Funções de Classificação

As Funções de Classificação retornam um valor de classificação para cada linha em uma partição. Dependendo da função usada, algumas linhas podem receber o mesmo valor que outras. As funções de classificação são **não determinísticas.**

As Quatro Funções de Classificação que o Transact-SQL fornece são :

- **Rank ;**
- **Ntile ;**
- **Row_Number ;**
- **Dense_Rank .**

Rank

A função de `RANK` retorna a classificação de cada linha na partição de um conjunto de resultados. A classificação de uma linha é um mais o número de classificações que vêm antes da linha em questão.

`ROW_NUMBER` e `RANK` são similares. `ROW_NUMBER` numera todas as linhas em sequência (por exemplo 1, 2, 3, 4, 5). `RANK` fornece o mesmo valor numérico para empates (por exemplo 1, 2, 2, 4, 5).

Se duas ou mais linhas empatarem em uma classificação, cada linha empatada receberá a mesma classificação. Por exemplo, se os dois melhores vendedores tiverem o mesmo valor `SalesYTD`, ambos serão classificados com o número um. O vendedor com o próximo `SalesYTD` maior será classificado com o número três, porque há duas linhas com classificação mais alta. Portanto, a função `RANK` nem sempre retorna inteiros consecutivos.

A ordem de classificação usada para a consulta inteira determina a ordem na qual as linhas aparecem em um conjunto de resultados.

`RANK` é não determinístico.

Exemplo

Classificando linhas dentro de uma partição : O exemplo a seguir classifica os produtos em inventário nos locais de inventário especificados de acordo com suas quantidades. O conjunto de resultados é particionado por `LocationID` e ordenado logicamente por `Quantity`. Observe que produtos 494 e 495 têm a mesma quantidade. Como eles estão vinculados, ambos são classificados como um.

```
USE AdventureWorks2012;
GO
SELECT i.ProductID, p.Name, i.LocationID, i.Quantity
    ,RANK() OVER
    (PARTITION BY i.LocationID ORDER BY i.Quantity DESC) AS Rank
FROM Production.ProductInventory AS i
INNER JOIN Production.Product AS p
    ON i.ProductID = p.ProductID
WHERE i.LocationID BETWEEN 3 AND 4
ORDER BY i.LocationID;
GO
```

Este é o conjunto de resultados apresentado :

```
ProductID   Name                    LocationID   Quantity Rank
----------- ----------------------- ------------ -------- ----
494         Paint - Silver          3            49       1
495         Paint - Blue            3            49       1
493         Paint - Red             3            41       3
496         Paint - Yellow          3            30       4
492         Paint - Black           3            17       5
495         Paint - Blue            4            35       1
496         Paint - Yellow          4            25       2
493         Paint - Red             4            24       3
492         Paint - Black           4            14       4
494         Paint - Silver          4            12       5
 (10 row(s) affected)
```

Ntile

Distribui as linhas de uma partição ordenada em um número de grupos especificado. Os grupos são numerados, iniciando em um. Para cada linha, `NTILE` retorna o número do grupo ao qual a linha pertence.

Se o número de linhas em uma partição não for divisível por *`integer_expression`*, isso causará grupos de dois tamanhos que diferem por um membro. Grupos maiores aparecem antes de grupos menores na ordem especificada pela cláusula `OVER`. Por exemplo, se o número total de linhas for 53 e o número de grupos for cinco, os três primeiros grupos terão 11 linhas e os dois grupos restantes terão 10 linhas cada. Por outro lado, se o número total de linhas for divisível pelo número de grupos, as linhas serão igualmente distribuídas entre os grupos. Por exemplo, se o número total de linhas for 50 e houver cinco grupos, cada bucket conterá 10 linhas.

`NTILE` é não determinística.

Exemplo

Dividindo linhas em grupos : O exemplo a seguir divide linhas em quatro grupos de funcionários com base nas vendas no ano até o momento. Como o número total de linhas não é divisível pelo número de grupos, os dois primeiros grupos terão quatro linhas e os grupos restantes terão três linhas cada.

```
USE AdventureWorks2012;
GO
SELECT p.FirstName, p.LastName
    ,NTILE(4) OVER(ORDER BY SalesYTD DESC) AS Quartile
    ,CONVERT(NVARCHAR(20),s.SalesYTD,1) AS SalesYTD
    , a.PostalCode
FROM Sales.SalesPerson AS s
INNER JOIN Person.Person AS p
    ON s.BusinessEntityID = p.BusinessEntityID
INNER JOIN Person.Address AS a
    ON a.AddressID = p.BusinessEntityID
WHERE TerritoryID IS NOT NULL
    AND SalesYTD <> 0;
GO
```

Este é o conjunto de resultados apresentado :

```
FirstName      LastName             Quartile  SalesYTD       PostalCode
-------------  -------------------- --------- -------------- ----------
Linda          Mitchell             1         4,251,368.55   98027
Jae            Pak                  1         4,116,871.23   98055
Michael        Blythe               1         3,763,178.18   98027
Jillian        Carson               1         3,189,418.37   98027
Ranjit         Varkey Chudukatil    2         3,121,616.32   98055
José           Saraiva              2         2,604,540.72   98055
Shu            Ito                  2         2,458,535.62   98055
Tsvi           Reiter               2         2,315,185.61   98027
Rachel         Valdez               3         1,827,066.71   98055
Tete           Mensa-Annan          3         1,576,562.20   98055
David          Campbell             3         1,573,012.94   98055
Garrett        Vargas               4         1,453,719.47   98027
Lynn           Tsoflias             4         1,421,810.92   98055
Pamela         Ansman-Wolfe         4         1,352,577.13   98027

(14 row(s) affected)
```

Row_Number

Numera a saída de um conjunto de resultados. Mais especificamente, retorna o número sequencial de uma linha em uma partição de um conjunto de resultados, começando em 1 na primeira linha de cada partição.

`ROW_NUMBER` e `RANK` são semelhantes. `ROW_NUMBER` numera todas as linhas em sequência (por exemplo 1, 2, 3, 4, 5). `RANK` fornece o mesmo valor numérico para empates (por exemplo 1, 2, 2, 4, 5).

Não há nenhuma garantia de que as linhas retornadas por uma consulta que usa `ROW_NUMBER()` serão ordenadas exatamente da mesma maneira com cada execução, a menos que as condições a seguir sejam verdadeiras.

1. Os valores da coluna particionada sejam exclusivos.
2. Os valores das `ORDER BY` colunas são exclusivos.
3. As combinações de valores da coluna de partição e colunas `ORDER BY` são exclusivas.

`ROW_NUMBER()` é não determinístico.

Exemplo

Retornando o número de linha para vendedores : O exemplo a seguir calcula um número de linha para os vendedores da Ciclos da Adventure Works com base em sua classificação de vendas no ano até a data.

```
USE AdventureWorks2012;
GO
SELECT ROW_NUMBER() OVER(ORDER BY SalesYTD DESC) AS Row,
    FirstName, LastName, ROUND(SalesYTD,2,1) AS "Sales YTD"
FROM Sales.vSalesPerson
WHERE TerritoryName IS NOT NULL AND SalesYTD <> 0;
```

Este é o conjunto de resultados apresentado :

```
Row FirstName    LastName                SalesYTD
--- -----------  ----------------------- -----------------
1   Linda        Mitchell                4251368.54
2   Jae          Pak                     4116871.22
3   Michael      Blythe                  3763178.17
4   Jillian      Carson                  3189418.36
5   Ranjit       Varkey Chudukatil       3121616.32
6   José         Saraiva                 2604540.71
7   Shu          Ito                     2458535.61
8   Tsvi         Reiter                  2315185.61
9   Rachel       Valdez                  1827066.71
10  Tete         Mensa-Annan             1576562.19
11  David        Campbell                1573012.93
12  Garrett      Vargas                  1453719.46
13  Lynn         Tsoflias                1421810.92
14  Pamela       Ansman-Wolfe            1352577.13
```

Dense_Rank

Esta função retorna a posição de cada linha dentro de uma partição do conjunto de resultados, sem nenhum intervalo nos valores de classificação. A classificação de uma linha é um mais o número de valores de classificação distintos que vêm antes da linha em questão.

Se duas ou mais linhas tiverem o mesmo valor de classificação na mesma partição, cada uma dessas linhas receberá a mesma classificação. Por exemplo, se os dois melhores vendedores tiverem o mesmo valor de SalesYTD, ambos terão um valor de classificação igual a um. O vendedor com o próximo SalesYTD mais alto terá um valor de classificação igual a dois. Isso excede o número de linhas distintas que vêm antes da linha em questão por um. Portanto, os números retornados pela função `DENSE_RANK` não têm lacunas e sempre têm valores de classificação consecutivos.

A ordem de classificação usada para a consulta inteira determina a ordem das linhas no conjunto de resultados. Isso significa que uma linha classificada com o número um não precisa ser a primeira linha da partição.

`DENSE_RANK` é não determinístico.

Exemplo

Classificando linhas dentro de uma partição : Este exemplo classifica os produtos em inventário pelos locais de inventário especificados, de acordo com suas quantidades. `DENSE_RANK` particiona o conjunto de resultados por `LocationID` e ordena logicamente o conjunto de resultados por `Quantity`. Observe que produtos 494 e 495 têm a mesma quantidade. Como ambos têm o mesmo valor de quantidade, ambos têm um valor de classificação igual a um.

```
USE AdventureWorks2012;
GO
SELECT i.ProductID, p.Name, i.LocationID, i.Quantity
    ,DENSE_RANK() OVER
    (PARTITION BY i.LocationID ORDER BY i.Quantity DESC) AS Rank
FROM Production.ProductInventory AS i
INNER JOIN Production.Product AS p
    ON i.ProductID = p.ProductID
WHERE i.LocationID BETWEEN 3 AND 4
ORDER BY i.LocationID;
GO
```

Este é o conjunto de resultados apresentados :

```
ProductID   Name                                LocationID Quantity Rank
----------- ----------------------------------- ---------- -------- -----
494         Paint - Silver                      3          49       1
495         Paint - Blue                        3          49       1
493         Paint - Red                         3          41       2
496         Paint - Yellow                      3          30       3
492         Paint - Black                       3          17       4
495         Paint - Blue                        4          35       1
496         Paint - Yellow                      4          25       2
493         Paint - Red                         4          24       3
492         Paint - Black                       4          14       4
494         Paint - Silver                      4          12       5

(10 row(s) affected)
```

Bibliografia

- https://docs.microsoft.com/pt-br/sql/t-sql/functions/rank-transact-sql?view=sql-server-ver15
- https://docs.microsoft.com/pt-br/sql/t-sql/functions/ntile-transact-sql?view=sql-server-ver15
- https://docs.microsoft.com/pt-br/sql/t-sql/functions/row-number-transact-sql?view=sql-server-ver15
- https://docs.microsoft.com/pt-br/sql/t-sql/functions/dense-rank-transact-sql?view=sql-server-ver15
- https://docs.microsoft.com/pt-br/sql/t-sql/functions/ranking-functions-transact-sql?view=sql-server-ver15