Usando PIVOT e UNPIVOT

Você pode usar os operadores relacionais PIVOT e UNPIVOT para alterar uma expressão com valor de tabela em outra tabela. PIVOT gira uma expressão com valor de tabela ao transformar os valores exclusivos de uma coluna na expressão em colunas múltiplas na saída, além de executar agregações onde forem necessárias em quaisquer valores de coluna remanescentes que sejam desejados na saída final. UNPIVOT executa a operação oposta a PIVOT, girando colunas de uma expressão com valor de tabela em valores de coluna.

Observação
Quando PIVOT e UNPIVOT são usados em bancos de dados atualizados para o SQL Server 2005 ou posterior, o nível de compatibilidade do banco de dados deve ser definido como 90 ou superior. Para obter mais informações sobre como definir o nível de compatibilidade do banco de dados, consulte sp_dbcmptlevel (Transact-SQL).

A sintaxe fornecida para PIVOT é simples e mais legível do que a sintaxe que, caso contrário, poderia ser especificada em uma série complexa de instruções SELECT...CASE. Para uma descrição completa da sintaxe para PIVOT, consulte FROM (Transact-SQL).

A sintaxe anotada para PIVOT está abaixo.

```
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>;
```

Exemplo de PIVOT básico

O seguinte exemplo de código produz uma tabela de duas colunas que tem quatro linhas.

```
USE AdventureWorks2008R2 ;
GO
SELECT DaysToManufacture, AVG(StandardCost) AS AverageCost
FROM Production.Product
GROUP BY DaysToManufacture;
```

Aqui está o conjunto de resultados.

DaysToManufacture	AverageCost
0	5.0885
1	223.88
2	359.1082
4	949.4105

Nenhum produto está definido com três `DaysToManufacture`.

O código a seguir exibe o mesmo resultado, dinamizado de forma que os valores `DaysToManufacture` tornem-se títulos de coluna. Uma coluna é criada para três dias `[3]`, embora os resultados sejam `NULL`.

```
-- Pivot table with one row and five columns
SELECT 'AverageCost' AS Cost_Sorted_By_Production_Days,
[0], [1], [2], [3], [4]
FROM
(SELECT DaysToManufacture, StandardCost
    FROM Production.Product) AS SourceTable
PIVOT
(
AVG(StandardCost)
FOR DaysToManufacture IN ([0], [1], [2], [3], [4])
) AS PivotTable;
```

Aqui está o conjunto de resultados.

Cost_Sorted_By_Production_Days	0	1	2	3	4
AverageCost	5.0885	223.88	359.1082	NULL	949.4105

Exemplo de PIVOT complexo

Um cenário comum em que `PIVOT` pode ser útil ocorre quando você deseja gerar relatórios de tabulação cruzada para resumir dados. Por exemplo, suponha que você deseje consultar a tabela `PurchaseOrderHeader` no banco de dados de

exemplo `AdventureWorks2008R2` para determinar o número de ordens de compra colocadas por alguns funcionários. A consulta a seguir fornece esse relatório, ordenado por fornecedor.

```
USE AdventureWorks2008R2;
GO
SELECT VendorID, [250] AS Emp1, [251] AS Emp2, [256] AS Emp3, [257] AS Emp4, [260] AS Emp5
FROM
(SELECT PurchaseOrderID, EmployeeID, VendorID
FROM Purchasing.PurchaseOrderHeader) p
PIVOT
(
COUNT (PurchaseOrderID)
FOR EmployeeID IN
( [250], [251], [256], [257], [260] )
) AS pvt
ORDER BY pvt.VendorID;
```

Veja um conjunto parcial de resultados.

VendorID	Emp1	Emp2	Emp3	Emp4	Emp5
1492	2	5	4	4	4
1494	2	5	4	5	4
1496	2	4	4	5	5
1498	2	5	4	4	4
1500	3	4	4	5	4

Os resultados retornados por essa instrução subselecionar são dinamizados na coluna `EmployeeID`.

```
SELECT PurchaseOrderID, EmployeeID, VendorID
FROM PurchaseOrderHeader;
```

Isso significa que os valores exclusivos retornados pela coluna `EmployeeID` tornam-se campos no conjunto de resultados final. Portanto, há uma coluna para cada número de `EmployeeID` especificado na cláusula pivot: neste caso, os funcionários `164`, `198`, `223`, `231` e `233`. A coluna `PurchaseOrderID` serve como a coluna de valor, contra a qual as colunas retornadas na saída final, que são chamadas de colunas de agrupamento, são agrupadas. Neste caso, as colunas de agrupamento são agregadas pela função `COUNT`. Observe que surge uma mensagem de aviso que indica que nenhum valor nulo que apareça na coluna `PurchaseOrderID` foi levado em conta ao computar a `COUNT` para cada funcionário.

Importante

Quando as funções de agregação são usadas com PIVOT, a presença de qualquer valor nulo na coluna de valor não é considerada ao computar uma agregação.

UNPIVOT executa praticamente a operação inversa de PIVOT, girando colunas em linhas. Suponha que a tabela produzida no exemplo anterior seja armazenada no banco de dados como `pvt` e que você deseje girar os identificadores de coluna `Emp1`, `Emp2`, `Emp3`, `Emp4` e `Emp5` em valores de linhas que correspondam a um fornecedor específico. Isso significa que você deve identificar duas colunas adicionais. A coluna que conterá os valores de coluna que você está girando (`Emp1`, `Emp2`,...)

será chamada de `Employee`, e a coluna que conterá os valores que atualmente residem nas colunas que estão sendo giradas será chamada de `Orders`. Essas colunas correspondem, respectivamente, a *pivot_column* e *value_column* na definição Transact-SQL. Aqui está a consulta.

```
--Create the table and insert values as portrayed in the previous example.
CREATE TABLE pvt (VendorID int, Emp1 int, Emp2 int,
    Emp3 int, Emp4 int, Emp5 int);
GO
INSERT INTO pvt VALUES (1,4,3,5,4,4);
INSERT INTO pvt VALUES (2,4,1,5,5,5);
INSERT INTO pvt VALUES (3,4,3,5,4,4);
INSERT INTO pvt VALUES (4,4,2,5,5,4);
INSERT INTO pvt VALUES (5,5,1,5,5,5);
GO
--Unpivot the table.
SELECT VendorID, Employee, Orders
FROM
   (SELECT VendorID, Emp1, Emp2, Emp3, Emp4, Emp5
   FROM pvt) p
UNPIVOT
   (Orders FOR Employee IN
      (Emp1, Emp2, Emp3, Emp4, Emp5)
)AS unpvt;
GO
```

Veja um conjunto parcial de resultados.

VendorID Employee Orders

---------- ---------- ------

1 Emp1 4

1 Emp2 3

1 Emp3 5

1 Emp4 4

1 Emp5 4

2 Emp1 4

2 Emp2 1

2 Emp3 5

2 Emp4 5

2 Emp5 5

...

Observe que UNPIVOT não é o contrário exato de PIVOT. PIVOT executa uma agregação e, portanto, mescla possíveis linhas múltiplas em uma única linha na saída. UNPIVOT não reproduz o resultado da expressão com valor de tabela original porque as linhas foram mescladas. Além disso, os valores nulos na entrada de UNPIVOT desaparecem na saída, enquanto pode ter havido valores nulos originais na entrada antes da operação PIVOT.

A exibição Sales.vSalesPersonSalesByFiscalYears no banco de dados de exemplo AdventureWorks2008R2 usa PIVOT para retornar as vendas totais para cada vendedor em cada ano fiscal. Para gerar um script da exibição no SQL Server Management Studio, no **Pesquisador de Objetos**, localize a exibição na pasta **Exibições** do banco de dados AdventureWorks2008R2. Clique com o botão direito do mouse no nome da exibição e, depois, selecione **Script de Exibição como**.

Consulte também

Referência

FROM (Transact-SQL)
CASE (Transact-SQL)

Contribuições da comunidade