Introduction to SQL Server ROW\_NUMBER() function

The ROW\_NUMBER() is a [window function](https://www.sqlservertutorial.net/sql-server-window-functions/) that assigns a sequential integer to each row within the partition of a result set. The row number starts with 1 for the first row in each partition.

The following shows the syntax of the ROW\_NUMBER() function:

ROW\_NUMBER() OVER (

[PARTITION BY partition\_expression, ... ]

ORDER BY sort\_expression [ASC | DESC], ...

)

Code language: SQL (Structured Query Language) (sql)

Let’s examine the syntax of the ROW\_NUMBER() function in detail.

PARTITION BY

The PARTITION BY clause divides the result set into partitions (another term for groups of rows). The ROW\_NUMBER() function is applied to each partition separately and reinitialized the row number for each partition.

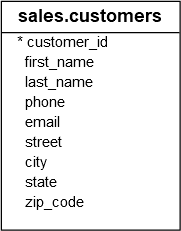
The PARTITION BY clause is optional. If you skip it, the ROW\_NUMBER() function will treat the whole result set as a single partition.

ORDER BY

The ORDER BY clause defines the logical order of the rows within each partition of the result set. The ORDER BY clause is mandatory because the ROW\_NUMBER() function is order sensitive.

SQL Server ROW\_NUMBER() examples

We’ll use the sales.customers table from the [sample database](https://www.sqlservertutorial.net/sql-server-sample-database/) to demonstrate the ROW\_NUMBER() function.



Using SQL Server ROW\_NUMBER() function over a result set example

The following statement uses the ROW\_NUMBER() to assign each customer row a sequential number:

SELECT

ROW\_NUMBER() OVER (

ORDER BY first\_name

) row\_num,

first\_name,

last\_name,

city

FROM

sales.customers;

Code language: SQL (Structured Query Language) (sql)

Here is the partial output:



In this example, we skipped the PARTITION BY clause, therefore, the ROW\_NUMBER() treated the whole result set as a single partition.

Using SQL Server ROW\_NUMBER() over partitions example

The following example uses the ROW\_NUMBER() function to assign a sequential integer to each customer. It resets the number when the city changes:

SELECT

first\_name,

last\_name,

city,

ROW\_NUMBER() OVER (

PARTITION BY city

ORDER BY first\_name

) row\_num

FROM

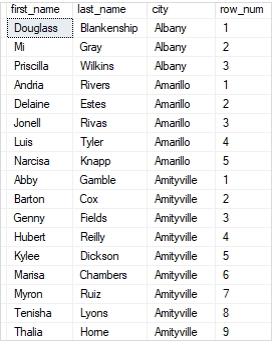
sales.customers

ORDER BY

city;

Code language: SQL (Structured Query Language) (sql)

The following picture shows the partial output:



In this example, we used the PARTITION BY clause to divide the customers into partitions by city. The row number was reinitialized when the city changed.

Using SQL Server ROW\_NUMBER() for pagination

The ROW\_NUMBER() function is useful for pagination in applications. For example, you can display a list of customers by page, where each page has 10 rows.

The following example uses the ROW\_NUMBER() to return customers from row 11 to 20, which is the second page:

WITH cte\_customers AS (

SELECT

ROW\_NUMBER() OVER(

ORDER BY

first\_name,

last\_name

) row\_num,

customer\_id,

first\_name,

last\_name

FROM

sales.customers

) SELECT

customer\_id,

first\_name,

last\_name

FROM

cte\_customers

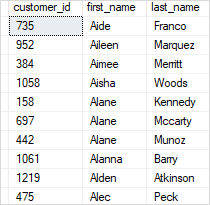
WHERE

row\_num > 20 AND

row\_num <= 30;

Code language: SQL (Structured Query Language) (sql)

The output is as follows:



In this example:

* First, the [CTE](https://www.sqlservertutorial.net/sql-server-basics/sql-server-cte/) used the ROW\_NUMBER() function to assign every row in the result set a sequential integer.
* Second, the outer query returned the rows of the second page, which have the row number between 11 to 20.