**UNPIVOT operator in T-SQL🡪**

PIVOT operator turns ROWS into COLUMNS, where as UNPIVOT turns COLUMNS into ROWS.

|  |  |  |
| --- | --- | --- |
| David 960 520 360  John 970 540 800 | select SalesAgent, Country, product  from tblProductSales\_unpiot\_test  unpivot  (  product  for Country in(India, UK, US)  )as example | David India 960  David UK 360  David US 520  John India 970  John UK 800  John US 540 |
| with value  as  (  select Row\_Number() over(order by SalesAgent) as ID, SalesAgent, India, US, UK  from tblProductSales\_unpiot\_test  )  select ID, SalesAgent, SalesCountry, ammount  from value  unpivot  (  ammount  for SalesCountry in(India, US, UK)  )as pivtest |  | 1 David India 960  1 David US 520  1 David UK 360  2 John India 970  2 John US 540  2 John UK 800 |

**Pivot operator in T-SQL**🡪

|  |  |  |
| --- | --- | --- |
| Tom UK 200  John US 180  John UK 260  David India 450  Tom India 350  David US 200  Tom US 130  John India 540  John UK 120  David UK 220  John UK 420  David US 320  Tom US 340  Tom UK 660  John India 430  David India 230  David India 280  Tom UK 480  John US 360  David UK 140 | select SalesAgent, SalesCountry,Sum(SalesAmount)  from tblProductSales\_pivot\_test  group by SalesCountry, SalesAgent  order by SalesCountry, SalesAgent | Select SalesAgent, India, US, UK  From tblProductSales\_pivot\_test  PIVOT  (  Sum(SalesAmount)  For SalesCountry in(India, US, UK)  ) as example |
| David India 960  John India 970  Tom India 350  David UK 360  John UK 800  Tom UK 1340  David US 520  John US 540  Tom US 470 | David 960 520 360  John 970 540 800  Tom 350 470 1340 |
| 1 Tom UK 200  2 John US 180  3 John UK 260  4 David India 450  5 Tom India 350  6 David US 200  7 Tom US 130  8 John India 540  9 John UK 120  10 David UK 220  11 John UK 420  12 David US 320  13 Tom US 340  14 Tom UK 660  15 John India 430  16 David India 230  17 David India 280  18 Tom UK 480  19 John US 360  20 David UK 140 | Select SalesAgent, India, US, UK  From tblProductSales\_pivot\_test  PIVOT  (  Sum(SalesAmount)  For SalesCountry in(India, US< UK)  ) as example | David 450 NULL NULL  David NULL 200 NULL  David NULL NULL 220  David NULL 320 NULL  David 230 NULL NULL  David 280 NULL NULL  David NULL NULL 140  John NULL 180 NULL  John NULL NULL 260  John 540 NULL NULL  John NULL NULL 120  John NULL NULL 420  John 430 NULL NULL  John NULL 360 NULL  Tom NULL NULL 200  Tom 350 NULL NULL  Tom NULL 130 NULL  Tom NULL 340 NULL  Tom NULL NULL 660  Tom NULL NULL 480 |
| USING CTE🡪  with ash  as  (  select SalesAgent, SalesCountry, SalesAmount  from tblProductSales\_pivot\_test  )  select SalesAgent, India, US, UK  from ash  pivot  (  sum(SalesAmount)  for SalesCountry in ([India],[US],[UK])  )as pivoteaxp | USING DERIVED TABLE🡪  Select SalesAgent, India, US, UK  from  (  Select SalesAgent, SalesCountry, SalesAmount from tblProductSales\_pivot\_test  ) as SourceTable  Pivot  (  Sum(SalesAmount) for SalesCountry in (India, US, UK)  ) as PivotTable  🡪🡪🡪🡪🡪🡪 | David 960 520 360  John 970 540 800  Tom 350 470 1340 |

**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>;

\*\*\*\*\*If you have more columns in SOURCE TABLE THAN THE TABLE YOU WANT, then you have to use TABLE EXPRESSIONS…………………

**A PIVOTED table can’t be returned by using UNPIVOT clause🡪**

|  |  |  |  |
| --- | --- | --- | --- |
| David India 960  David US 520  John India 970  John US 540  Pivoted form🡪  David 960 520  John 970 540 | select SalesAgent, India, US  from tblProductSales\_pivot\_test\_2  pivot  (  sum(SalesAmount)  for country in(India, US)  )as pibo  **🡨🡨Both returns SAME🡪🡪**  **David India 960**  **David US 520**  **John India 970**  **John US 540** | | select SalesAgent, Country, SalesAmount  from  (  select SalesAgent, India, US  from tblProductSales\_pivot\_test\_2  pivot  (  sum(SalesAmount)  for country in(India, US)  )as pibo  ) p  unpivot  (  SalesAmount  for country in (India, US)  )as big |
| **Updating table to get DUPLICATE VALUES** | | David India 960  David US 520  John India 970  John US 540  David India 150 | |

|  |  |  |
| --- | --- | --- |
| select SalesAgent, India, US  from tblProductSales\_pivot\_test\_2  pivot  (  sum(SalesAmount)  for country in(India, US)  )as pibo  **SalesA India US**  David 1110 520  John 970 540 | select SalesAgent, City, Total  from  (select SalesAgent, India, US  from tblProductSales\_pivot\_test\_2  pivot  (  sum(SalesAmount)  for Country in (India, US)  ) as pic) p  unpivot  (  Total  for City in(India, US)  )as vola | **SalesA City Total**  David India 1110  David US 520  John India 970  John US 540  \*\*we are not getting exactly 5 rows a before\*\* |

**CHOOSE FUNCTION in T-SQL🡪**

Returns the item at the specified index from a list of values in SQL Server.

**CHOOSE ( index, val\_1, val\_2 [, val\_n ] )**

|  |  |  |
| --- | --- | --- |
| SELECT Name, DateOfBirth,  case DATEPART(MM, DateOfBirth)  when 1 then 'Jan'  when 2 then 'feb'  when 3 then 'mar'  when 4 then 'apr'  when 5 then 'may'  when 6 then 'june'  when 7 then 'jul'  when 8 then 'aug'  when 9 then 'sept'  when 10 then 'oct'  when 11 then 'nov'  when 12 then 'dec'  end  from Employees\_CHOOSE\_FUN\_TEST | select Name, DateOfBirth, CHOOSE(datepart(MM, DateOfBirth), 'jan', 'feb', 'mar', 'apr', 'may', 'june', 'july', 'aug', 'sept', 'oct', 'nov', 'dec') as month  from Employees\_CHOOSE\_FUN\_TEST | 1 Mark 1980-01-11  2 John 1981-12-12  3 Amy 1979-11-21  4 Ben 1978-05-14  5 Sara 1970-03-17  6 David 1978-04-05  Mark 1980-01-11 jan  John 1981-12-12 dec  Amy 1979-11-21 nov  Ben 1978-05-14 may  Sara 1970-03-17 mar  David 1978-04-05 apr |

**IIF functions in T-SQL🡪**

Returns one of two values, depending on whether the Boolean expression evaluates to true or false in SQL Server.

**Ex:**

|  |  |
| --- | --- |
| DECLARE @tab int  set @tab=1  select IIF(@tab=2, 'Male', 'female') | * + **female** |

**Achieving same result using CASE, CHOOSE, IIF functions;**

|  |  |  |
| --- | --- | --- |
| SELECT Name, GenderId, **IIF**(GenderId=1, 'Male', 'female')  from Employees\_IIF\_FUN\_TEST | select Name, GenderId,  **case** when GenderId=1  then 'Male'  else 'female'  end  from Employees\_IIF\_FUN\_TEST | select Name, GenderId, **CHOOSE**(GenderId, 'Male', 'female')  from Employees\_IIF\_FUN\_TEST |
| 1 Mark 1  2 John 1  3 Amy 2  4 Ben 1  5 Sara 2  6 David 1 | Mark 1 Male  John 1 Male  Amy 2 female  Ben 1 Male  Sara 2 female  David 1 Male | |

* Introduced in SQL Server 2012
* Returns one of two the values, depending on whether the Boolean expression evaluates to true or false
* IIF is a shorthand way for writing a CASE expression

**Syntax :**IIF ( boolean\_expression, true\_value, false\_value )