Pivot dinámico con SQL Server

Por **FREDDY LEANDRO ANGARITA C.  
SqlServer MVP**   
[Perfil MVP](https://mvp.support.microsoft.com/es-es/mvp/Freddy%20Leandro%20Angarita%20Castellanos-4028407)   
  
[freddy\_angarita@hotmail.com](mailto:freddy_angarita@hotmail.com)   
<http://geeks.ms/blogs/fangarita/default.aspx>

**Pivotes Dinámicos**

En muchas ocasiones tenemos información en filas y queremos convertirlas en columnas, esto se presenta especialmente en escenarios de análisis en donde tener la información en columnas es un requerimiento.

Revisemos el ejemplo que presenta [MSDN](http://msdn.microsoft.com/es-co/library/ms177410.aspx) en su documentación

Si tenemos una consulta:

* 1. USE AdventureWorks2008R2 ;
  2. GO
  3. SELECT DaysToManufacture, AVG(StandardCost) AS AverageCost
  4. FROM Production.Product
  5. GROUP BY DaysToManufacture;

La cual arroja los siguientes resultados

|  |  |
| --- | --- |
| DaysToManufacture | AverageCost |
| 0 | 5.0885 |
| 1 | 223.88 |
| 2 | 359.1082 |
| 4 | 949.4105 |

Y queremos verla en columnas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Cost\_Sorted\_By\_Production\_Days | 0 | 1 | 2 | 3 | 4 |
| AverageCost | 5.0885 | 223.88 | 359.1082 | NULL | 949.4105 |

Debemos usar PIVOT

* 1. SELECT 'AverageCost' AS
  2. Cost\_Sorted\_By\_Production\_Days, [0], [1], [2], [3], [4]
  3. FROM
  4. (SELECT DaysToManufacture, StandardCost FROM Production.Product) AS SourceTable
  5. PIVOT
  6. (AVG(StandardCost)
  7. FOR DaysToManufacture IN ([0], [1], [2], [3], [4])
  8. ) AS PivotTable;

Revisando la sintaxis de la consulta tenemos que debemos definir una a una las columnas que deseamos obtener con la consulta y que éstas correspondan al Valor que queremos poner en esa columna, es decir, para cero (0) días de manufactura debemos crear una columan con el nombre 0 para que se asocie el valor que se tiene para esa columna, y se observa también que se debe incluir una función de agregación para crear un contexto para cada celda en el momento de poner el valor en la columna, esto es clave para poder generar el pivote.

Si se ha trabajado anteriormente con tablas dinámicas se verá que el comportamiento es similar tan solo debemos escribir nosotros mismos la consulta

EN SQL 2000

Acá tenemos una idea de cómo se podría solucionar en SQL 2000, algunas de las funciones que se presentan en ésta solución pronto será descontinuadas en las futuras versiones de SQL

* 1. use AdventureWorks2008R2;
  2. go
  3. SELECT
  4. [No. Sales per year]=CASE WHEN row IS NULL THEN 'Sum'
  5. ELSE CONVERT(VARCHAR(80),[row]) END ,
  6. [1996] =SUM( CASE col WHEN '2005' THEN data ELSE 0 END ),
  7. [1997] =SUM( CASE col WHEN '2006' THEN data ELSE 0 END ),
  8. [1998] =SUM( CASE col WHEN '2007' THEN data ELSE 0 END ),
  9. [Total]= SUM( data )
  10. FROM
  11. (SELECT [row]=P.firstname+' '+ P.lastname,
  12. [col]=YEAR(SOH.OrderDate),
  13. [data]=COUNT(\*)
  14. FROM Sales.SalesPerson SP INNER JOIN Sales.SalesOrderHeader SOH ON (SP.BusinessEntityID=SOH.SalesPersonID)
  15. inner join HumanResources.Employee E ON SP.BusinessEntityID = E.BusinessEntityID
  16. inner join Person.Person P ON E.BusinessEntityID = P.BusinessEntityID
  17. GROUP BY P.firstname+' '+ P.lastname, YEAR(SOH.OrderDate)
  18. )f
  19. GROUP BY row WITH ROLLUP
  20. ORDER BY GROUPING(row),total DESC

IDEAS

Siempre que aprendemos algo se nos ocurren más ideas, tales como: ¿Qué pasa si se tienen días de manufactura diferentes? ¿Qué pasa si quiero una columna para cada uno de esos días, que pasa si lo hacemos por producto? ¿Por país? qué pasa si cada uno de ellos tiene un comportamiento diferente?

Muchas de esas preguntas se responden realizando un pivote con columnas dinámicas, es decir, al contrario de la sintaxis normal en donde debemos definir cada columna, que se interprete y de obtengan todas las columnas que necesito sin necesidad de definirlas una a una

CONSULTA DINÁMICA

* 1. use AdventureWorks2008R2;
  2. go
  3. declare @columnas varchar(max)
  4. set @columnas = ''
  5. select @columnas = coalesce(@columnas + '[' + cast(DaysToManufacture as varchar(12)) + '],', '')
  6. FROM (select distinct DaysToManufacture from Production.Product) as DTM
  7. set @columnas = left(@columnas,LEN(@columnas)-1)
  8. DECLARE @SQLString nvarchar(500);
  9. set @SQLString = N'
  10. SELECT \*
  11. FROM
  12. (SELECT DaysToManufacture, StandardCost
  13. FROM Production.Product) AS SourceTable
  14. PIVOT
  15. (
  16. AVG(StandardCost)
  17. FOR DaysToManufacture IN (' + @columnas + ')
  18. ) AS PivotTable;'

EXECUTE sp\_executesql @SQLString

Pero esta solución tiene el inconveniente de tener que lidiar con consultas Ad-Hoc (Dinámicas) que asocian posibles problemas de seguridad y rendimiento a la solución, esta solución así como la demás que se puedan implementar debe considerar dónde se van a usar, posiblemente sea más eficiente realizar el pivote en reporting services (Matriz) que en SQL, esto debe evaluarse para cada y depende de lo que se quiera realizar

**OPCION CTE - Nombres de columna dinámicos**

Para 2005 o superior existe una alternativa para facilitar, mas no soluciona del todo el problema y es crear una consulta que asigna valores (números a las columnas) facilitando la construcción final de la consulta. En este caso queremos pivotear por el nombre del mes, pero supongamos que el rango que damos en la primera consulta cambia, entonces los nombre de las columnas (Nombre del mes) no podrían ser usados.

Por ejemplo si enviamos como parámetro el primero de agosto,  hasta el 31 de diciembre o si enviamos el 1 de enero al 30 de junio, tendríamos que cambiar constantemente la consulta o presentar todos los meses, para solucionar esto tenemos el siguiente ejemplo

* 1. use AdventureWorksDW2008R2;
  2. go
  3. ;With monthyears as (
  4. SELECT DISTINCT SpanishMonthName as CalendarMonthYearName, MonthNumberOfYear as MonthOfYear,CalendarYear
  5. FROM dbo.DimDate
  6. WHERE DateKey > @FechaInicio AND DateKey < @FechaFinal
  7. ),
  8. CalMonthYears AS
  9. (
  10. SELECT CalendarMonthYearName AS 'CalendarMonthYearName',
  11. MonthOfYear,
  12. CalendarYear,
  13. ROW\_NUMBER() OVER (ORDER BY CalendarYear DESC) AS 'YearMonthNum'
  14. FROM monthyears
  15. )
  16. SELECT ProductKey,
  17. [1],[2],[3],[4],[5],[6]
  18. FROM
  19. (
  20. -- your query to get your data, pivot this data
  21. SELECT YearMonthNum,s.ProductKey,
  22. SUM(OrderQuantity) AS 'Quantity'
  23. FROM FactResellerSales s
  24. INNER JOIN dbo.DimDate d ON s.OrderDateKey = d.DateKey
  25. INNER JOIN CalMonthYears cmy ON d.SpanishMonthName = cmy.CalendarMonthYearName
  26. GROUP BY YearMonthNum,s.ProductKey
  27. ) AS SourceTable
  28. PIVOT
  29. (
  30. SUM(Quantity)
  31. FOR YearMonthNum IN (
  32. [1],[2],[3],[4],[5],[6]
  33. )
  34. ) AS PivotTable

Espero sea de ayuda,

**FREDY LEANDRO ANGARITA CASTELLANOS**