# **RANK, DENSE\_RANK, FIRST and LAST Analytic Functions**

## Setup The examples in this document require the following table.

DROP TABLE myemp;

CREATE TABLE myemp (

myempno NUMBER(4) CONSTRAINT pk\_myemp PRIMARY KEY,

ename VARCHAR2(10),

job VARCHAR2(9),

mgr NUMBER(4),

hiredate DATE,

sal NUMBER(7,2),

comm NUMBER(7,2),

deptno NUMBER(2)

);

INSERT INTO myemp VALUES (7369,'SMITH','CLERK',7902,to\_date('17-12-1980','dd-mm-yyyy'),800,NULL,20);

INSERT INTO myemp VALUES (7499,'ALLEN','SALESMAN',7698,to\_date('20-2-1981','dd-mm-yyyy'),1600,300,30);

INSERT INTO myemp VALUES (7521,'WARD','SALESMAN',7698,to\_date('22-2-1981','dd-mm-yyyy'),1250,500,30);

INSERT INTO myemp VALUES (7566,'JONES','MANAGER',7839,to\_date('2-4-1981','dd-mm-yyyy'),2975,NULL,20);

INSERT INTO myemp VALUES (7654,'MARTIN','SALESMAN',7698,to\_date('28-9-1981','dd-mm-yyyy'),1250,1400,30);

INSERT INTO myemp VALUES (7698,'BLAKE','MANAGER',7839,to\_date('1-5-1981','dd-mm-yyyy'),2850,NULL,30);

INSERT INTO myemp VALUES (7782,'CLARK','MANAGER',7839,to\_date('9-6-1981','dd-mm-yyyy'),2450,NULL,10);

INSERT INTO myemp VALUES (7788,'SCOTT','ANALYST',7566,to\_date('13-JUL-87','dd-mm-rr')-85,3000,NULL,20);

INSERT INTO myemp VALUES (7839,'KING','PRESIDENT',NULL,to\_date('17-11-1981','dd-mm-yyyy'),5000,NULL,10);

INSERT INTO myemp VALUES (7844,'TURNER','SALESMAN',7698,to\_date('8-9-1981','dd-mm-yyyy'),1500,0,30);

INSERT INTO myemp VALUES (7876,'ADAMS','CLERK',7788,to\_date('13-JUL-87', 'dd-mm-rr')-51,1100,NULL,20);

INSERT INTO myemp VALUES (7900,'JAMES','CLERK',7698,to\_date('3-12-1981','dd-mm-yyyy'),950,NULL,30);

INSERT INTO myemp VALUES (7902,'FORD','ANALYST',7566,to\_date('3-12-1981','dd-mm-yyyy'),3000,NULL,20);

INSERT INTO myemp VALUES (7934,'MILLER','CLERK',7782,to\_date('23-1-1982','dd-mm-yyyy'),1300,NULL,10);

COMMIT;

**RANK 相等的排名一样**

**The basic description for the RANK analytic function is shown below. The analytic clause is described in more detail;**

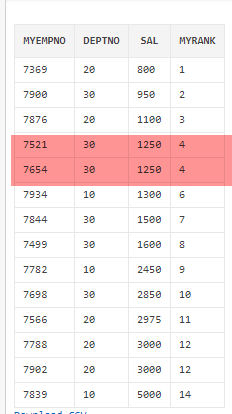
**SELECT myempno,**

**deptno,**

**sal,**

**rank() over (order by sal) as myrank**

**FROM myemp;**



**What we see here is where two employees have the same salary they are assigned the same rank. When multiple rows share the same rank the next rank in the sequence is not consecutive. This is like Olympic medaling in that if two people share the gold, there is no silver medal etc.**

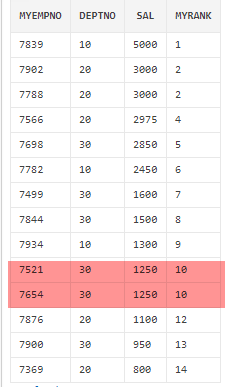
**SELECT myempno,**

**deptno,**

**sal,**

**rank() over (order by sal desc) as myrank**

**FROM myemp;**



* **Bottom two low salaried employees.**

**SELECT \***

**FROM (SELECT myempno,**

**deptno,**

**sal,**

**RANK() OVER ( ORDER BY sal) AS myrank**

**FROM myemp)**

**WHERE myrank <= 2;**



* **Top two low salaried employees.**

**SELECT \***

**FROM (SELECT myempno,**

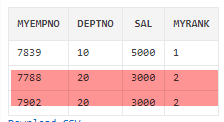
**deptno,**

**sal,**

**RANK() OVER ( ORDER BY sal desc ) AS myrank**

**FROM myemp)**

**WHERE myrank <= 2;**



RANK() OVER ([ query\_partition\_clause ] order\_by\_clause)

**Let's assume we want to assign a sequential order, or rank, to people within a department based on salary, we might use the RANK function like this.**

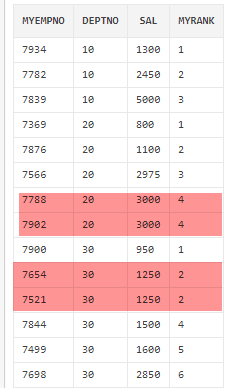
**SELECT myempno,**

**deptno,**

**sal,**

**RANK() OVER (PARTITION BY deptno ORDER BY sal) AS myrank**

**FROM myemp;**



**The fact we can rank the rows in the department means we are able to do a**[**Top-N query**](https://oracle-base.com/articles/misc/top-n-queries)**on a per-department basis. The example below assigns the rank in the inline view, then uses that rank to restrict the rows to the bottom 2 (worst paid) employees in each department.**

**SELECT \***

**FROM (SELECT myempno,**

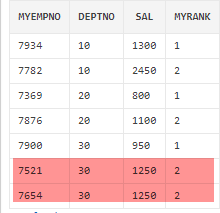
**deptno,**

**sal,**

**RANK() OVER (PARTITION BY deptno ORDER BY sal) AS myrank**

**FROM myemp)**

**WHERE myrank <= 2;**



**DENSE\_RANK 相等的排名也不同，全是连续数字**

DENSE\_RANK() OVER([ query\_partition\_clause ] order\_by\_clause)

**The DENSE\_RANK function acts like the RANK function except that it assigns consecutive ranks, so this is not like Olympic medaling**.

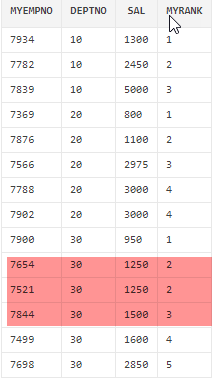
**SELECT myempno,**

**deptno,**

**sal,**

**DENSE\_RANK() OVER (PARTITION BY deptno ORDER BY sal) AS myrank**

**FROM myemp;**



**As with the RANK analytic function, we can do a**[**Top-N query**](https://oracle-base.com/articles/misc/top-n-queries)**on a per-department basis. The example below assigns the dense rank in the inline view, then uses that rank to restrict the rows to the top 2 (best paid) employees in each department.**

**SELECT \***

**FROM (SELECT myempno,**

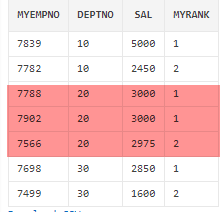
**deptno,**

**sal,**

**DENSE\_RANK() OVER (PARTITION BY deptno ORDER BY sal DESC) AS myrank**

**FROM myemp)**

**WHERE myrank <= 2;**



**FIRST and LAST**

**The FIRST and LAST functions can be used to return the first or last value from an ordered sequence. Say we want to display the salary of each employee, along with the lowest and highest within their department we may use something like.**

**SELECT myempno,**

**deptno,**

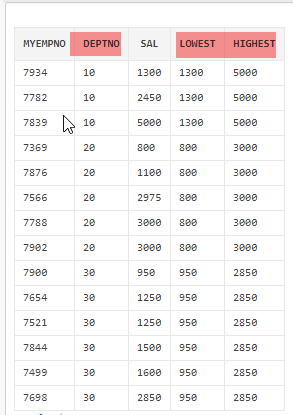
**sal,**

**MIN(sal) KEEP (DENSE\_RANK FIRST ORDER BY sal) OVER (PARTITION BY deptno) lowest,**

**MAX(sal) KEEP (DENSE\_RANK LAST ORDER BY sal) OVER (PARTITION BY deptno) highest**

**FROM myemp**

**ORDER BY deptno, sal;**



**The MIN and MAX functions are almost irrelevant here as it's FIRST, LAST and KEEP that are picking the row whose value will be used. We can demonstrate this by using MIN for both the high and low value.**

**SELECT myempno,**

**deptno,**

**sal,**

**MIN(sal) KEEP (DENSE\_RANK FIRST ORDER BY sal) OVER (PARTITION BY deptno) lowest,**

**MIN(sal) KEEP (DENSE\_RANK LAST ORDER BY sal) OVER (PARTITION BY deptno) highest**

**FROM myemp ORDER BY deptno, sal;**

**SELECT myempno,**

**deptno,**

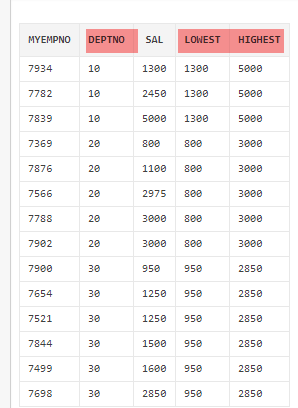
**sal,**

**MAX(sal) KEEP (DENSE\_RANK FIRST ORDER BY sal) OVER (PARTITION BY deptno) lowest,**

**MAX(sal) KEEP (DENSE\_RANK LAST ORDER BY sal) OVER (PARTITION BY deptno) highest**

**FROM myemp**

**ORDER BY deptno, sal;**



=== FROM CLASS PRACTICE EXERCISE

Find employee number, name of employee(s) who have maximum number of projects assigned.

**WITH PROJCNTRANK**

**AS**

**( select empno,count(projid) cnt,**

**rank () over (order by count(projid) desc) as myrank**

**from ap\_proemp**

**group by empno )**

**SELECT A.EMPNO, A.ENAME, B.CNT, B.MYRANK**

**FROM AP\_EMP A JOIN PROJCNTRANK B ON A.EMPNO=B.EMPNO AND**

**MYRANK<2;**