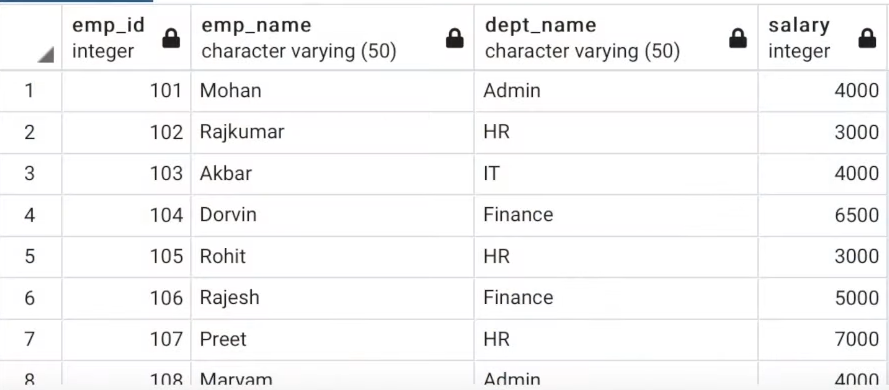
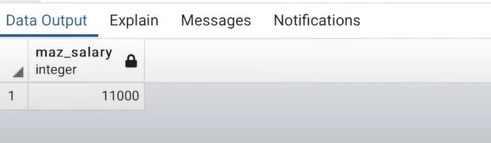
EMPLOYEE TABLE:



Q-> MAX SALARY EARNED BY AN EMPLOYEE?

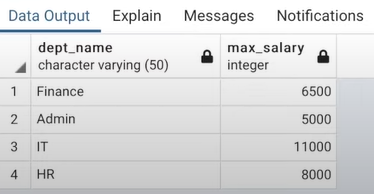
SELECT MAX(SALARY) AS MAXIMUM FROM EMPLOYEE;

O/P:



Q->DEPT NAME AND MAXIMUM SALARY FOR THE EMPLOYEE IN EACH DEPARTMENT?

SELECT DEPT\_NAME, MAX(SALARY) AS EACH\_MAX FROM EMPLOYEE GROUP BY DEPT\_NAME;

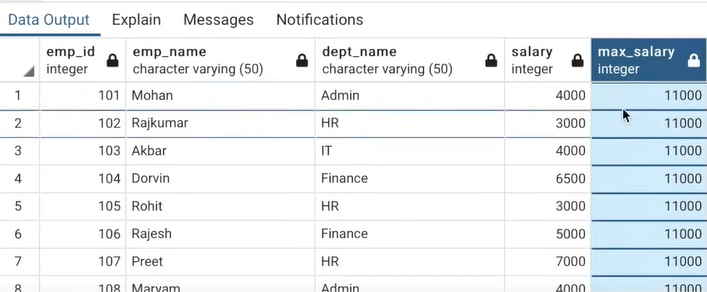


Q-> ALL DETAILS AND MAXIMUM SALARY FOR THE EMPLOYEE IN EACH DEPARTMENT?

-Since we could not use aggregate functions here, so window functions come in use

🡪PROCEDURE:

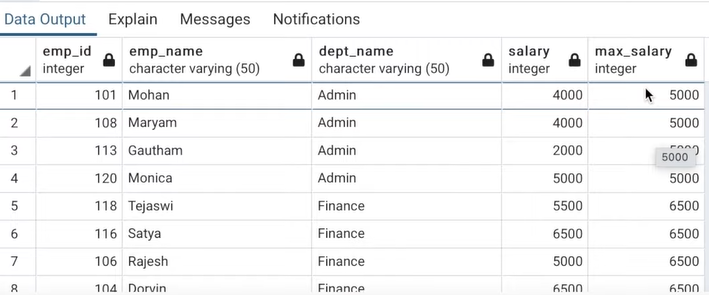
SELECT E.\*, MAX(SALARY) OVER() AS MAX\_SALARY FROM EMPLOYEE E;



HERE NO COLS IS GIVEN IN OVER() WINDOW FUNCTION SO SQL WILL CREATE ONE WINDOW FOR THAT SINGLE(MAX) ROW AND APPLY THE MAX(SALARY) FUNCTION. THAT’S WHY WE GOT SAME REPEATED VALUES IN MAX\_SALARY

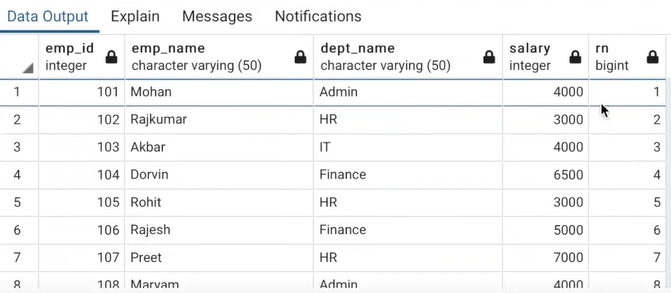
MAXIMUM SALARY CORRESPONDING TO EACH DEPARTMENT, THEREFORE:

SELECT E.\*, MAX(SALARY) OVER(PARTITION BY DEPT\_NAME) AS MAX\_SALARY FROM EMPLOYEE E;



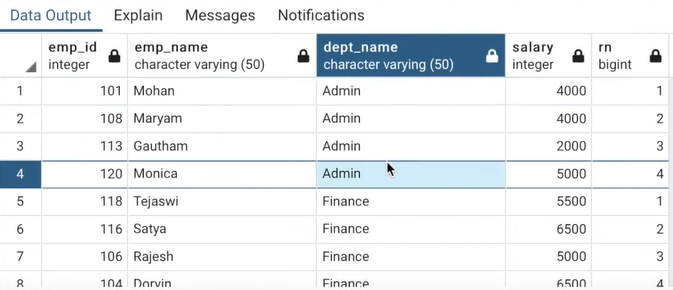
Q-> ASSIGN ROW NUMBER(S.NO) TO EACH ROW OF TABLE

SELECT E.\*, ROW\_NUMBER() OVER() AS RN FROM EMPLOYEE E:



Q-> ASSIGN ROW NUMBER TO EACH ROW , DEPARTMENT WISE?

SELECT E.\*, ROW\_NUMBER() OVER(PARTITION BY DEPT\_NAME) AS RN FROM EMPLOYEE E;

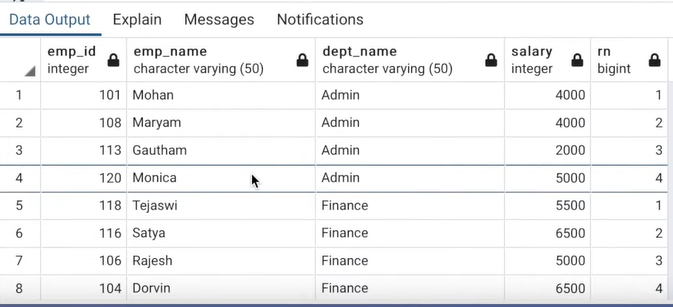


* For each dept, the row number is reset

Q-> FETCH THE RECORD FOR FIRST TWO EMPLOYEES THAT JOINED THE COMPANY IN EACH DEPT.(assume emp id of employee who joined previously(old) is less than newbies)

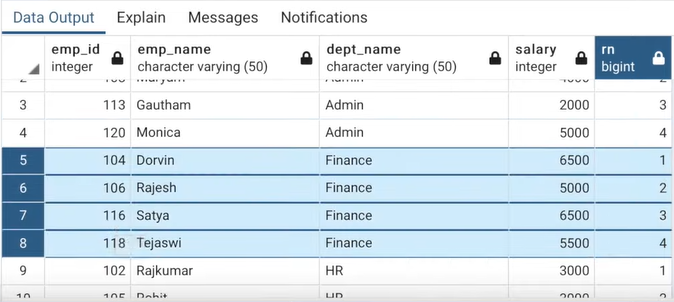
PROCEDURE->

SELECT E.\*, ROW\_ NUMBER () OVER(PARTITION BY DEPT\_NAME) AS RN FROM EMPLOYEE E;



The id’s are not sorted yet, so-

SELECT E.\*, ROW\_NUMBER() OVER(PARTITION BY DEPT\_NAME ORDER BY EMP\_ID) AS RN FROM EMPLOYEE E;

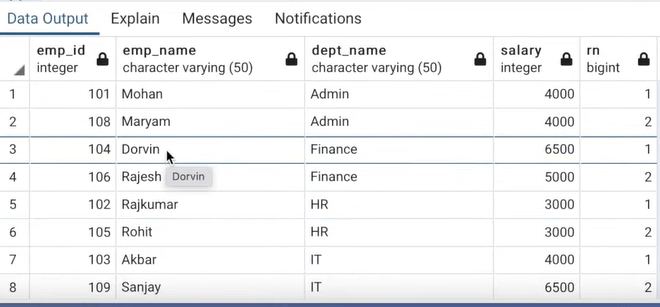


Now I want only top two employees who are oldest:

SELECT \* FROM (

(SELECT E.\*, ROW\_NUMBER() OVER( PARTITION BY DEPT\_NAME ORDER BY EMP\_ID ) AS RN FROM EMPLOYEE E) X

WHERE X.RN<=2;



Therefore, only 8 rows are there.

Q-> FETCH THE DETAILS OF TOP 3 EMPLOYEE EARNING THE MAX SALARY?

SELECT \* FROM(

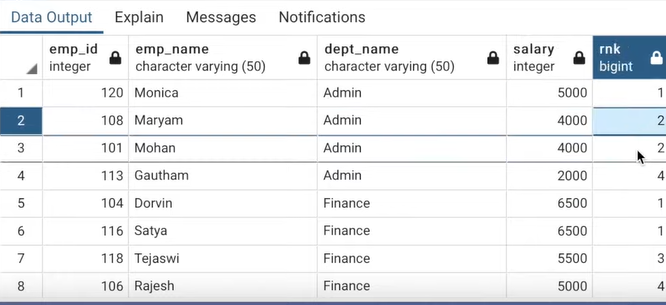
SELECT E.\*, ROW\_NUMBER () OVER( PARTITION BY DEPT\_NAME ORDER BY SALARY ASC ) AS RN FROM EMPLOYEE E)X

WHERE X.RN<=3;

OR-

PROCEDURE-

SELECT E.\*, RANK() OVER(PARTITION BY DEPT\_NAME ORDER BY SALARY DESC) AS RNK FROM EMPLOYEE E;

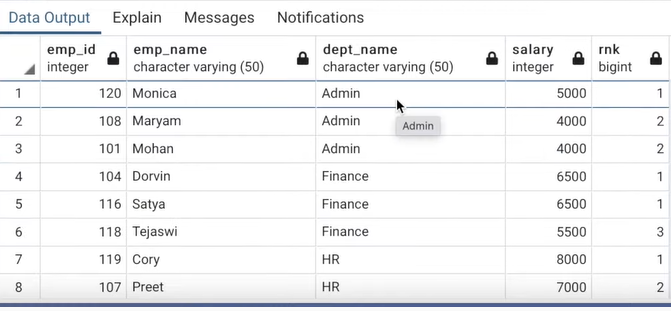
rank assignes same rank to duplicate values(4000 me 2) and then resumes with the original indexing

NOW-

SELECT \* FROM(

SELECT E.\*, RANK() OVER(PARTITION BY DEPT\_NAME ORDER BY SALARY DESC) AS RNK FROM EMPLOYEE E)X

WHERE X.RNK<=3;



RANK() SKIPS THE INDEX WHEN PREV TUPLE HAD A DUPLICATE VALUES,

DENSE\_RANK() DOES NOT SKIP THE INDEX WHEN PREV TUPLE HAD A DUPLICATE VALUES,

RANK() INDEXES AS PER CONDITION EX-

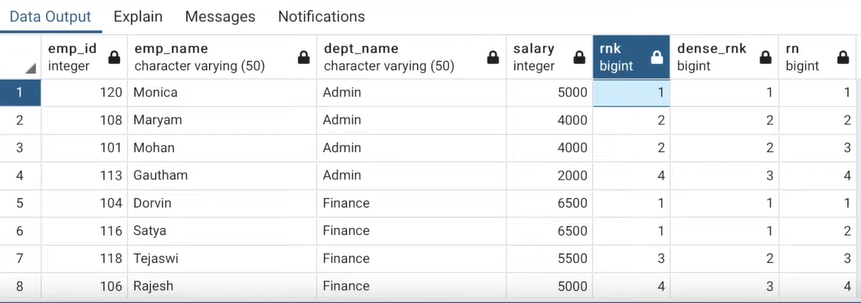
SELECT E.\*,

RANK() OVER(PARTITION BY DEPT\_NAME ORDER BY SALARY DESC) AS RNK,

DENSE\_RANK() OVER(PARTITION BY DEPT\_NAME ORDER BY SALARY DESC) AS DENSE\_RNK,

ROW\_NUMBER() OVER(PARTITION BY DEPT\_NAME ORDER BY SALARY DESC) AS RN

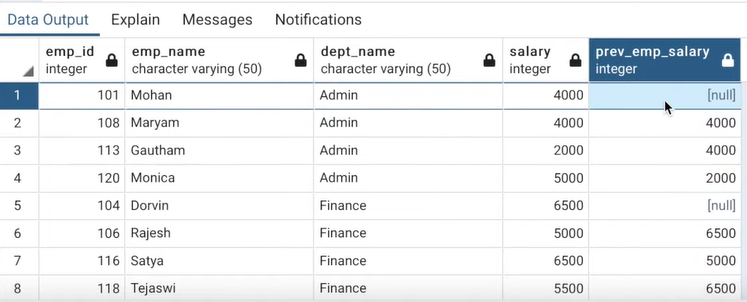
FROM EMPLOYEE ;



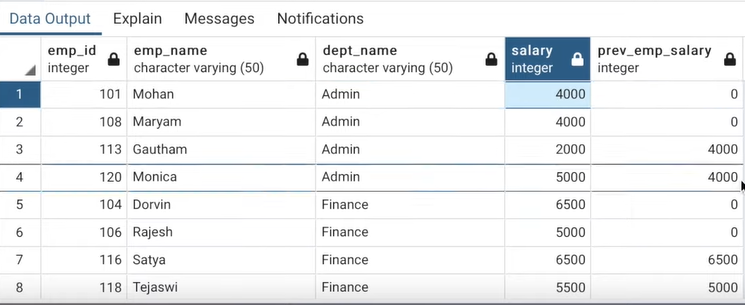
Q-> FETCH THE COL WITH THE PREV RECORD OF EACH ROW?

LAG(COL\_NAME) FETCHES THE VALUES OF PREV RECORDS, HERE 1ST ROW THUS WILL HAVE NULL VALUE QKI 1ST ROW KI KOI PREV ROW HE HI NHI

SYNTAX- LAG(COL\_NAME, N,M) N-> nth NUM OF RECORD PRIOR TO CURRUNT ROW, M-> THE VALUE TO SET IF NO RECORD ARE FOUND

SELECT \* ,LAG(SALARY) OVER(PARTITION BY DEPT\_NAME ORDER BY DEPT\_ID) AS PREV\_EMP\_SALRY FROM EMPLOYEE;

SELECT \* ,LAG(SALARY,2,0) OVER(PARTITION BY DEPT\_NAME ORDER BY DEPT\_ID) AS PREV\_EMP\_SALRY FROM EMPLOYEE;

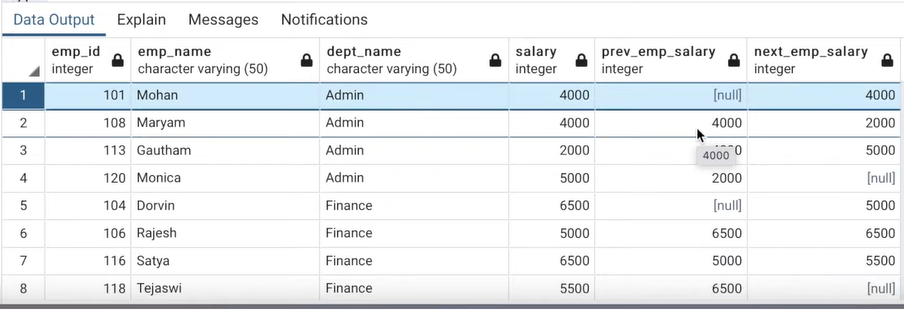


SELECT \* ,

LAG(SALARY) OVER(PARTITION BY DEPT\_NAME ORDER BY DEPT\_ID) AS PREV\_EMP\_SALRY,

LEAD(SALARY) OVER(PARTITION BY DEPT\_NAME ORDER BY DEPT\_ID) AS NEXT\_EMP\_SALRY

FROM EMPLOYEE;



Q->FETCH A QUERY TO DISPLAY IF THE SALARY OF AN EMPLOYEE IS HIGHER/EQUAL OR LOWER THAN PREV EMPLOYEE?

SELECT \* FROM(

SELECT \* ,

LAG(SALARY) OVER(PARTITION BY DEPT\_NAME ORDER BY DEPT\_ID) AS PREV\_EMP\_SALRY,

LEAD(SALARY) OVER(PARTITION BY DEPT\_NAME ORDER BY DEPT\_ID) AS NEXT\_EMP\_SALRY

FROM EMPLOYEE)X WHERE

CASE

WHEN X.PREV\_EMP\_SALARY>SALARY THEN “HIGHER”

WHEN X.PREV\_EMP\_SALARY=SALARY THEN “EQUAL”

ELSE “LOWE”

END STATUS