LAB1: Practical Assignment 1

Title: Topic: Analytical functions

Objective: To Learn about Rank(), Dense_rank(), Row_number(),Top N query

Theory:

- For analytic functions, you can use all of the regular group functions:
- SUM
- MAX
- MIN
- AVG
- COUNT
- Plus list of additional analytical functions that can be used only for window queries:
- LAG
- LEAD
- FIRST
- LAST
- FIRST VALUE
- LAST VALUE
- ROW_NUMBER
- DENSE_RANK

Rank(): The RANK function allows the business analyst to compute the rank of a set of values, which is done by comparing all of the values in a set.

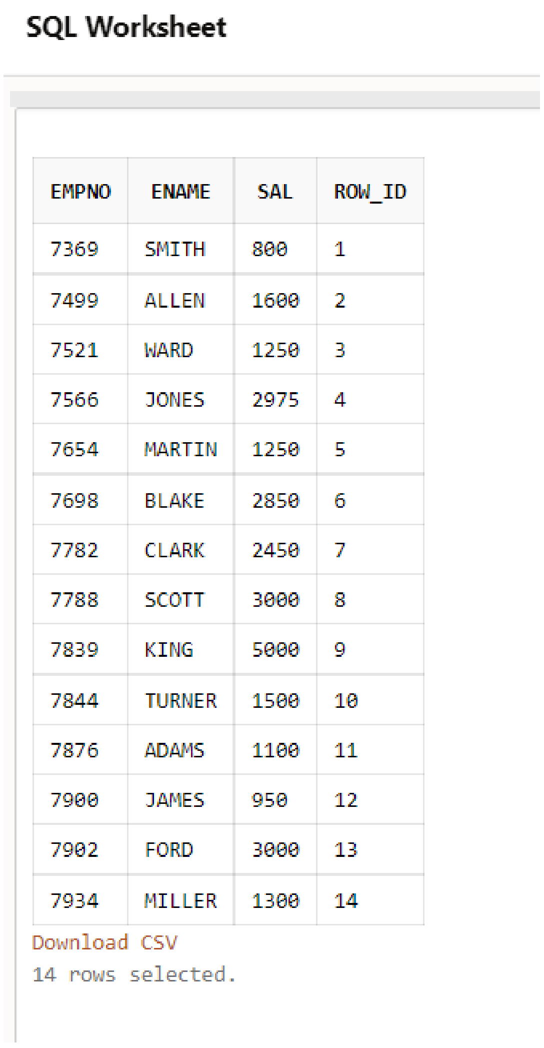
Dense_rank(): dense_rank(): The DENSE_RANK function is similar to the RANK function except for one major difference that It does not skip sequential ranking numbers.

1. Display empno, ename, sal from emp table and give numbers to each row.

Script:

select empno, ename, sal, ROW_NUMBER() over (order by empno, ename, sal) as Row_id from emp;

Output:



SQL Worksheet

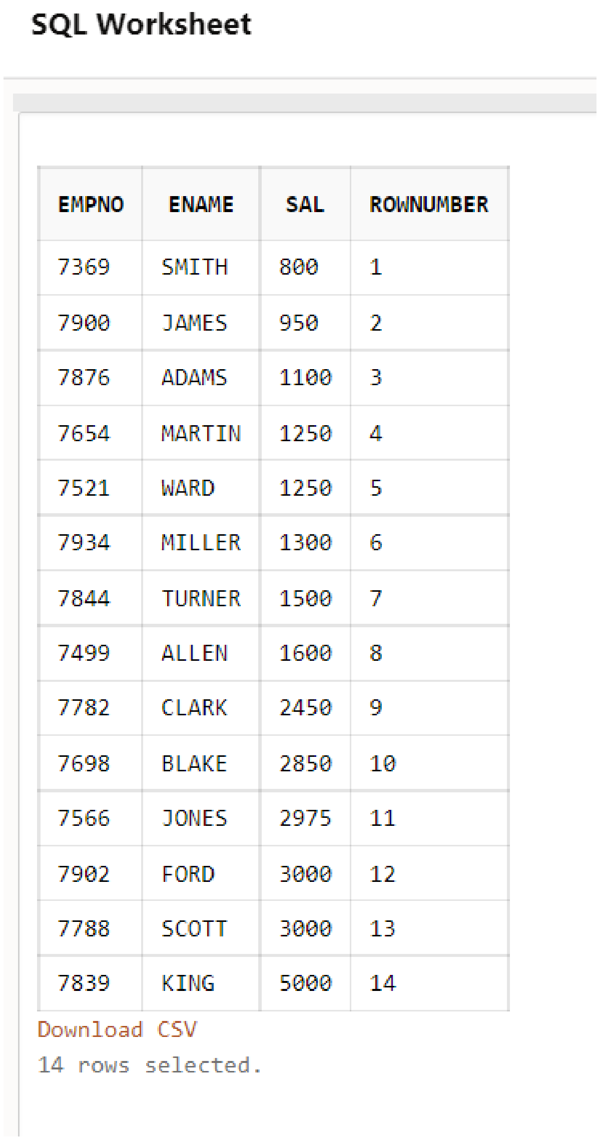
EMPNO	ENAME	SAL	ROW_ID
7369	SMITH	800	1
7499	ALLEN	1600	2
7521	WARD	1250	3
7566	JONES	2975	4
7654	MARTIN	1250	5
7698	BLAKE	2850	6
7782	CLARK	2450	7
7788	SCOTT	3000	8
7839	KING	5000	9
7844	TURNER	1500	10
7876	ADAMS	1100	11
7900	JAMES	950	12
7902	FORD	3000	13
7934	MILLER	1300	14

Download CSV
14 rows selected.

2. Display empno, ename, sal and give numbers to each row in ascending order of salary.

Script: SELECT empno, ename, sal, ROW_NUMBER() OVER (ORDER BY sal) AS RowNumber FROM scott.emp;

Output:


SQL Worksheet

EMPNO	ENAME	SAL	ROWNUMBER
7369	SMITH	800	1
7900	JAMES	950	2
7876	ADAMS	1100	3
7654	MARTIN	1250	4
7521	WARD	1250	5
7934	MILLER	1300	6
7844	TURNER	1500	7
7499	ALLEN	1600	8
7782	CLARK	2450	9
7698	BLAKE	2850	10
7566	JONES	2975	11
7902	FORD	3000	12
7788	SCOTT	3000	13
7839	KING	5000	14

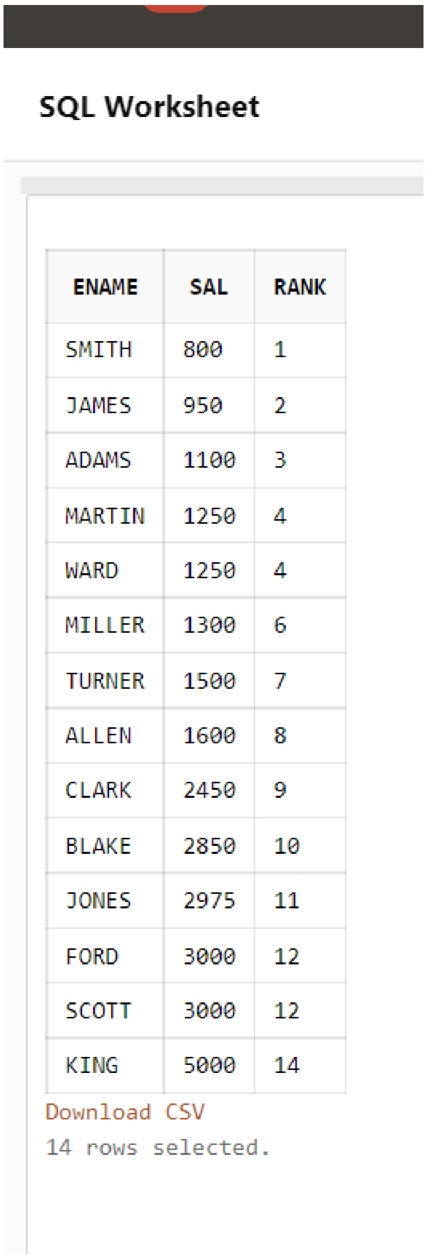
Download CSV

```
14 rows selected.
```

3. Assign the ranks to employees in ascending order of salary.

Script: **SELECT ename, sal, RANK() OVER(ORDER BY sal ASC) Rank FROM scott.emp;**

Output:



SQL Worksheet

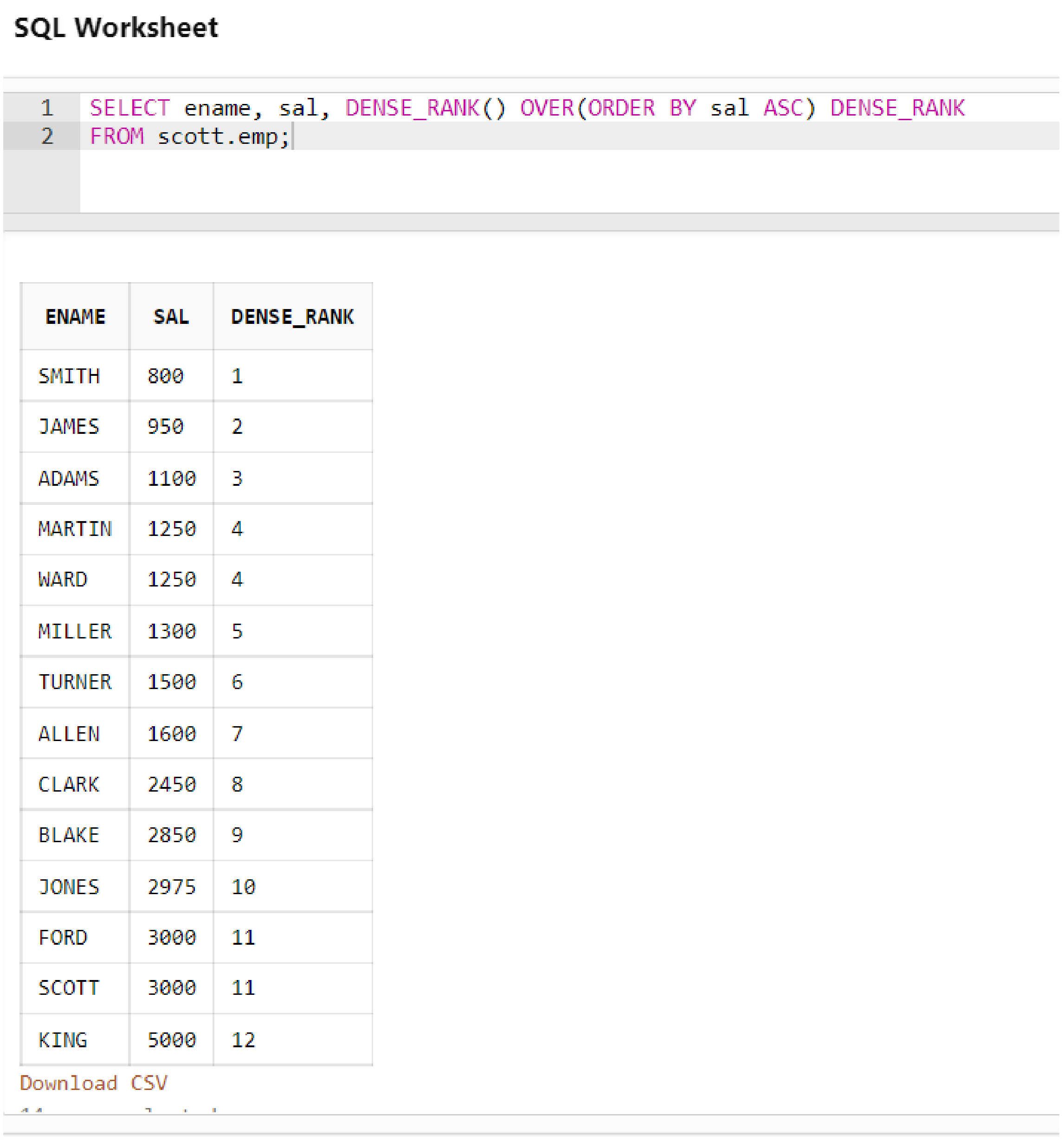
ENAME	SAL	RANK
SMITH	800	1
JAMES	950	2
ADAMS	1100	3
MARTIN	1250	4
WARD	1250	4
MILLER	1300	6
TURNER	1500	7
ALLEN	1600	8
CLARK	2450	9
BLAKE	2850	10
JONES	2975	11
FORD	3000	12
SCOTT	3000	12
KING	5000	14

```
Download CSV
14 rows selected.
```

4. Assign the ranks to employees in ascending order of salary using dense rank and point out the difference.

Script: **SELECT ename, sal, DENSE_RANK() OVER(ORDER BY sal ASC) DENSE_RANK FROM scott.emp;**

Output:



SQL Worksheet

```
SELECT ename, sal, DENSE_RANK() OVER(ORDER BY sal ASC) DENSE_RANK
FROM scott.emp;
```

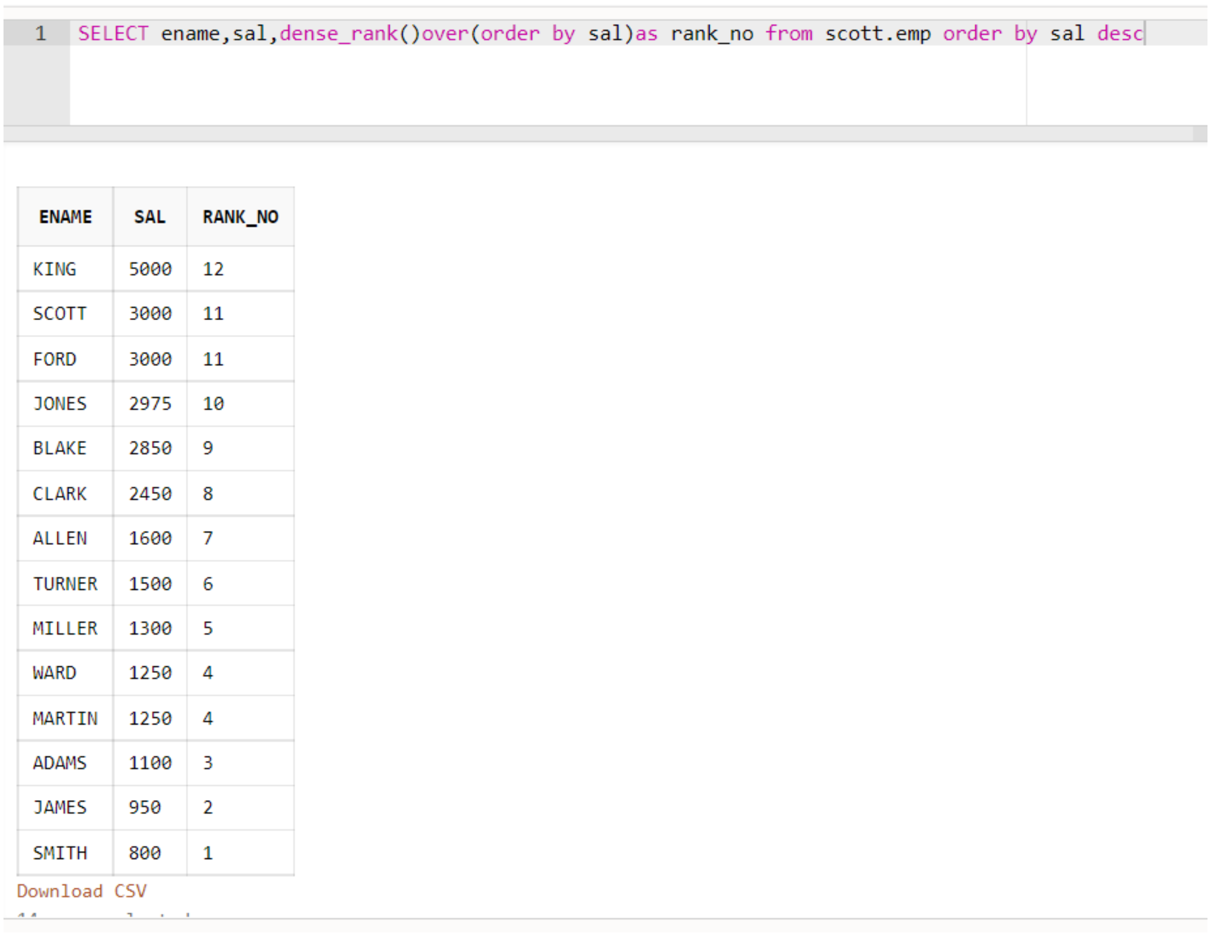
ENAME	SAL	DENSE_RANK
SMITH	800	1
JAMES	950	2
ADAMS	1100	3
MARTIN	1250	4
WARD	1250	4
MILLER	1300	5
TURNER	1500	6
ALLEN	1600	7
CLARK	2450	8
BLAKE	2850	9
JONES	2975	10
FORD	3000	11
SCOTT	3000	11
KING	5000	12

Download CSV

5. Assign the ranks to employees in ascending order of salary but display records in descending order of salary.

Script: SELECT ename,sal,dense_rank()over(order by sal)as rank_no from scott.emp order by sal desc

Output:



```sql
SELECT ename,sal,dense_rank()over(order by sal)as rank_no from scott.emp order by sal desc
```

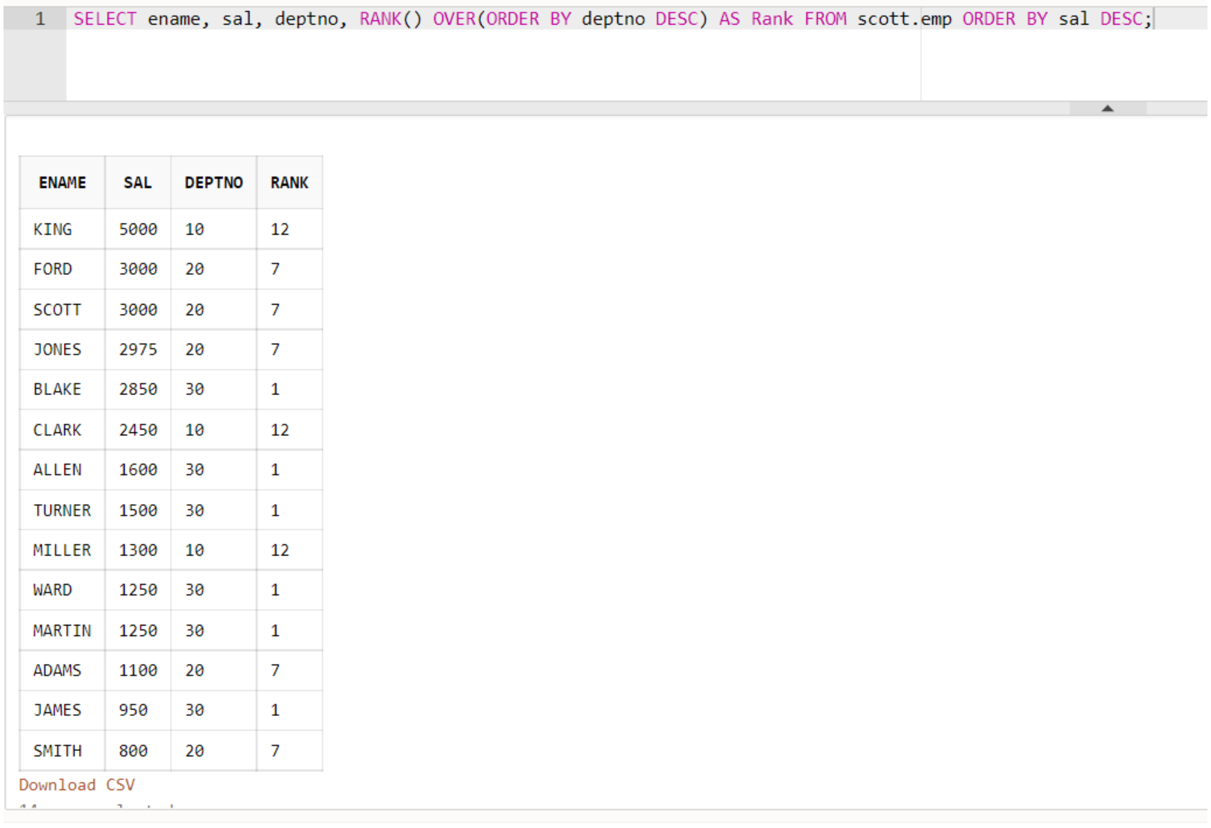
ENAME	SAL	RANK_NO
KING	5000	12
SCOTT	3000	11
FORD	3000	11
JONES	2975	10
BLAKE	2850	9
CLARK	2450	8
ALLEN	1600	7
TURNER	1500	6
MILLER	1300	5
WARD	1250	4
MARTIN	1250	4
ADAMS	1100	3
JAMES	950	2
SMITH	800	1

Download CSV

6. Assign the rank to emp table rows in the ascending order of department and salary.

Script: SELECT ename, sal, deptno, RANK() OVER(ORDER BY deptno DESC) AS Rank FROM scott.emp ORDER BY sal DESC;

Output:



```
SELECT ename, sal, deptno, RANK() OVER(ORDER BY deptno DESC) AS Rank FROM scott.emp ORDER BY sal DESC;
```

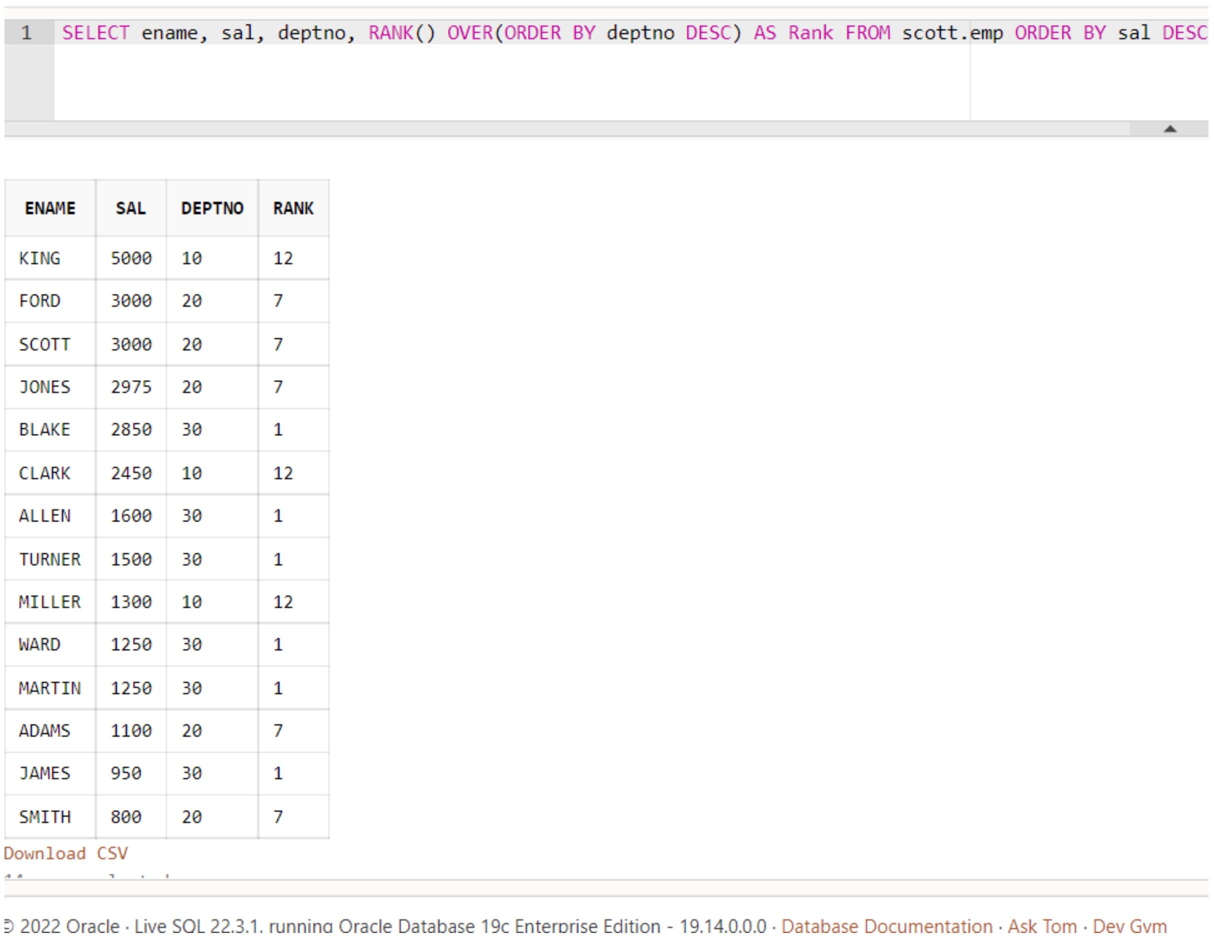
ENAME	SAL	DEPTNO	RANK
KING	5000	10	12
FORD	3000	20	7
SCOTT	3000	20	7
JONES	2975	20	7
BLAKE	2850	30	1
CLARK	2450	10	12
ALLEN	1600	30	1
TURNER	1500	30	1
MILLER	1300	10	12
WARD	1250	30	1
MARTIN	1250	30	1
ADAMS	1100	20	7
JAMES	950	30	1
SMITH	800	20	7

Download CSV

7. Assign the rank to emp table rows in the ascending order of department and descending order of salary.

Script: SELECT ename, sal, deptno, RANK() OVER(ORDER BY deptno DESC) AS Rank FROM scott.emp ORDER BY sal DESC;

Output:



```
SELECT ename, sal, deptno, RANK() OVER(ORDER BY deptno DESC) AS Rank FROM scott.emp ORDER BY sal DESC
```

ENAME	SAL	DEPTNO	RANK
KING	5000	10	12
FORD	3000	20	7
SCOTT	3000	20	7
JONES	2975	20	7
BLAKE	2850	30	1
CLARK	2450	10	12
ALLEN	1600	30	1
TURNER	1500	30	1
MILLER	1300	10	12
WARD	1250	30	1
MARTIN	1250	30	1
ADAMS	1100	20	7
JAMES	950	30	1
SMITH	800	20	7

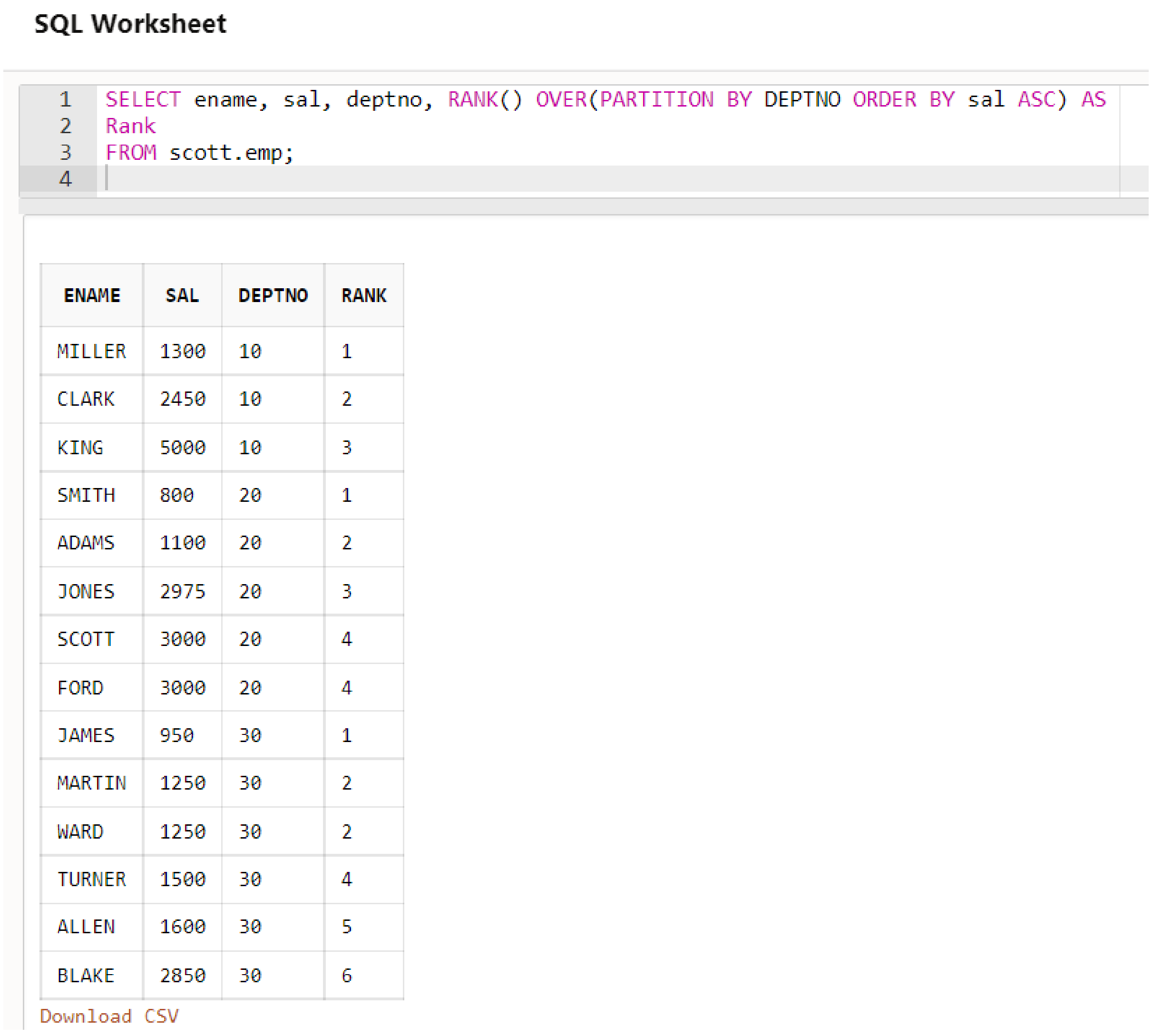
Download CSV

© 2022 Oracle · Live SQL 22.3.1, running Oracle Database 19c Enterprise Edition - 19.14.0.0.0 · Database Documentation · Ask Tom · Dev Gym

8. Calculate the ranks of employee for each department according to sal.

Script: SELECT ename, sal, deptno, RANK() OVER(PARTITION BY DEPTNO ORDER BY sal ASC) AS Rank FROM scott.emp;

Output:



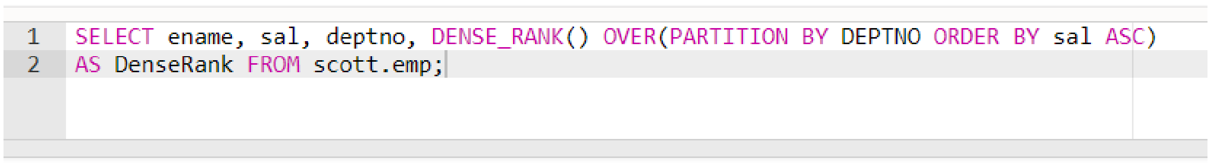
```
SELECT ename, sal, deptno, RANK() OVER(PARTITION BY DEPTNO ORDER BY sal ASC) AS
Rank
FROM scott.emp;
```

ENAME	SAL	DEPTNO	RANK
MILLER	1300	10	1
CLARK	2450	10	2
KING	5000	10	3
SMITH	800	20	1
ADAMS	1100	20	2
JONES	2975	20	3
SCOTT	3000	20	4
FORD	3000	20	4
JAMES	950	30	1
MARTIN	1250	30	2
WARD	1250	30	2
TURNER	1500	30	4
ALLEN	1600	30	5
BLAKE	2850	30	6

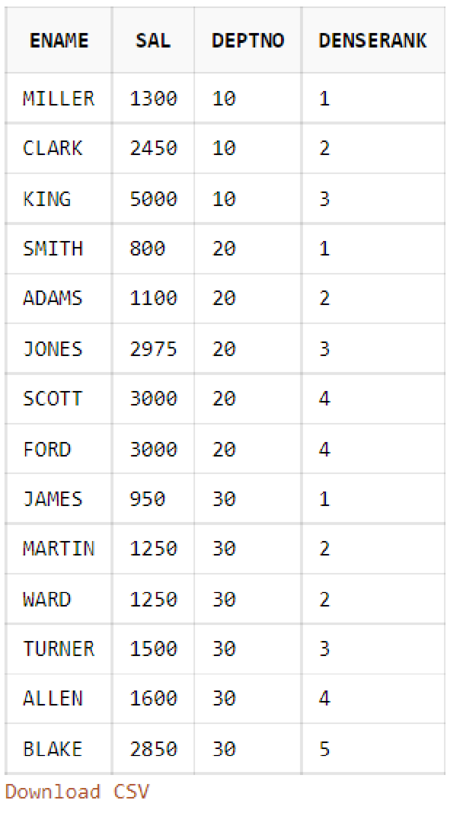
9. For the above query use dense_rank() and point out the difference.

Script: SELECT ename, sal, deptno, DENSE_RANK() OVER(PARTITION BY DEPTNO ORDER BY sal ASC) AS DenseRank FROM scott.emp;

Output:



```
SELECT ename, sal, deptno, DENSE_RANK() OVER(PARTITION BY DEPTNO ORDER BY sal ASC)
AS DenseRank FROM scott.emp;
```



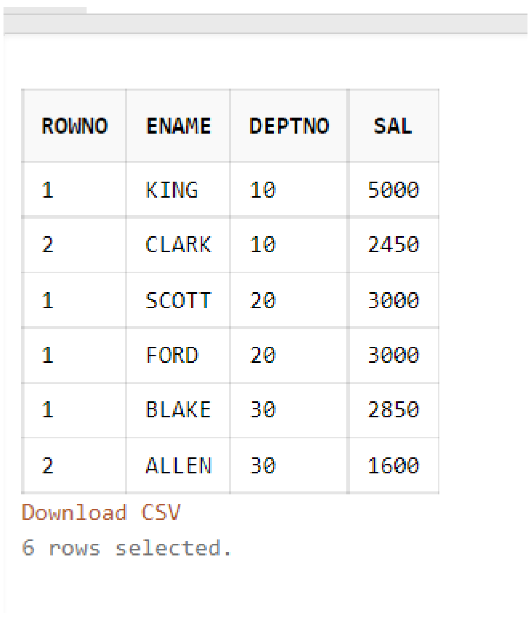
ENAME	SAL	DEPTNO	DENSERANK
MILLER	1300	10	1
CLARK	2450	10	2
KING	5000	10	3
SMITH	800	20	1
ADAMS	1100	20	2
JONES	2975	20	3
SCOTT	3000	20	4
FORD	3000	20	4
JAMES	950	30	1
MARTIN	1250	30	2
WARD	1250	30	2
TURNER	1500	30	3
ALLEN	1600	30	4
BLAKE	2850	30	5

Download CSV

10. Calculate the ranks of employee for each department &amp; display only top 2 high salaried employees for each of them.

Script: **SELECT RowNo,ename,deptno,sal**
FROM (SELECT RANK() OVER(PARTITION BY deptno ORDER BY sal DESC)RowNo,empno,deptno,ename,sal FROM Scott.emp) WHERE RowNo<=2;

Output:



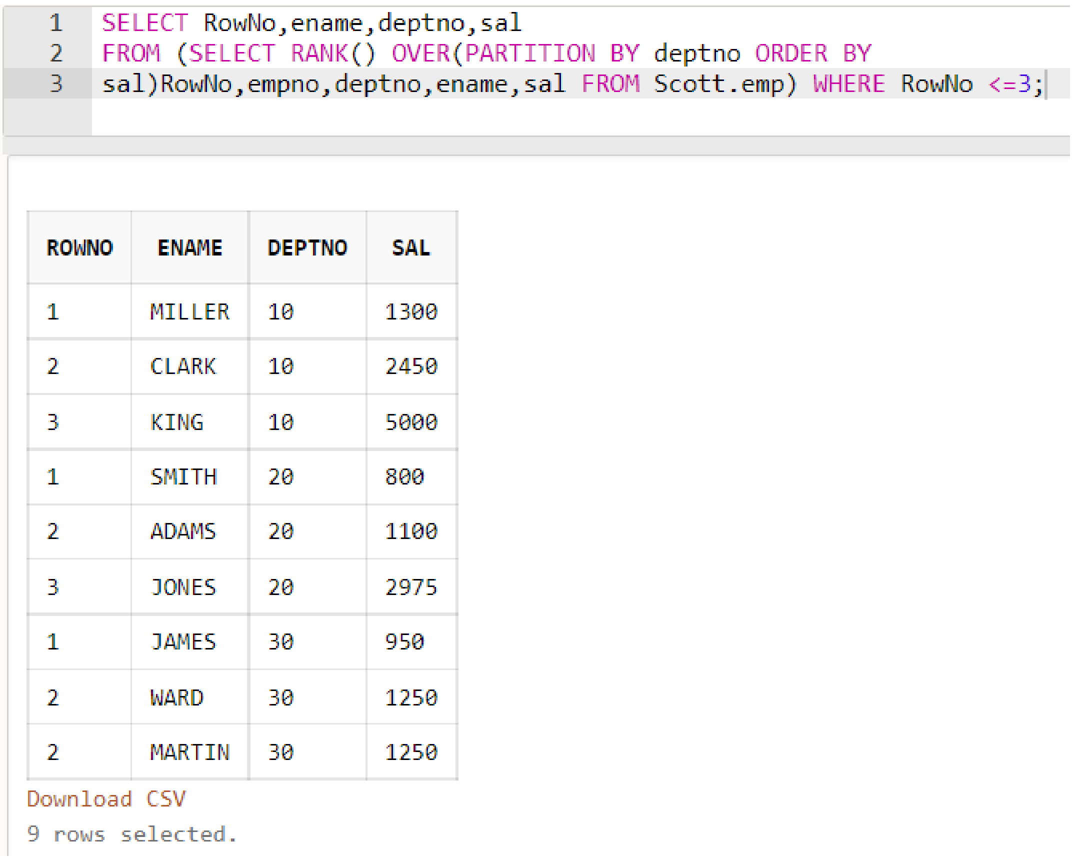
ROWNO	ENAME	DEPTNO	SAL
1	KING	10	5000
2	CLARK	10	2450
1	SCOTT	20	3000
1	FORD	20	3000
1	BLAKE	30	2850
2	ALLEN	30	1600

```
Download CSV
6 rows selected.
```

11. Find out top 3 low salaried employees for each department.

**Script: SELECT RowNo,ename,deptno,sal
FROM (SELECT RANK() OVER(PARTITION BY deptno ORDER BY sal)RowNo,empno,deptno,ename,sal FROM Scott.emp) WHERE RowNo <=3;**

Output:



```
SELECT RowNo,ename,deptno,sal
FROM (SELECT RANK() OVER(PARTITION BY deptno ORDER BY
sal)RowNo,empno,deptno,ename,sal FROM Scott.emp) WHERE RowNo <=3;
```

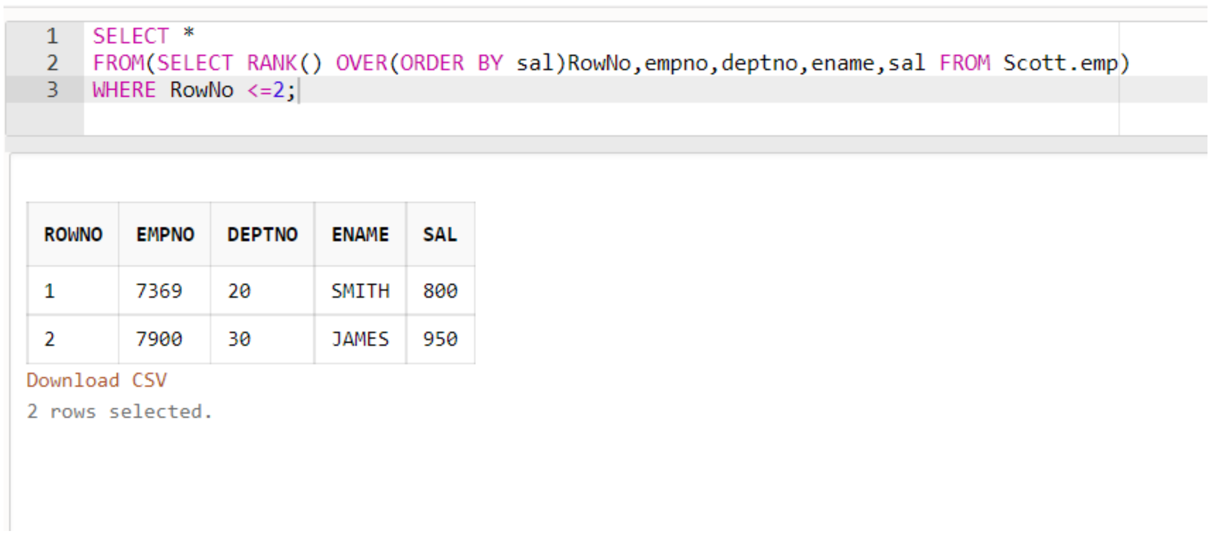
ROWNO	ENAME	DEPTNO	SAL
1	MILLER	10	1300
2	CLARK	10	2450
3	KING	10	5000
1	SMITH	20	800
2	ADAMS	20	1100
3	JONES	20	2975
1	JAMES	30	950
2	WARD	30	1250
2	MARTIN	30	1250

Download CSV
9 rows selected.

12. Find out top 2 low salaried employees.

Script: **SELECT * FROM(SELECT RANK() OVER(ORDER BY sal)RowNo,empno,deptno,ename,sal FROM Scott.emp)
WHERE RowNo <=2;**

Output:



```
SELECT *
FROM(SELECT RANK() OVER(ORDER BY sal)RowNo,empno,deptno,ename,sal FROM Scott.emp)
WHERE RowNo <=2;
```

ROWNO	EMPNO	DEPTNO	ENAME	SAL
1	7369	20	SMITH	800
2	7900	30	JAMES	950

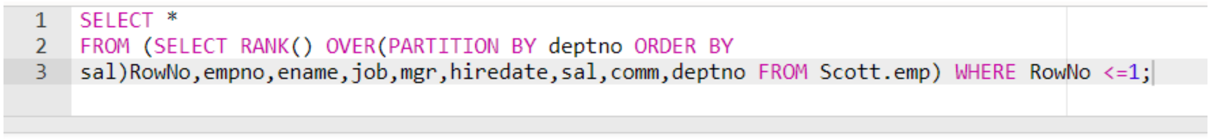
Download CSV

2 rows selected.

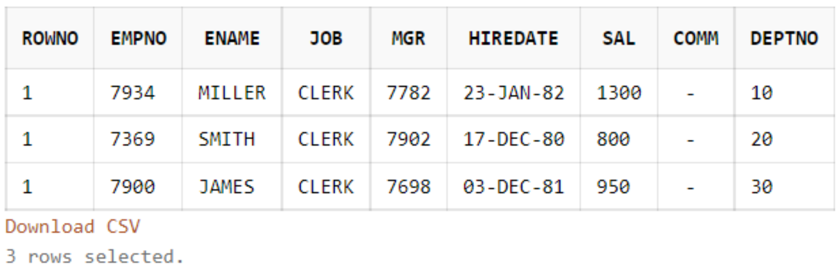
13. Find information of employee who is having lowest sal in each department.

Script: **SELECT ***
FROM (SELECT RANK() OVER(PARTITION BY deptno ORDER BY sal)RowNo,empno,ename,job,mgr,hiredate,sal,comm,deptno FROM Scott.emp) WHERE RowNo <=1;

Output:



```
SELECT *
FROM (SELECT RANK() OVER(PARTITION BY deptno ORDER BY
sal)RowNo,empno,ename,job,mgr,hiredate,sal,comm,deptno FROM Scott.emp) WHERE RowNo <=1;
```



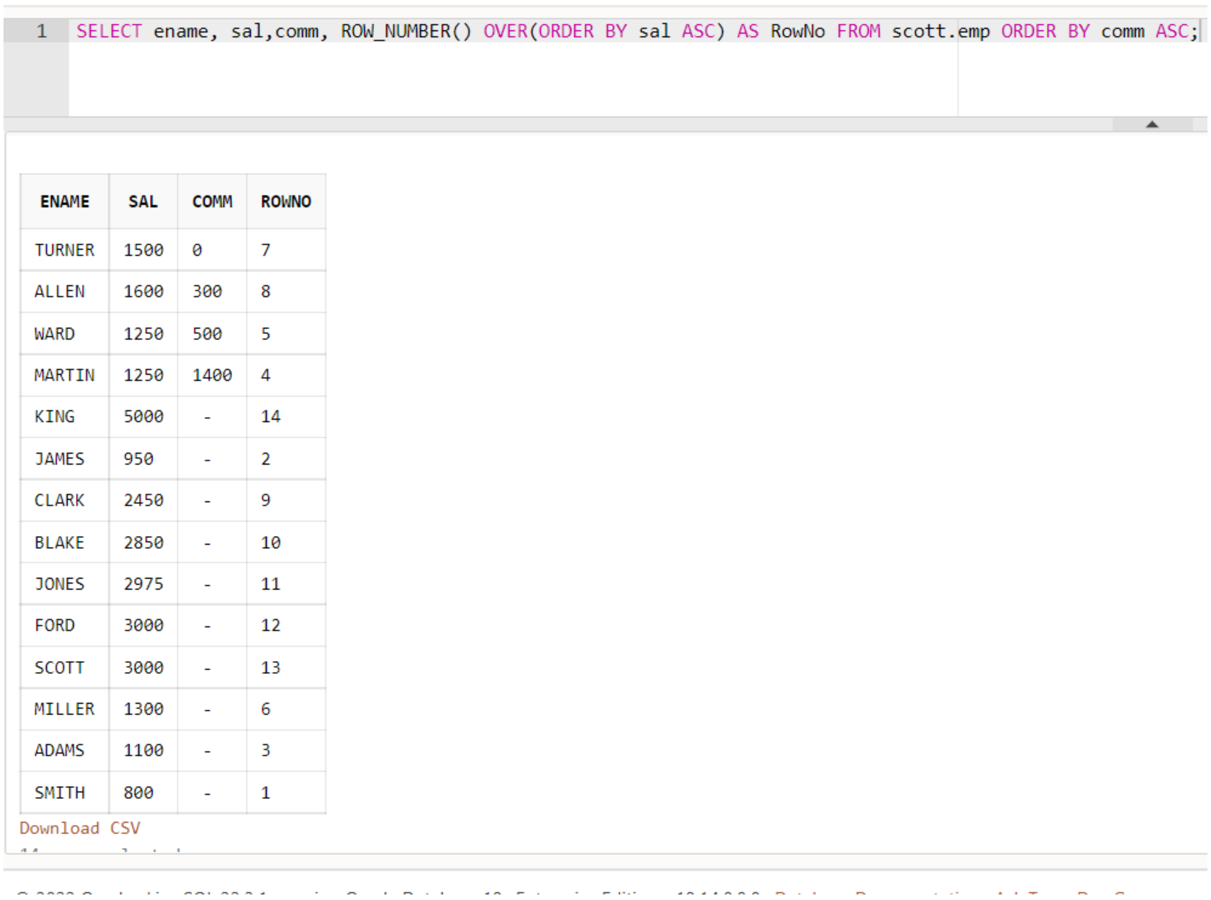
ROWNO	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7934	MILLER	CLERK	7782	23-JAN-82	1300	-	10
1	7369	SMITH	CLERK	7902	17-DEC-80	800	-	20
1	7900	JAMES	CLERK	7698	03-DEC-81	950	-	30

Download CSV
3 rows selected.

14. Assign row numbers in desc order of salary. Display the records in asc order of commission and null values of commission should come last.

Script: SELECT ename, sal,comm, ROW_NUMBER() OVER(ORDER BY sal ASC) AS RowNo FROM scott.emp ORDER BY comm ASC;

Output:



```
SELECT ename, sal,comm, ROW_NUMBER() OVER(ORDER BY sal ASC) AS RowNo FROM scott.emp ORDER BY comm ASC;
```

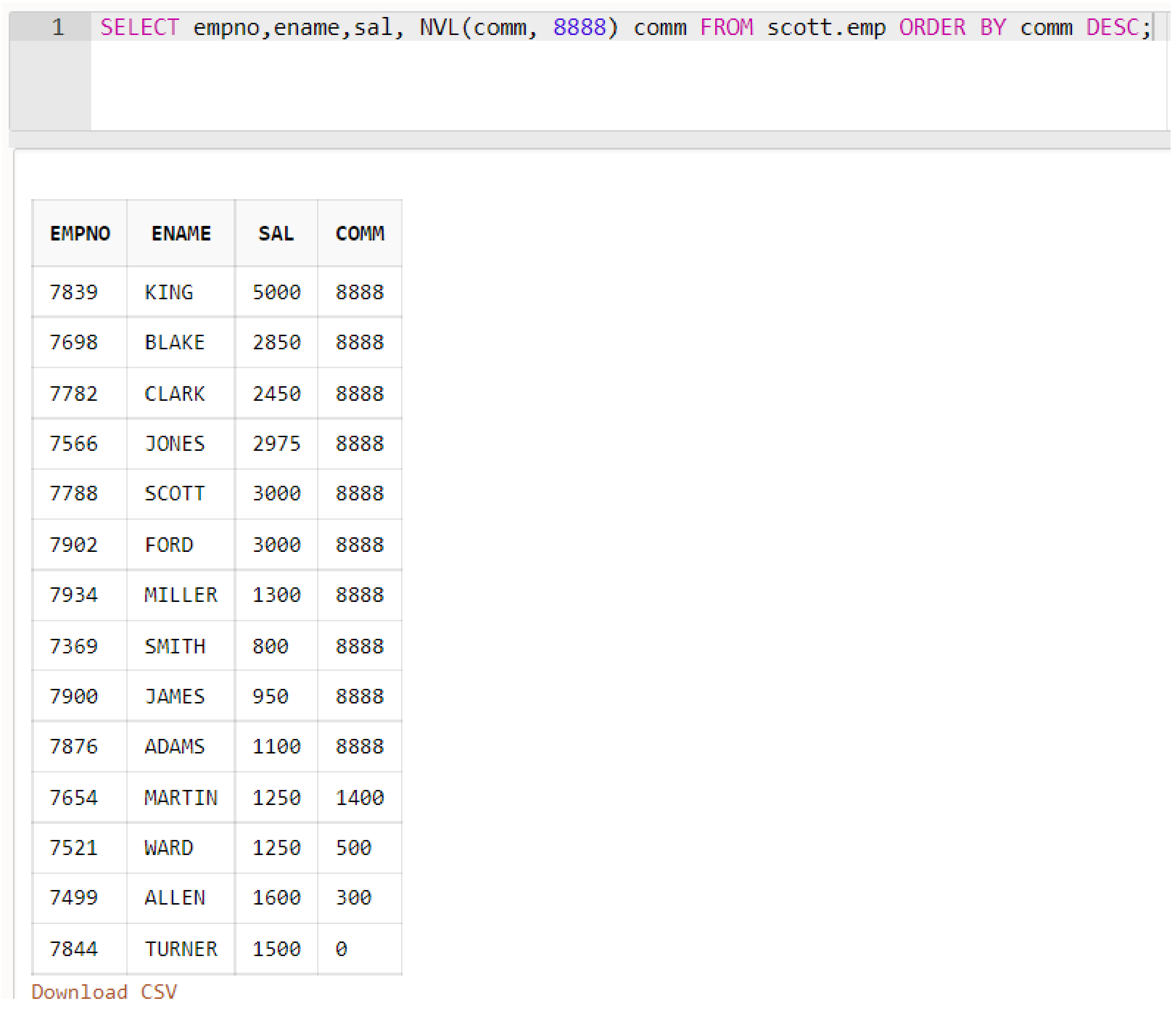
ENAME	SAL	COMM	ROWNO
TURNER	1500	0	7
ALLEN	1600	300	8
WARD	1250	500	5
MARTIN	1250	1400	4
KING	5000	-	14
JAMES	950	-	2
CLARK	2450	-	9
BLAKE	2850	-	10
JONES	2975	-	11
FORD	3000	-	12
SCOTT	3000	-	13
MILLER	1300	-	6
ADAMS	1100	-	3
SMITH	800	-	1

Download CSV

15. Display empno, ename, sal, comm., in desc order of comm. And replace all null values of comm. by 8888.

Script: **SELECT empno,ename,sal, NVL(comm, 8888) comm FROM scott.emp ORDER BY comm DESC;**

Output:



```
SELECT empno,ename,sal, NVL(comm, 8888) comm FROM scott.emp ORDER BY comm DESC;
```

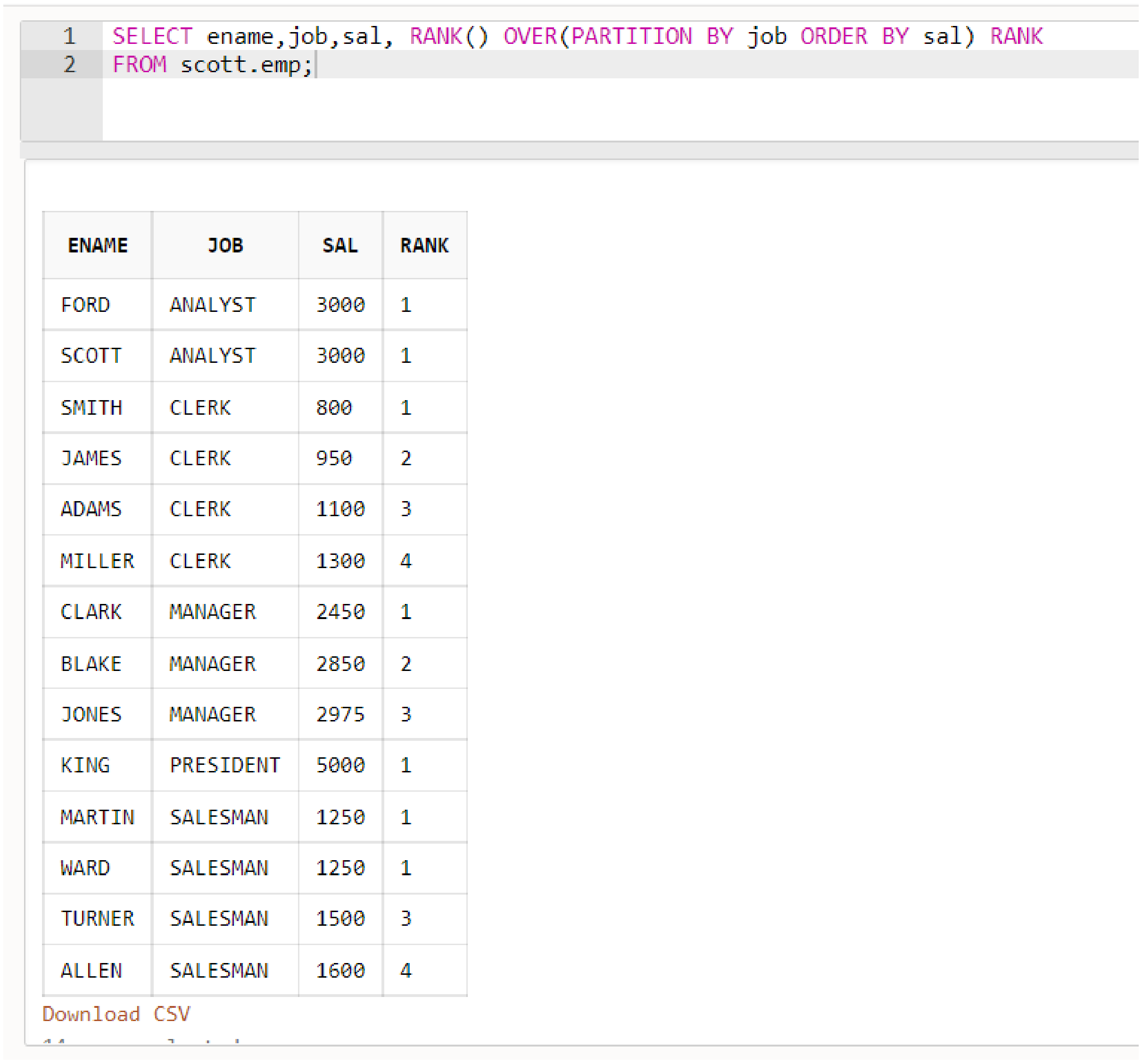
EMPNO	ENAME	SAL	COMM
7839	KING	5000	8888
7698	BLAKE	2850	8888
7782	CLARK	2450	8888
7566	JONES	2975	8888
7788	SCOTT	3000	8888
7902	FORD	3000	8888
7934	MILLER	1300	8888
7369	SMITH	800	8888
7900	JAMES	950	8888
7876	ADAMS	1100	8888
7654	MARTIN	1250	1400
7521	WARD	1250	500
7499	ALLEN	1600	300
7844	TURNER	1500	0

Download CSV

16. Display empname, job &amp; sal. Give ranking to sal job wise.

Script: SELECT ename,job,sal, RANK() OVER(PARTITION BY job ORDER BY sal) RANK FROM scott.emp;

Output:



```
SELECT ename,job,sal, RANK() OVER(PARTITION BY job ORDER BY sal) RANK
FROM scott.emp;
```

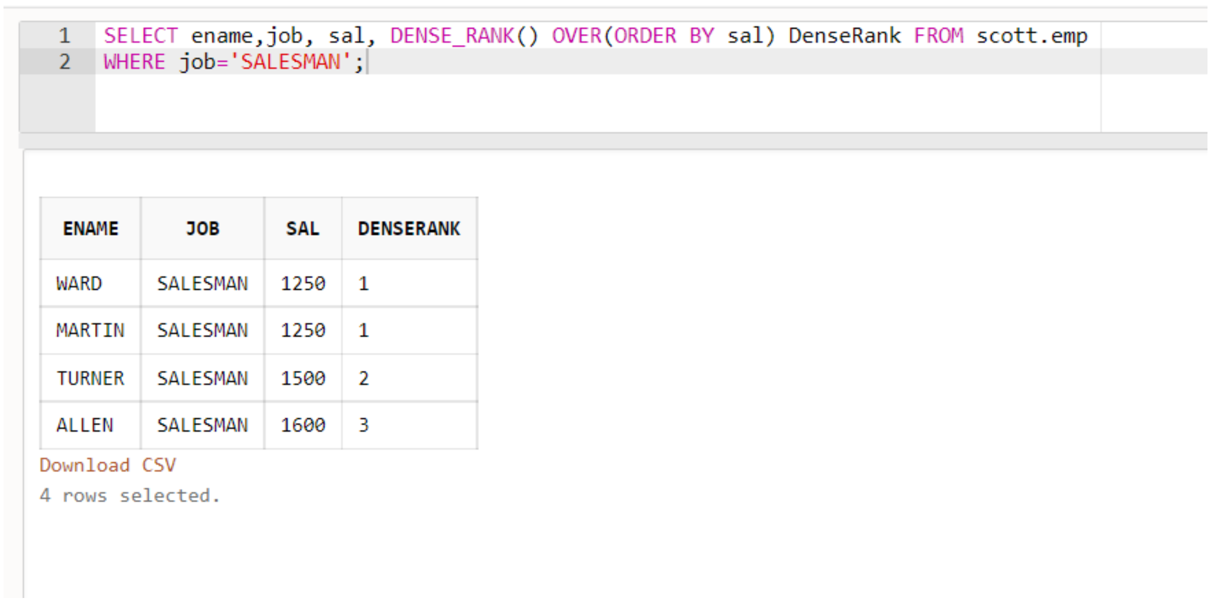
ENAME	JOB	SAL	RANK
FORD	ANALYST	3000	1
SCOTT	ANALYST	3000	1
SMITH	CLERK	800	1
JAMES	CLERK	950	2
ADAMS	CLERK	1100	3
MILLER	CLERK	1300	4
CLARK	MANAGER	2450	1
BLAKE	MANAGER	2850	2
JONES	MANAGER	2975	3
KING	PRESIDENT	5000	1
MARTIN	SALESMAN	1250	1
WARD	SALESMAN	1250	1
TURNER	SALESMAN	1500	3
ALLEN	SALESMAN	1600	4

Download CSV

17. Display the details of all salesman using dense rank on ascending order of salary.

Script: SELECT ename,job, sal, DENSE_RANK() OVER(ORDER BY sal) DenseRank FROM scott.emp WHERE job='SALESMAN';

Output:



```
SELECT ename,job, sal, DENSE_RANK() OVER(ORDER BY sal) DenseRank FROM scott.emp
WHERE job='SALESMAN';
```

ENAME	JOB	SAL	DENSERANK
WARD	SALESMAN	1250	1
MARTIN	SALESMAN	1250	1
TURNER	SALESMAN	1500	2
ALLEN	SALESMAN	1600	3

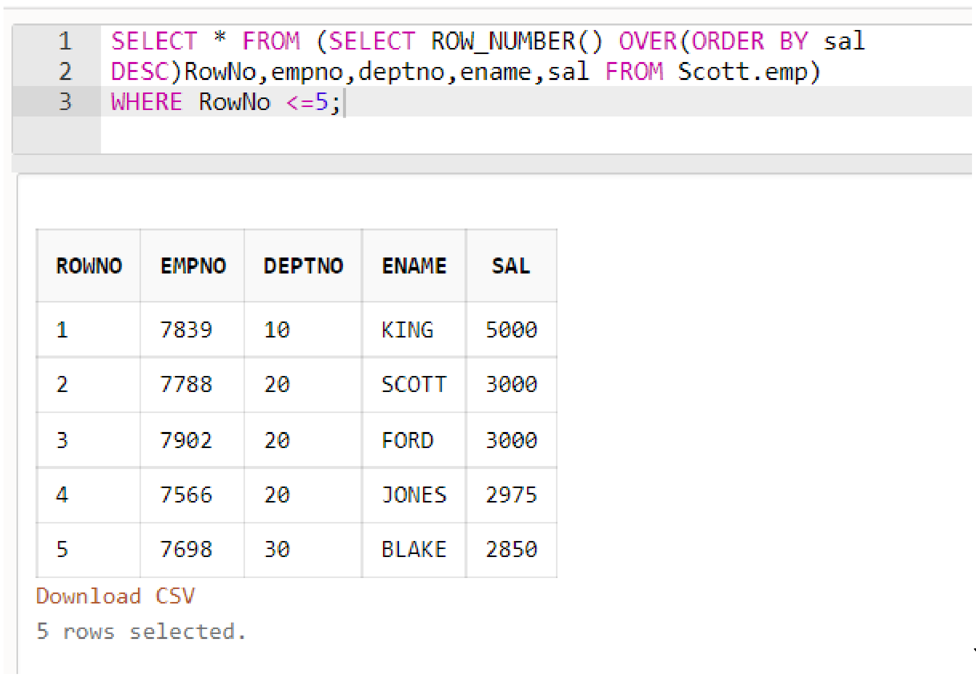
Download CSV

4 rows selected.

18. Display first 5 records of employee in descending order of salary.

Script: SELECT * FROM (SELECT ROW_NUMBER() OVER(ORDER BY sal DESC)RowNo,empno,deptno,ename,sal FROM Scott.emp) WHERE RowNo <=5;

Output:



```
SELECT * FROM (SELECT ROW_NUMBER() OVER(ORDER BY sal
DESC)RowNo,empno,deptno,ename,sal FROM Scott.emp)
WHERE RowNo <=5;
```

ROWNO	EMPNO	DEPTNO	ENAME	SAL
1	7839	10	KING	5000
2	7788	20	SCOTT	3000
3	7902	20	FORD	3000
4	7566	20	JONES	2975
5	7698	30	BLAKE	2850

Download CSV

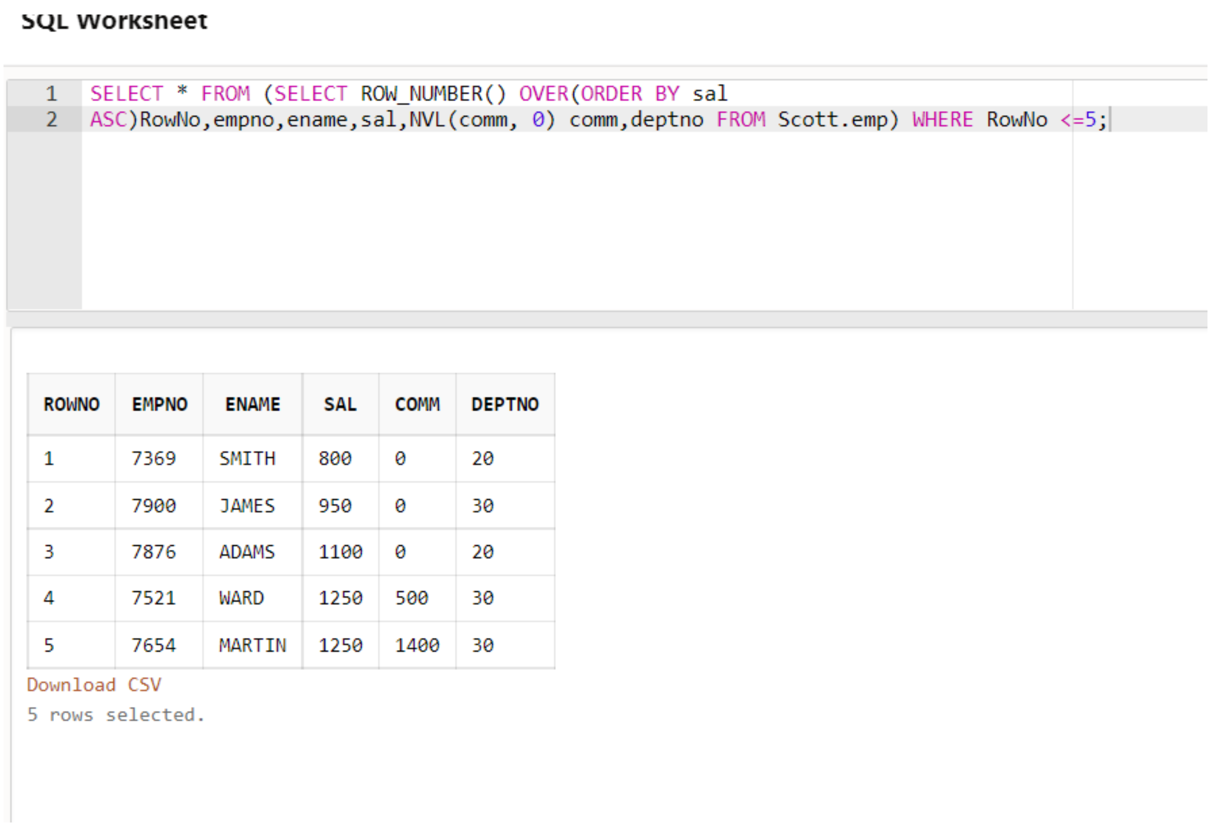
5 rows selected.

`

19. Display first 5 records of employee in ascending order of salary, replace null values of comm. by zero.

Script: **SELECT * FROM (SELECT ROW_NUMBER() OVER(ORDER BY sal ASC)RowNo,empno,ename,sal,NVL(comm, 0) comm,deptno FROM Scott.emp) WHERE RowNo <=5;**

Output:


SQL Worksheet

```
SELECT * FROM (SELECT ROW_NUMBER() OVER(ORDER BY sal
ASC)RowNo,empno,ename,sal,NVL(comm, 0) comm,deptno FROM Scott.emp) WHERE RowNo <=5;
```

ROWNO	EMPNO	ENAME	SAL	COMM	DEPTNO
1	7369	SMITH	800	0	20
2	7900	JAMES	950	0	30
3	7876	ADAMS	1100	0	20
4	7521	WARD	1250	500	30
5	7654	MARTIN	1250	1400	30

Download CSV
5 rows selected.

20. Create weather table with fields month, year and avgtemp. Values could be:

Month Year Avgtemp

1 2012 14.5

2 2012 34.5

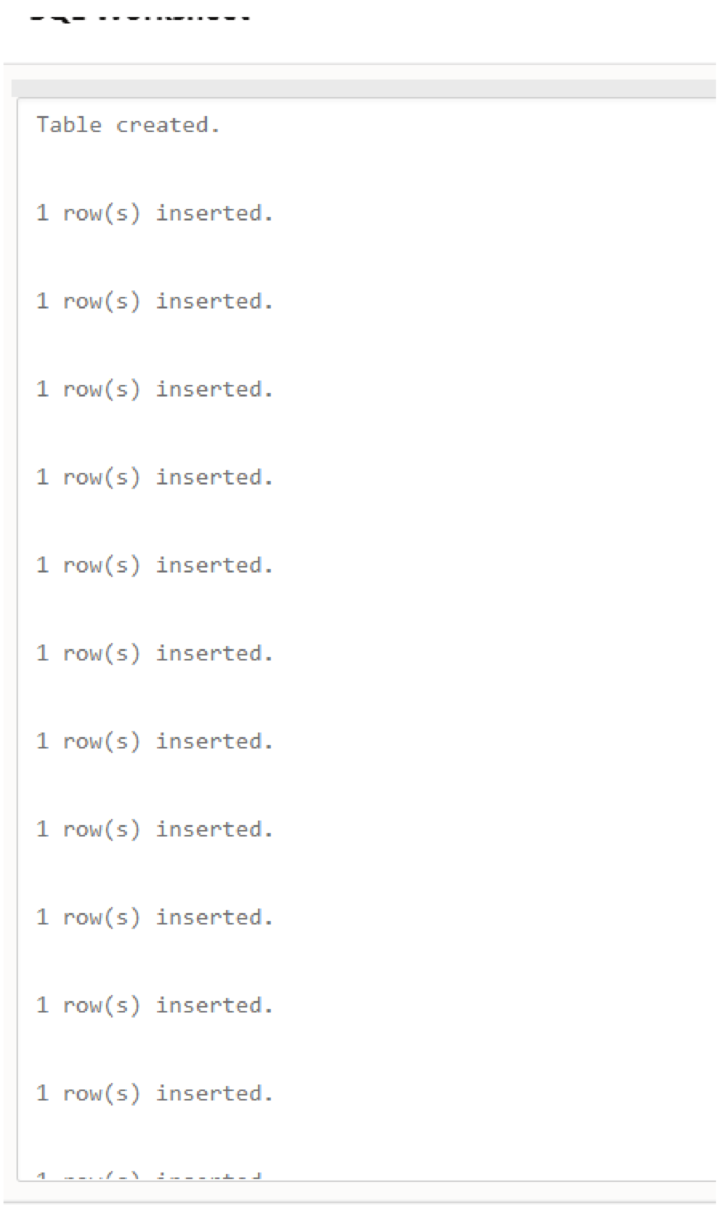
Put atleast 12 records for 4 different years.

a) Use rank function to display the information of weather in order of hottest to coolest month year to year.

b) Find the hottest month of every year.

Script:
```
create table Weather1 ( month number(15), year number(6),
avgtemp number(10,2));
insert into Weather1 values (1,2012,15.5);
insert into Weather1 values (2,2012,14.5);
insert into Weather1 values (3,2013,20.5);
insert into Weather1 values (4,2013,22.5);
insert into Weather1 values (5,2015,27.5);
insert into Weather1 values (6,2014,23.5);
insert into Weather1 values (7,2014,12.5);
insert into Weather1 values (8,2013,35.5);
insert into Weather1 values (9,2015,10.5);
insert into Weather1 values (10,2012,11.5);
insert into Weather1 values (11,2014,22.5);
insert into Weather1 values (12,2012,30.5);
```

Output:



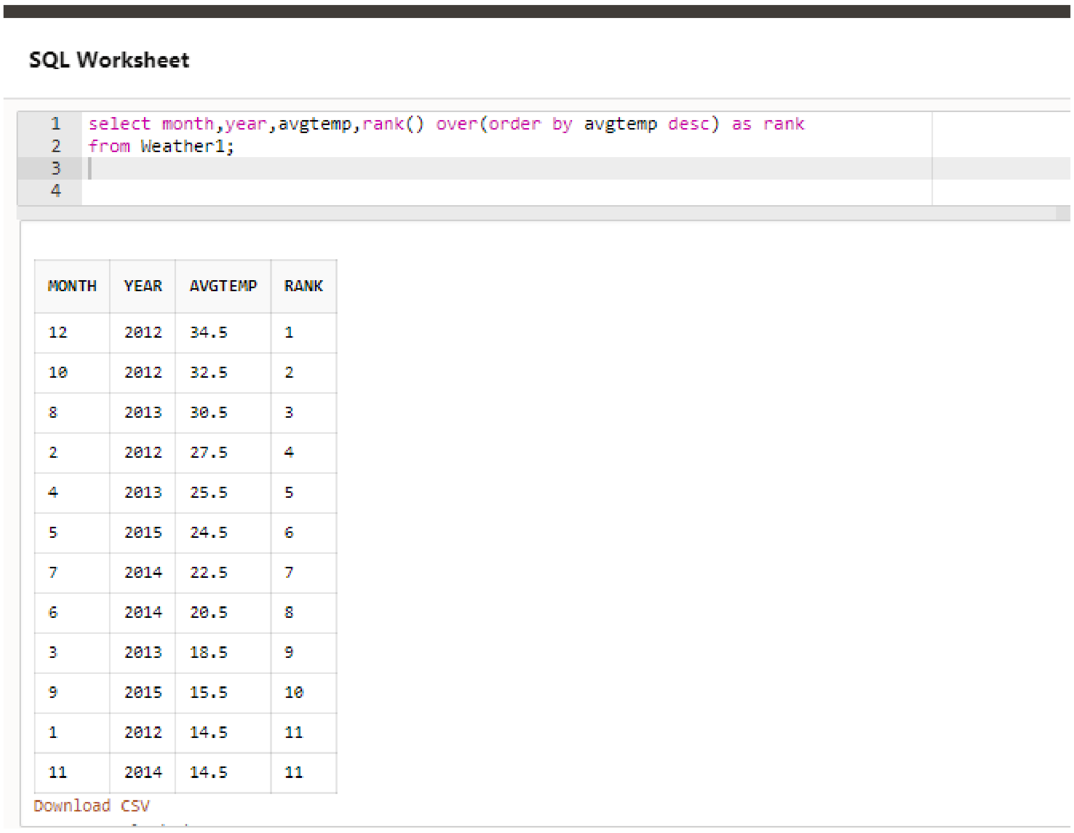
```
Table created.


1 row(s) inserted.


1 row(s) inserted.


1 row(s) inserted.


1 row(s) inserted.


1 row(s) inserted.


1 row(s) inserted.


1 row(s) inserted.


1 row(s) inserted.


1 row(s) inserted.


1 row(s) inserted.


1 row(s) inserted.
```

a) Use rank function to display the information of weather in order of hottest to coolest month year to year.

Code:

select month,year,avgtemp,rank() over(order by avgtemp desc) as rank from Weather1;

Output:



SQL Worksheet

```
select month,year,avgtemp,rank() over(order by avgtemp desc) as rank
from Weather1;
```

MONTH	YEAR	AVGTEMP	RANK
12	2012	34.5	1
10	2012	32.5	2
8	2013	30.5	3
2	2012	27.5	4
4	2013	25.5	5
5	2015	24.5	6
7	2014	22.5	7
6	2014	20.5	8
3	2013	18.5	9
9	2015	15.5	10
1	2012	14.5	11
11	2014	14.5	11

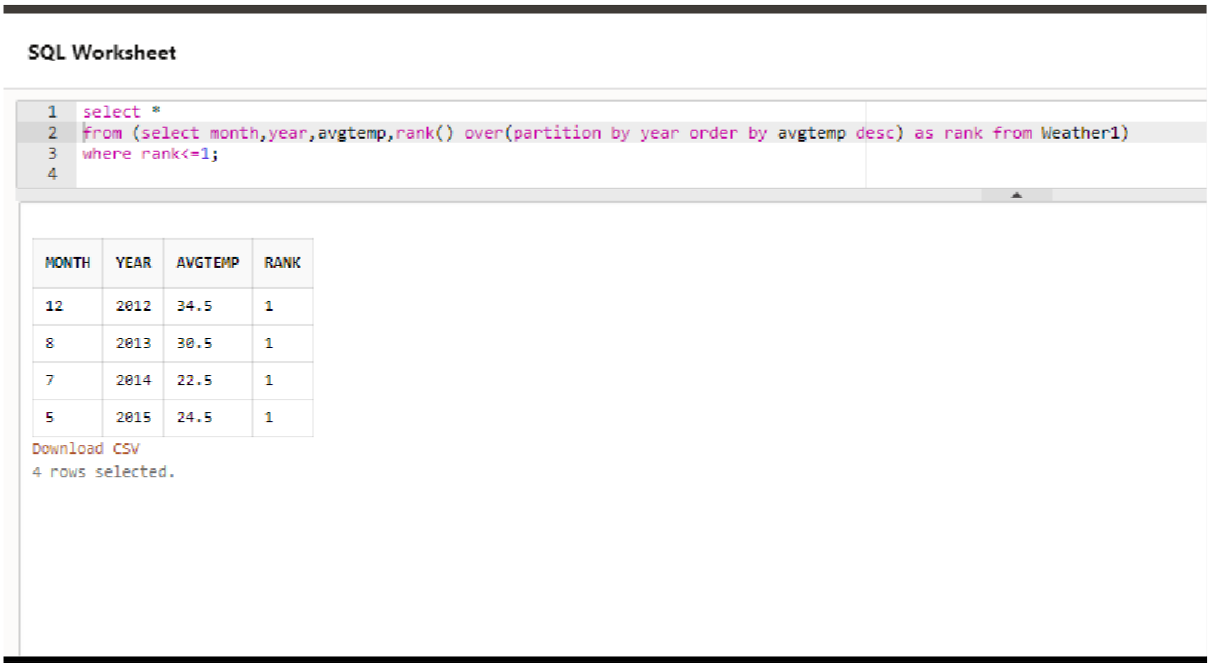
Download CSV

b) Find the hottest month of every year.

Code:

select * from (select month,year,avgtemp,rank() over(partition by year order by avgtemp desc) as rank from Weather1) where rank<=1;

Output:


SQL Worksheet

```
select *
from (select month,year,avgtemp,rank() over(partition by year order by avgtemp desc) as rank from Weather1)
where rank<=1;
```

MONTH	YEAR	AVGTEMP	RANK
12	2012	34.5	1
8	2013	30.5	1
7	2014	22.5	1
5	2015	24.5	1

Download CSV

4 rows selected.

Topic: Analytical functions

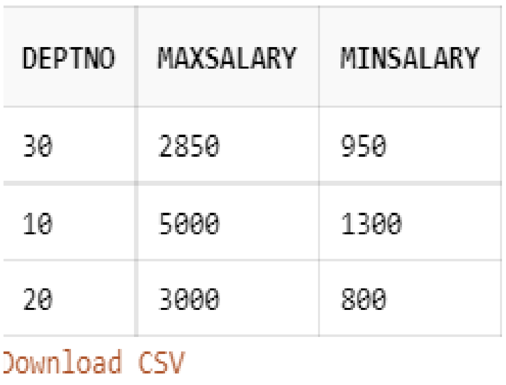
(keep…First, keep… Last, Lead(), Lag())

1. Write a query for finding highest and lowest salary of each department.

Script:

select deptno, max(sal) as MaxSalary, min(sal) as MinSalary from emp group by DEPTNO;

Output:



DEPTNO	MAXSALARY	MINSALARY
30	2850	950
10	5000	1300
20	3000	800

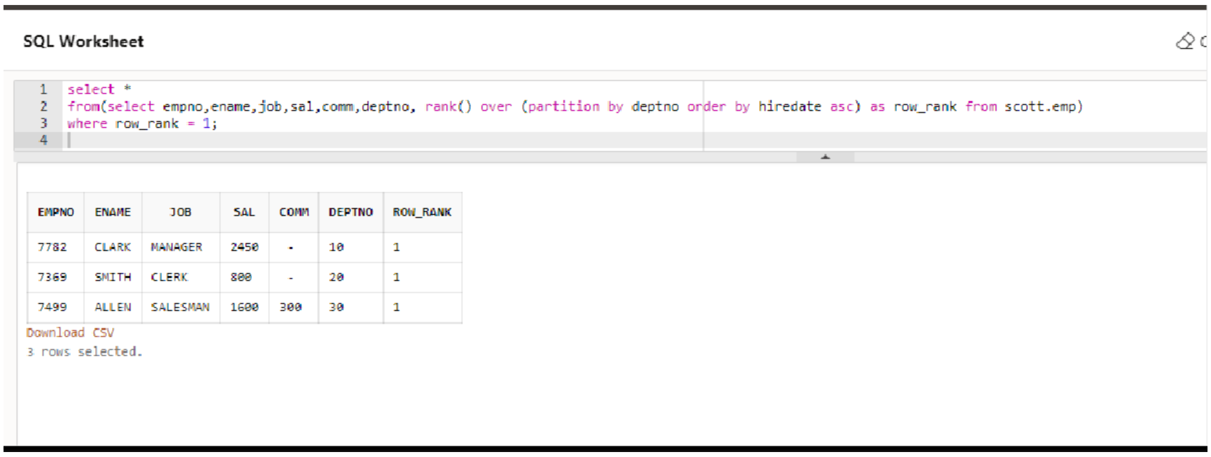
Download CSV

2. Write a query to find information of employees who were hired first in each department.

Script:

**select *
from(select empno,ename,job,sal,comm,deptno, rank() over (partition by deptno order by hiredate asc) as row_rank from scott.emp)
where row_rank = 1;**

Output:



SQL Worksheet

```
select *
from(select empno,ename,job,sal,comm,deptno, rank() over (partition by deptno order by hiredate asc) as row_rank from scott.emp)
where row_rank = 1;
```

EMPNO	ENAME	JOB	SAL	COMM	DEPTNO	ROW_RANK
7782	CLARK	MANAGER	2450	-	10	1
7369	SMITH	CLERK	800	-	20	1
7499	ALLEN	SALESMAN	1600	300	30	1

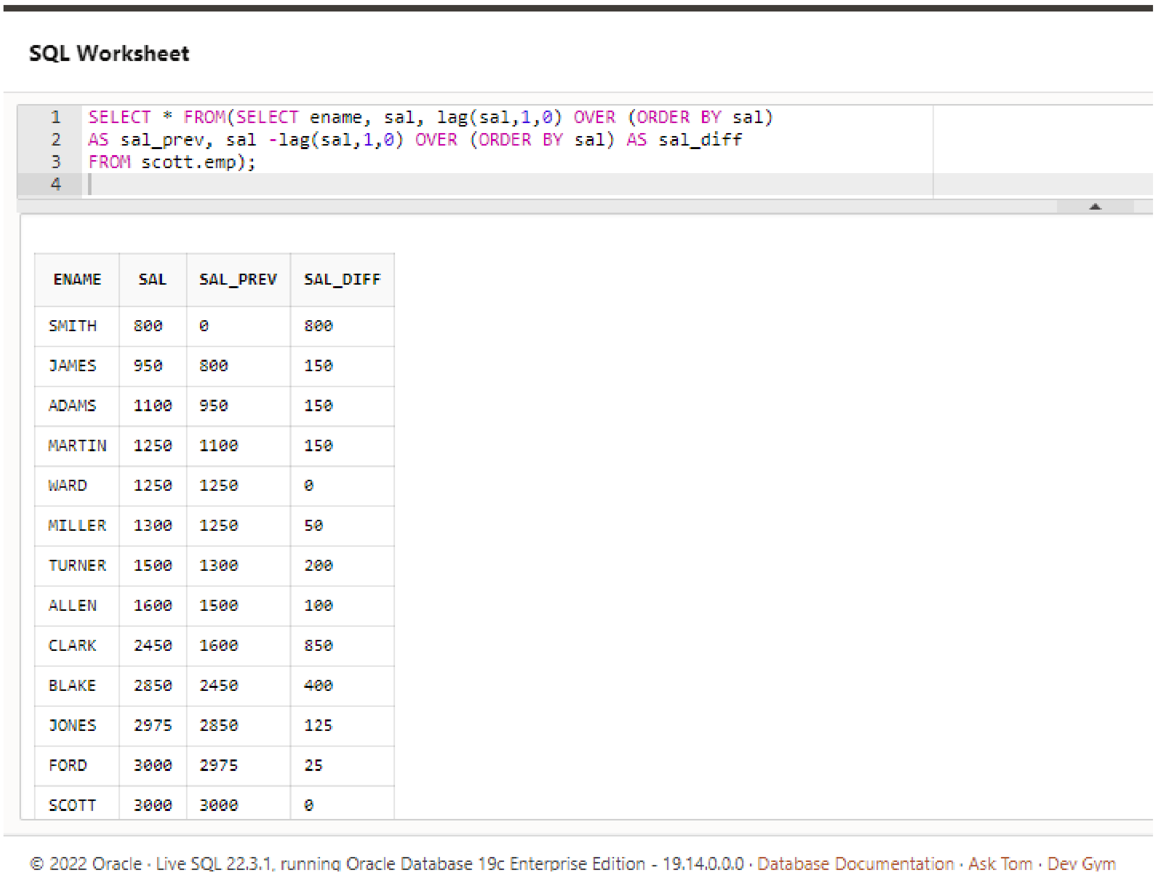
Download CSV
3 rows selected.

3. Write a query which returns the salary from previous row; give the column name as sal_prev. calculate the difference between sal of current row and that of previous row.

Script:

SELECT * FROM(SELECT ename, sal, lag(sal,1,0) OVER (ORDER BY sal) AS sal_prev, sal - lag(sal,1,0) OVER (ORDER BY sal) AS sal_diff
FROM scott.emp);

Output:


SQL Worksheet

```
SELECT * FROM(SELECT ename, sal, lag(sal,1,0) OVER (ORDER BY sal)
AS sal_prev, sal -lag(sal,1,0) OVER (ORDER BY sal) AS sal_diff
FROM scott.emp);
```

ENAME	SAL	SAL_PREV	SAL_DIFF
SMITH	800	0	800
JAMES	950	800	150
ADAMS	1100	950	150
MARTIN	1250	1100	150
WARD	1250	1250	0
MILLER	1300	1250	50
TURNER	1500	1300	200
ALLEN	1600	1500	100
CLARK	2450	1600	850
BLAKE	2850	2450	400
JONES	2975	2850	125
FORD	3000	2975	25
SCOTT	3000	3000	0

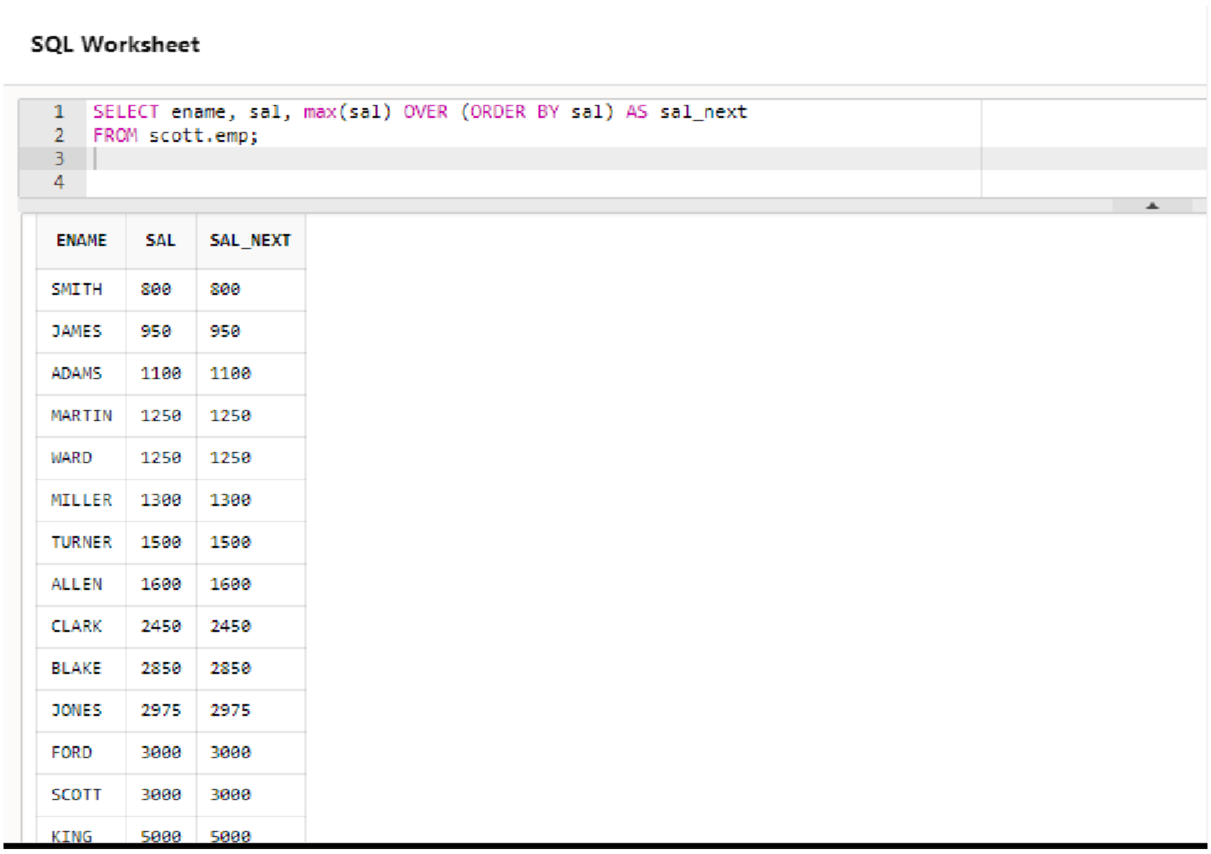
© 2022 Oracle · Live SQL 22.3.1, running Oracle Database 19c Enterprise Edition - 19.14.0.0.0 · Database Documentation · Ask Tom · Dev Gym

4. Write a query which returns a salary from next row, name it as sal_next and calculate the diff between the sal of current and following row.

Script:

SELECT ename, sal, max(sal) OVER (ORDER BY sal) AS sal_next FROM scott.emp;

Output:



```
SELECT ename, sal, max(sal) OVER (ORDER BY sal) AS sal_next
FROM scott.emp;
```

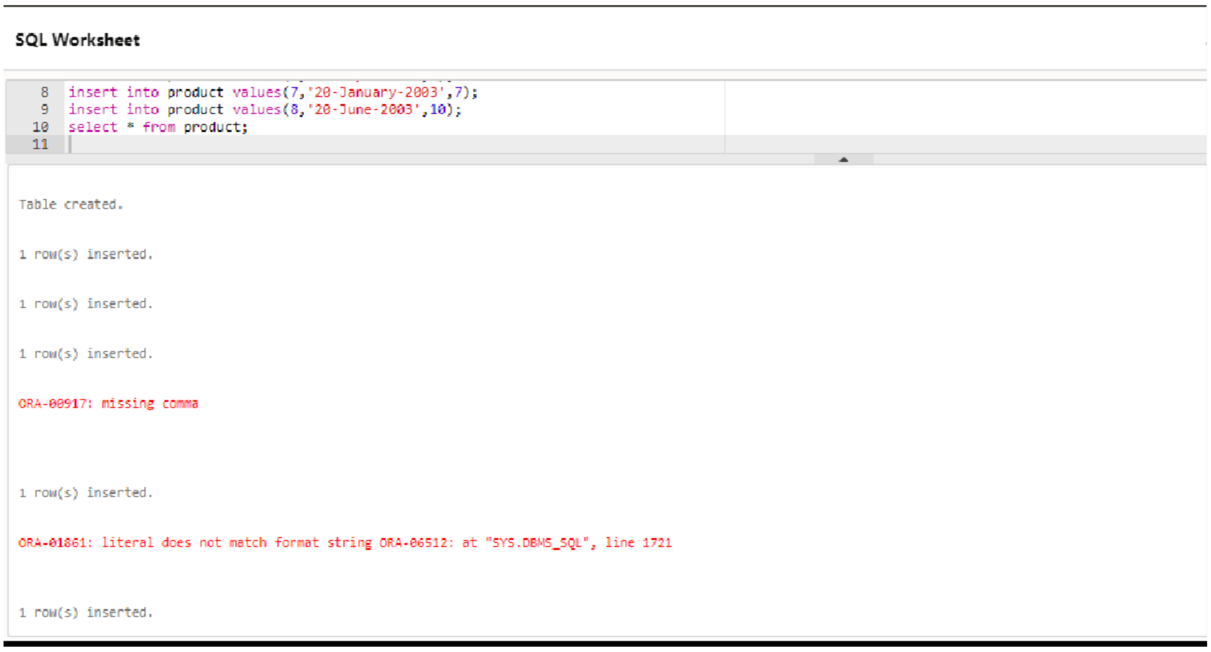
ENAME	SAL	SAL_NEXT
SMITH	800	800
JAMES	950	950
ADAMS	1100	1100
MARTIN	1250	1250
WARD	1250	1250
MILLER	1300	1300
TURNER	1500	1500
ALLEN	1600	1600
CLARK	2450	2450
BLAKE	2850	2850
JONES	2975	2975
FORD	3000	3000
SCOTT	3000	3000
KING	5000	5000

5. Create a table with fields pro_id, order date, quantity. Insert at least 6 records into it.

Script:

create table product(prod_id number(10) primary key, order_date date, quantity number(10));
insert into product values(1,'20-june-2001',5);
insert into product values(2,'11-january-2002',8);
insert into product values(3,'20-march-2002',4);
insert into product values(4,'20-june-2002'5);
insert into product values(5,'20-july-2002',6);
insert into product values(6,'20-Sept-2002',5);
insert into product values(7,'20-January-2003',7);
insert into product values(8,'20-June-2003',10);
select * from product;

Output:



SQL Worksheet

```
insert into product values(7,'20-January-2003',7);
insert into product values(8,'20-June-2003',10);
select * from product;
```

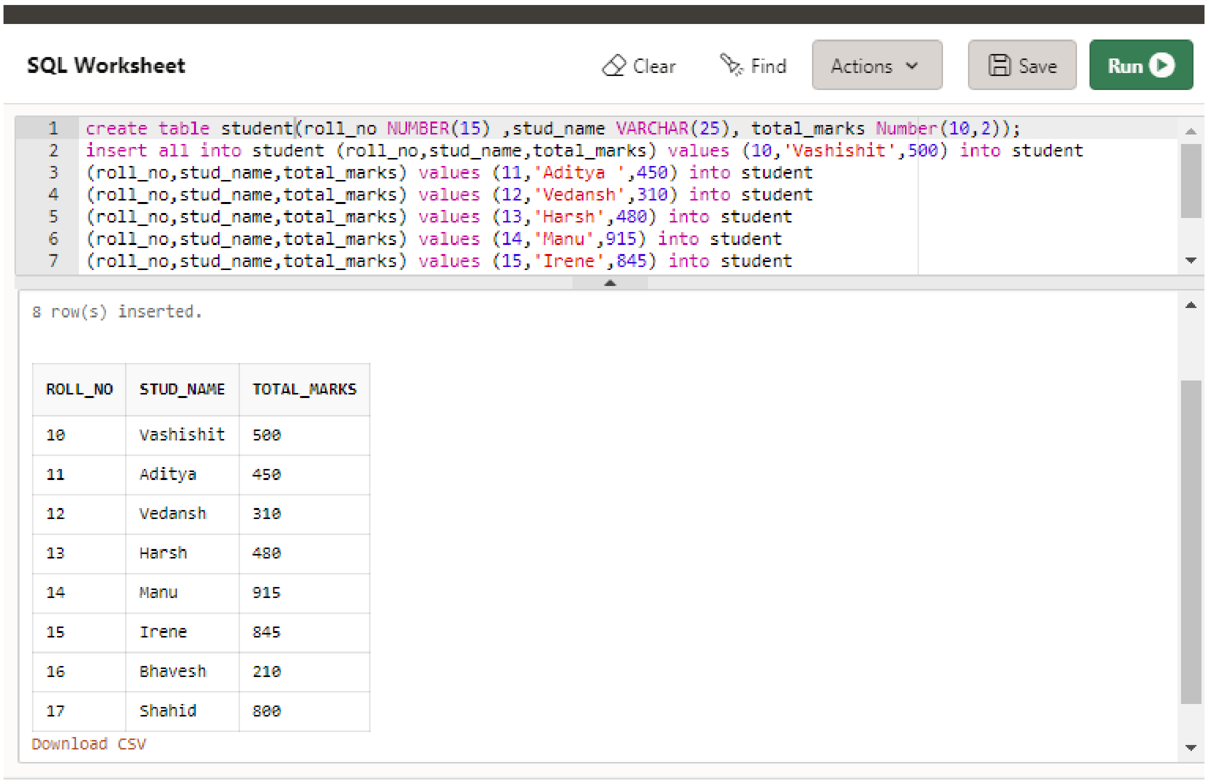
```
Table created.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

ORA-00917: missing comma



1 row(s) inserted.

ORA-01861: literal does not match format string ORA-06512: at "SYS.DBMS_SQL", line 1721


1 row(s) inserted.
```

6. Create a table student with roll no, stud_name, total_marks scored in last semester. Insert at least 6 records in it. Find out difference between different rank holders in class.

Script:

create table student(roll_no NUMBER(20) stud_name VARCHAR(25), total_marks Number(15,5));
insert all into student (roll_no,stud_name,total_marks) values (10,'Vashishit',500) into student
(roll_no,stud_name,total_marks) values (11,'Aditya ',450) into student
(roll_no,stud_name,total_marks) values (12,'Vedansh',310) into student
(roll_no,stud_name,total_marks) values (13,'Harsh',480) into student
(roll_no,stud_name,total_marks) values (14,'Manu',915) into student
(roll_no,stud_name,total_marks) values (15,'Irene',845) into student
(roll_no,stud_name,total_marks) values (16,'Bhavesh',210) into student
(roll_no,stud_name,total_marks) values (17,'Shahid ',800)
select * from dual; select * from student;

Output:



```
create table student(roll_no NUMBER(15) ,stud_name VARCHAR(25), total_marks Number(10,2));
insert all into student (roll_no,stud_name,total_marks) values (10,'Vashishit',500) into student
(roll_no,stud_name,total_marks) values (11,'Aditya ',450) into student
(roll_no,stud_name,total_marks) values (12,'Vedansh',310) into student
(roll_no,stud_name,total_marks) values (13,'Harsh',480) into student
(roll_no,stud_name,total_marks) values (14,'Manu',915) into student
(roll_no,stud_name,total_marks) values (15,'Irene',845) into student
```

```
8 row(s) inserted.
```

ROLL_NO	STUD_NAME	TOTAL_MARKS
10	Vashishit	500
11	Aditya	450
12	Vedansh	310
13	Harsh	480
14	Manu	915
15	Irene	845
16	Bhavesh	210
17	Shahid	800

Download CSV