In this video we will discuss **Rank and Dense\_Rank functions in SQL Server**  
  
   
  
**Rank and Dense\_Rank functions**

* Introduced in SQL Server 2005
* Returns a rank starting at 1 based on the ordering of rows imposed by the ORDER BY clause
* ORDER BY clause is required
* PARTITION BY clause is optional
* When the data is partitioned, rank is reset to 1 when the partition changes

**Difference between Rank and Dense\_Rank functions**  
Rank function skips ranking(s) if there is a tie where as Dense\_Rank will not.   
  
**For example :**If you have 2 rows at rank 1 and you have 5 rows in total.

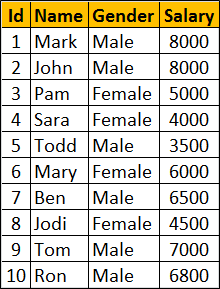
RANK() returns - 1, 1, 3, 4, 5

DENSE\_RANK returns - 1, 1, 2, 3, 4

**Syntax :**

RANK() OVER (ORDER BY Col1, Col2, ...)

DENSE\_RANK() OVER (ORDER BY Col1, Col2, ...)

**Example :** We will use the following **Employees**table for the examples in this video   
   
  
**SQl Script to create Employees table**

Create Table Employees

(

    Id int primary key,

    Name nvarchar(50),

    Gender nvarchar(10),

    Salary int

)

Go

Insert Into Employees Values (1, 'Mark', 'Male', 8000)

Insert Into Employees Values (2, 'John', 'Male', 8000)

Insert Into Employees Values (3, 'Pam', 'Female', 5000)

Insert Into Employees Values (4, 'Sara', 'Female', 4000)

Insert Into Employees Values (5, 'Todd', 'Male', 3500)

Insert Into Employees Values (6, 'Mary', 'Female', 6000)

Insert Into Employees Values (7, 'Ben', 'Male', 6500)

Insert Into Employees Values (8, 'Jodi', 'Female', 4500)

Insert Into Employees Values (9, 'Tom', 'Male', 7000)

Insert Into Employees Values (10, 'Ron', 'Male', 6800)

Go

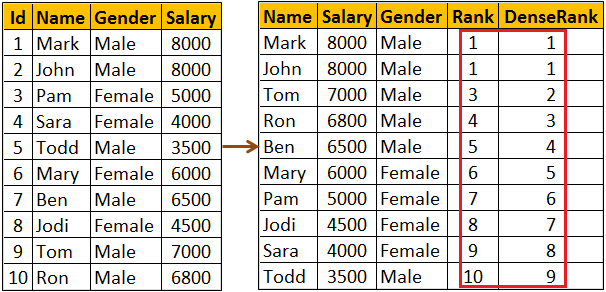
**RANK() and DENSE\_RANK() functions without PARTITION BY clause :** In this example, data is not partitioned, so RANK() function provides a consecutive numbering except when there is a tie. Rank 2 is skipped as there are 2 rows at rank 1. The third row gets rank 3.   
  
DENSE\_RANK() on the other hand will not skip ranks if there is a tie. The first 2 rows get rank 1. Third row gets rank 2. 

SELECT Name, Salary, Gender,

RANK() OVER (ORDER BY Salary DESC) AS [Rank],

DENSE\_RANK() OVER (ORDER BY Salary DESC) AS DenseRank

FROM Employees

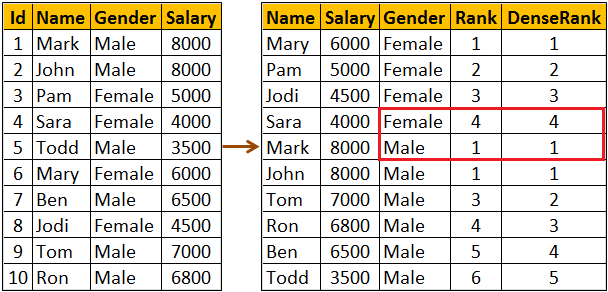
  
  
**RANK() and DENSE\_RANK() functions with PARTITION BY clause :** Notice when the partition changes from Female to Male Rank is reset to 1 

SELECT Name, Salary, Gender,

RANK() OVER (PARTITION BY Gender ORDER BY Salary DESC) AS [Rank],

DENSE\_RANK() OVER (PARTITION BY Gender ORDER BY Salary DESC)   
AS DenseRank

FROM Employees

  
  
**Use case for RANK and DENSE\_RANK functions :** Both these functions can be used to find Nth highest salary. However, which function to use depends on what you want to do when there is a tie. Let me explain with an example.   
  
**If there are 2 employees with the FIRST highest salary, there are 2 different business cases**

* If your business case is, not to produce any result for the SECOND highest salary, then use RANK function
* If your business case is to return the next Salary after the tied rows as the SECOND highest Salary, then use DENSE\_RANK function

Since we have 2 Employees with the FIRST highest salary. Rank() function will not return any rows for the SECOND highest Salary. 

WITH Result AS

(

    SELECT Salary, RANK() OVER (ORDER BY Salary DESC) AS Salary\_Rank

    FROM Employees

)

SELECT TOP 1 Salary FROM Result WHERE Salary\_Rank = 2

Though we have 2 Employees with the FIRST highest salary. Dense\_Rank() function returns, the next Salary after the tied rows as the SECOND highest Salary 

WITH Result AS

(

    SELECT Salary, DENSE\_RANK() OVER (ORDER BY Salary DESC) ASSalary\_Rank

    FROM Employees

)

SELECT TOP 1 Salary FROM Result WHERE Salary\_Rank = 2

You can also use RANK and DENSE\_RANK functions to find the Nth highest Salary among Male or Female employee groups. The following query finds the 3rd highest salary amount paid among the Female employees group 

WITH Result AS

(

    SELECT Salary, Gender,

           DENSE\_RANK() OVER (PARTITION BY Gender ORDER BY Salary DESC)

           AS Salary\_Rank

    FROM Employees

)

SELECT TOP 1 Salary FROM Result WHERE Salary\_Rank = 3

AND Gender = 'Female'