) AS [month], InvoiceAmount AS Amount FROM Invoice) AS s  PIVOT  (SUM(Amount) FOR [month] IN(jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec)) AS kk; |

که خروجی آن به صورت زیر می باشد :



و اما توضیح کدهای بالا :

در داخل دومین دستور Select که به صورت subquery نوشته شده است ، سه فیلد انتخاب کرده ایم که شامل :

|  |
| --- |
| select year(invoiceDate) as [year],left(datename(month,invoicedate),3) as [month],InvoiceAmount as Amount FROM Invoice |

 تا اینجا اگر این دستور را اجرا کنیم ، در خروجی شاهد سه ستون سال ، ماه و مقدار خواهیم بود .

فیلد سال که ثابت است و می خواهیم یکی از ستون های جدول باشد ، از Amount در قسمت Aggregate function از آن استفاده می کنیم و از month در قسمت for در دخل دستور pivot استفاده می کنیم .

مقدار ماه که دستور آن را به صورت زیر نوشته ایم :

|  |
| --- |
| left(datename(month,invoicedate),3)as [month] |

یکی از مقادیر زیر را بر می گرداند :

jan, feb, mar, apr,may, jun, jul, aug, sep, oct, nov, dec

در قسمت For باید ستونی را که می خواهیم مقدار آن را به چندین ستون تقسیم کنیم را می نویسیم ،که در اینجا ماه فیلد ماه را نوشته ایم  و بعد از آن کلمه کلیدی In  را نوشته و داخل پرانتز مقادیری را که ستون مقابل For بر میگرداند را می نویسیم ، که در این مثال ایم 12 ماه میلادی را نوشته ایم چون فیلد month برای هر ردیف یکی از این 12 ماه را بر میگرداند .

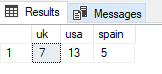
اگر در مقابل دستور For و داخل In  اسم هایی را بنویسیم که توسط فیلد مقابل For برگردانده نمی شود ، مقدار NULL را برای آن ستون خواهیم داشت. مثلا فرض کنید در مثال فوق علاوه بر اسم ماه ها ، دو فیلد اضافی دیگر هم با عنوان های esfand,bahman بنویسیم . چون این دو مقدار هیچ وقت از فیلد ماه با استفاده از دستوری که نوشتیم برگردانده نمی شوند ، پس در خروجی برای این دو فیلد شاهد مقدار NULL خواهیم بود .

اين بار به سراغ ديتابيس معروف Northwind مي رويم و مي خواهيم بدانيم در جدول Customers به ازاي کشورهاي uk، spain و usa چه تعداد کارمند داريم.

در ابتداي کار مي بايست تکليف سه مورد زير را مشخص کنيم:

1. Aggregate Column فيلد CustomerID
2. PIVOT Column فيلد Country
3. ليستي که قرار است گزارش براساس آن تهيه شود که در مثال فرضي ما، مقادير فيلد Country و شامل uk، spain و usa خواهد بود.

|  |
| --- |
| SELECT \*  FROM(SELECT Country, CustomerID FROM Customers) C  PIVOT(COUNT(CustomerID)  FOR Country IN([uk], [usa], [spain])) pTable; |



خوب، تا اين جاي کار همه چيز مطابق با خواسته ما بود اما آيا شما مي دانيد که در جدول Customers چه کشورهايي وجود دارد؟ اگر تعداد اين کشورها زياد باشد، آيا منطقي است که پس از شناسايي آن ها، ليست عريض و طويلي از عنوان کشورها را در جلو IN و در ساختار PIVOT، رديف کنيم؟ آيا اين امکان وجود ندارد که در آينده عناوين کشورهاي جديدي به جدول مان اضافه شوند؟ و …

پاسخ مناسب به حل مشکلات مطرح شده، استفاده از Dynamic T-SQL خواهد بود.

Dynamic T-SQL در واقع اسکريپت هايي است که به صورت Dynamic ايجاد مي کنيم و در همان لحظه، آن ها را اجرا مي کنيم. با استفاده از Dynamic T-SQL مي توان شرايطي پويا و متنوع در زمان اجراي يک کوئري ايجاد کرد.

در اسکريپت زير، متغيرهاي مورد نياز را تعريف و مقداردهي کرده و سپس با الحاق مناسبي با عبارات T-SQL، از طريق EXEC آن ها را اجرا مي کنيم.

|  |  |
| --- | --- |
| DECLARE @tblName VARCHAR(100),@fildName VARCHAR(100)  SET @tblName=N'Customers'  SET @fildName=N'CustomerID'  EXEC ('SELECT ' +@fildName+' FROM '+@tblName)  PRINT(@country) |  |

اکنون براي آن که بتوانيم يک Dynamic PIVOT داشته باشيم، دقيقا از اين تکنيک استفاده مي کنيم.

فرض مي کنيم که مي خواهيم بدانيم از هر کشور چه تعداد مشتري داريم. پس مي بايست ليست کشورهاي موجود را از جدول Customers استخراج کنيم.

|  |
| --- |
| Argentina,Austria,Belgium,Brazil,Canada,Denmark,Finland,France,Germany,Ireland,Italy,Mexico,Norway,Poland,Portugal,Spain,Sweden,Switzerland,UK,USA,Venezuela |

ابتدا متغير @country را تعريف مي کنيم. در خط دوم اسکريپت بالا، ابتدا مقدار @country را برابر Blank قرار مي دهيم. توجه داشته باشيد که اگر اين کار را انجام ندهيد با مشکل روبرو خواهيد شد زيرا در ابتدا، مقدار @country در هنگام تعريف، برابر با NULL خواهد بود و همواره جمع يک رشته با NULL برابر با NULL خواهد شد!

در عبارت SELECT، تمامي کشورها را از طريق جدول Customers در متغير @country به همراه جداکننده ويرگول، ذخيره مي کنيم.

توجه داشته باشيد که استفاده از GROUP BY به منظور جلوگيري از درج تکراري عناوين کشورها است.

در خط آخر هم با توجه به اين که در انتهاي رشته ي @country يک علامت ويرگول اضافي داريم، آن را حذف مي کنيم.

ليست تمامي کشورها در متغير @country ذخيره شده و مي بايست آن را به عنوان ليست مورد جستجو در جلو عبارت IN در ساختار PIVOT قرار دهيم.

|  |
| --- |
| DECLARE @country VARCHAR(MAX);  SET @country='';  SELECT @country=@country+Country+',' FROM Customers GROUP BY Country;  SET @country=LEFT(@country, LEN(@country)-1);  EXEC('SELECT \* FROM  (SELECT Country,customerID FROM Customers)C  PIVOT  (count(customerID) FOR Country IN ('+@country+'))pTable'); |

