DATA AND
ARTIFICIAL INTELLIGENCE

Window Functions in SQL

Learning Objectives

By the end of this lesson, you will be able to:

- Explain window functions and various clauses
- List the aggregate window functions
- Classify ranking window functions
- Categorize the miscellaneous window functions

Introduction to Window Functions

Window Functions

The window functions is like an SQL function that takes input values from a **window** of one or more rows of a SELECT statement’s result set.

The window functions perform various operations on a group of rows and provide an aggregated value for each row with a unique identity.

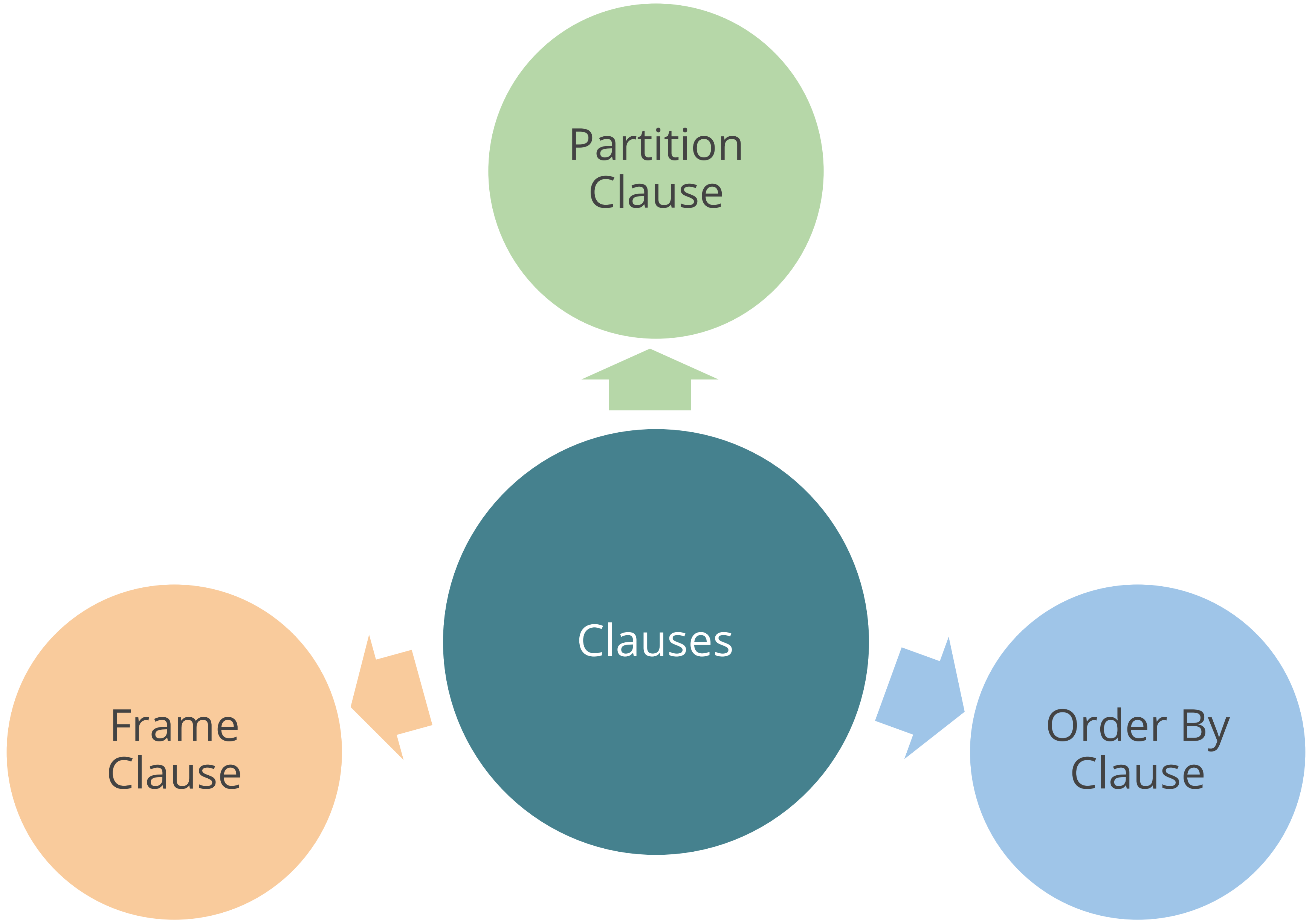
General Syntax

Syntax

```
window_function_name(expression)OVER([partition_definition]
[order_definition] [frame_definition] )
```

- Window function specifies the window function name with an expression.
- OVER clause can comprise partition definition, order definition, and frame definition.

Types of Clauses



Partition Clause

The partition clause is used to divide or split the rows into partitions, and the partition boundary is used to split two partitions.

Syntax

```
 PARTITION **BY**
<expression>[{,<expression>...}]
```

Order By Clause

Order By clause is an arrangement of rows inside a partition. It performs partitions using multiple keys where each key has an expression.

Syntax

```
ORDER BY <expression> [ASC|DESC],
[{,<expression>...}]
```

Frame Clause

Frame clause is defined as subset of the current position. It allows to move the subset within a partition based on the position of the current row in its partition.

Syntax

```
 frame_unit

{<frame_start>|<frame_between>}
```

The frame unit can be a row or range that specifies the kind of relationship between the current row and frame row.

Frame Clause

Keywords	Meaning
Frame unit	Rows: It assigns row number for offset of current and frame row.
Frame unit	Range: It assigns row values for offset of current and frame row.
Frame start	Frame start specifies the frame boundary.
Frame between	Frame between specifies the frame boundary.

Use Case for Window Functions

Problem Scenario:

The HR of a company wants to calculate the performance of employees department-wise based on the employee ratings.

Objective:

You are required to retrieve the employee ID, first name, role, department, and employee rating by calculating the maximum employee rating using PARTITION BY and MAX function on department and employee rating fields respectively.

Instructions:

Refer to the employee dataset given in the course resource section in LMS and create an employee table using fields mentioned in dataset. Insert the values accordingly to perform the above objectives.

Use Case for Window Functions

Field Name	Description
EMP_ID	Employee ID
FIRST_NAME	First name of the employee
LAST_NAME	Last name of the employee
GENDER	Gender of the employee (M/F)
ROLE	Designation of the employee (Junior, Senior, Lead, and Associate Data Scientist)
DEPT	Name of the department (Retail, Finance, Automotive, and Healthcare)

Use Case for Window Functions

Field Name	Description
EXP	Experience of the employee
COUNTRY	Country where the employee lives
CONTINENT	Continent based on the country
SALARY	Salary of the employee per month in dollars
EMP_RATING	Rating for the employee (1: Not achieved any goals, 2: Below expectation, 3: Meeting expectation, 4: Excellent performance, 5: Overachiever)
MANAGER_ID	Employee ID for the manager

Use Case for Window Functions

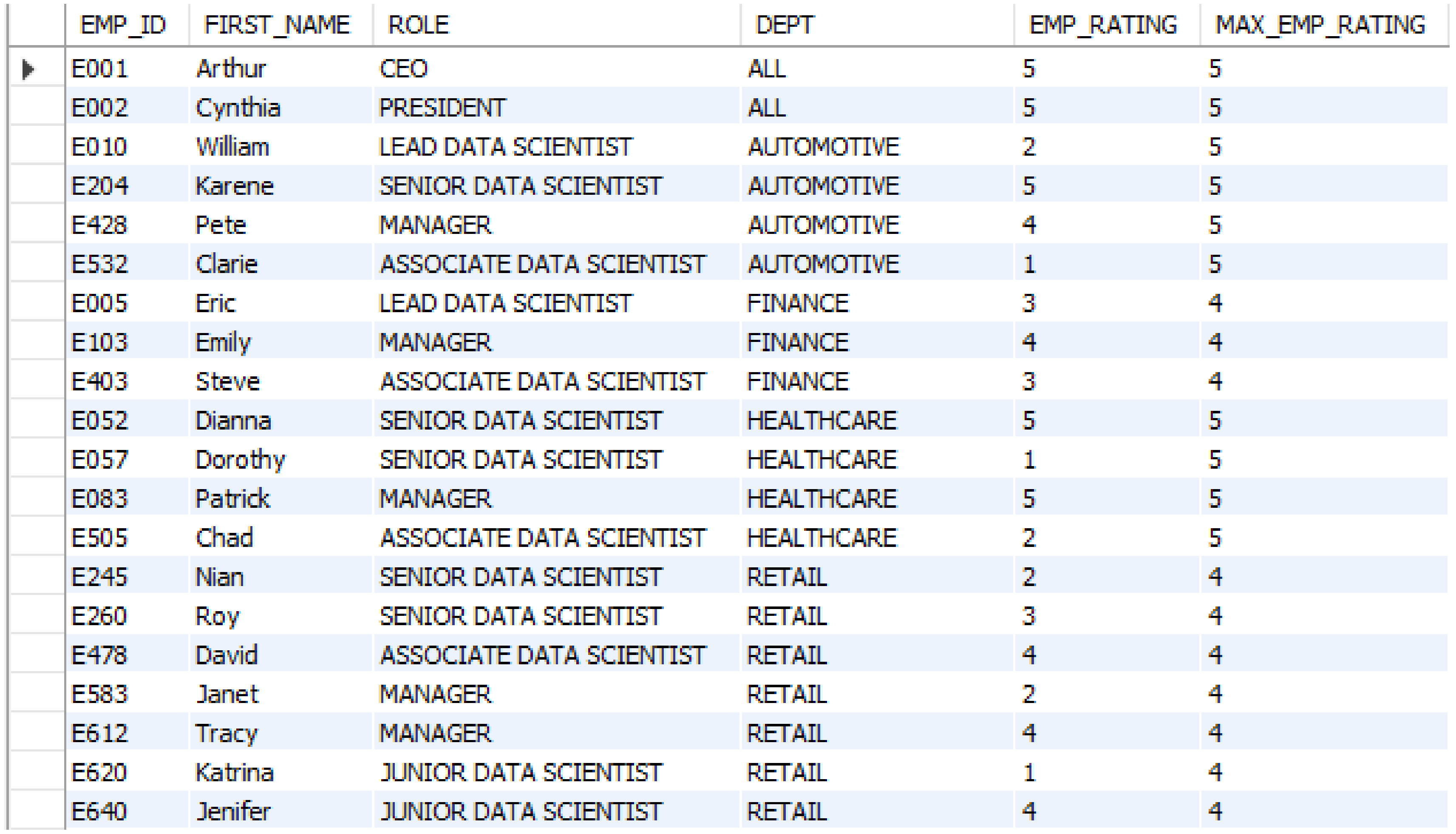
Solution:

```
# SELECT EMP_ID, FIRST_NAME,ROLE,DEPT,EMP_RATING and calculate the maximum EMP_RATING
in a department from the employee table using partition clause on department, Max
function.



SELECT EMP_ID,FIRST_NAME,ROLE,DEPT,EMP_RATING,MAX(EMP_RATING)OVER(PARTITION BY DEPT) AS
MAX_EMP_RATING FROM emp_table;
```

By executing this query, the HR can identify the maximum rating of the employee in a department.

Use Case for Window Functions

Output:



EMP_ID	FIRST_NAME	ROLE	DEPT	EMP_RATING	MAX_EMP_RATING
E001	Arthur	CEO	ALL	5	5
E002	Cynthia	PRESIDENT	ALL	5	5
E010	William	LEAD DATA SCIENTIST	AUTOMOTIVE	2	5
E204	Karene	SENIOR DATA SCIENTIST	AUTOMOTIVE	5	5
E428	Pete	MANAGER	AUTOMOTIVE	4	5
E532	Clarie	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	1	5
E005	Eric	LEAD DATA SCIENTIST	FINANCE	3	4
E103	Emily	MANAGER	FINANCE	4	4
E403	Steve	ASSOCIATE DATA SCIENTIST	FINANCE	3	4
E052	Dianna	SENIOR DATA SCIENTIST	HEALTHCARE	5	5
E057	Dorothy	SENIOR DATA SCIENTIST	HEALTHCARE	1	5
E083	Patrick	MANAGER	HEALTHCARE	5	5
E505	Chad	ASSOCIATE DATA SCIENTIST	HEALTHCARE	2	5
E245	Nian	SENIOR DATA SCIENTIST	RETAIL	2	4
E260	Roy	SENIOR DATA SCIENTIST	RETAIL	3	4
E478	David	ASSOCIATE DATA SCIENTIST	RETAIL	4	4
E583	Janet	MANAGER	RETAIL	2	4
E612	Tracy	MANAGER	RETAIL	4	4
E620	Katrina	JUNIOR DATA SCIENTIST	RETAIL	1	4
E640	Jenifer	JUNIOR DATA SCIENTIST	RETAIL	4	4

Aggregate Window Functions

Aggregate Window Functions

The aggregate window functions perform on a particular set of rows and provide the result in a single row.

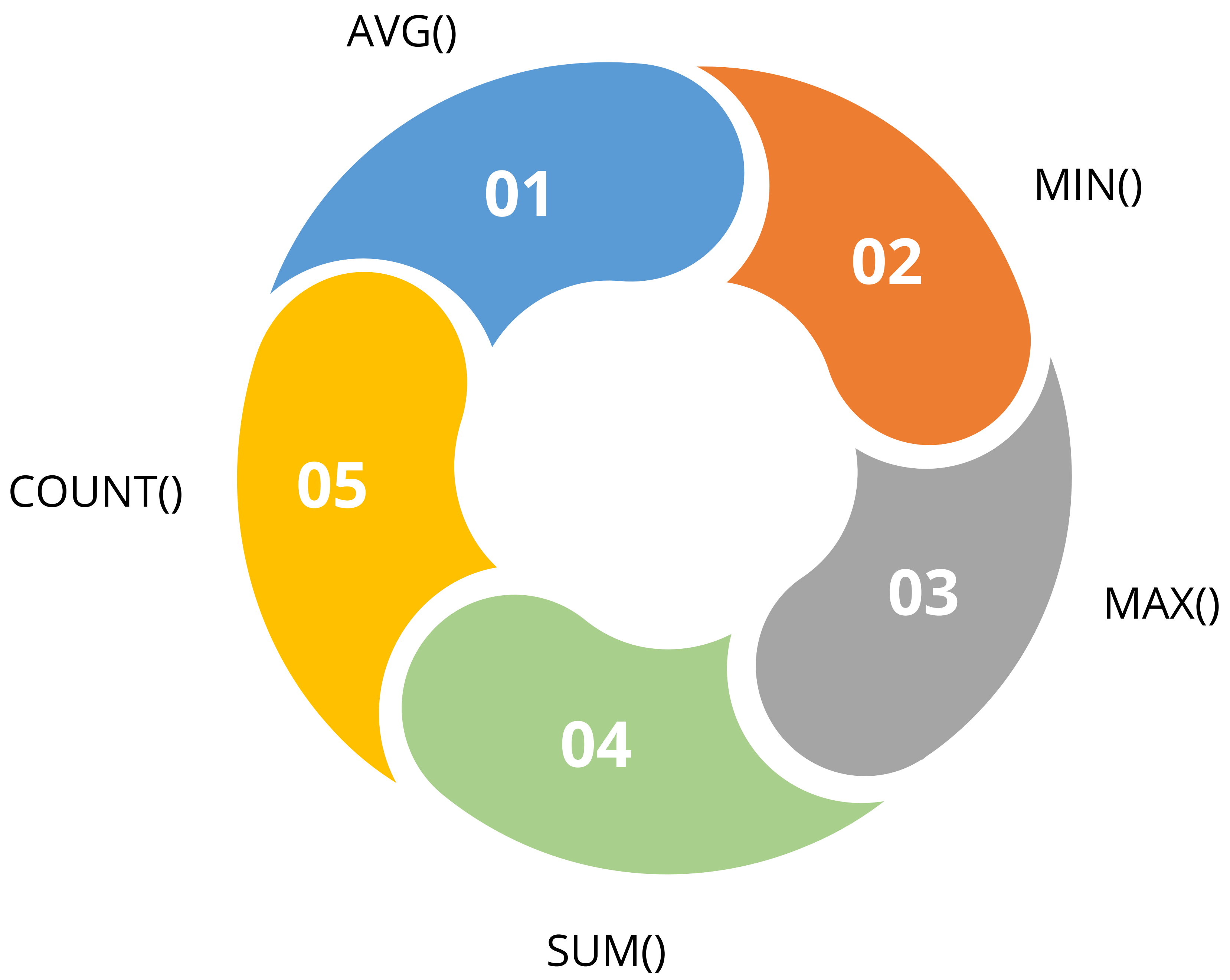
Syntax

```
window_function ( [ ALL ] expression )
   OVER ( [ PARTITION BY expr_list ] [
ORDER BY order_list frame_clause ] )
```

Arguments in Aggregate Window Functions

Keywords	Meaning
Window function	It can be any aggregate window function.
ALL	ALL helps to maintain all duplicate values from the expression.
OVER	It distinguishes window aggregation from general aggregation functions.
PARTITION BY	PARTITION BY provides a window if there are one or more expressions.
ORDER BY	ORDER BY is used to sort the rows within each partition.

Types of Aggregate Window Functions



Types of Aggregate Window Functions

Window Function	Argument Type	Return Type	Description
MIN()	BINARY, DECIMAL, VARCHAR, DATE, TIME, or TIMESTAMP	Same as argument type	It returns the minimum value of the expression across all input values.
MAX()	BINARY, DECIMAL, VARCHAR, DATE, TIME, or TIMESTAMP	Same as argument type	It returns the maximum value of the expression across all input values.

Types of Aggregate Window Functions

Window Function	Argument Type	Return Type	Description
AVG()	SMALLINT, INTEGER, BIGINT, FLOAT, DOUBLE, DECIMAL, INTERVALYEAR or INTERVALDAY	DECIMAL argument: returns in decimal Other types: double	It returns average value for the input expression values.
COUNT()	All argument data types	BIGINT	It counts the number of input rows.

Types of Aggregate Window Functions

Window Function	Argument Type	Return Type	Description
SUM()	SMALLINT, INTEGER, BIGINT, FLOAT, DOUBLE, DECIMAL, INTERVALDAY, or INTERVALYEAR	DECIMAL argument: returns in decimal Float: returns in double Other types : BIGINT	It returns the sum of the expression across all input values.

Use Case for MIN and MAX

Problem Scenario:

The HR of a company wants to identify the minimum and the maximum salary of the employees in a role.

Objective:

You are required to display the employee’s ID, first name, role, and salary by finding the minimum and maximum salary of the employees using PARTITION BY clause, MIN, and MAX functions on role and salary fields respectively.

Instructions:

Refer to the employee table which is created and perform the above objectives.

Use Case for MIN and MAX

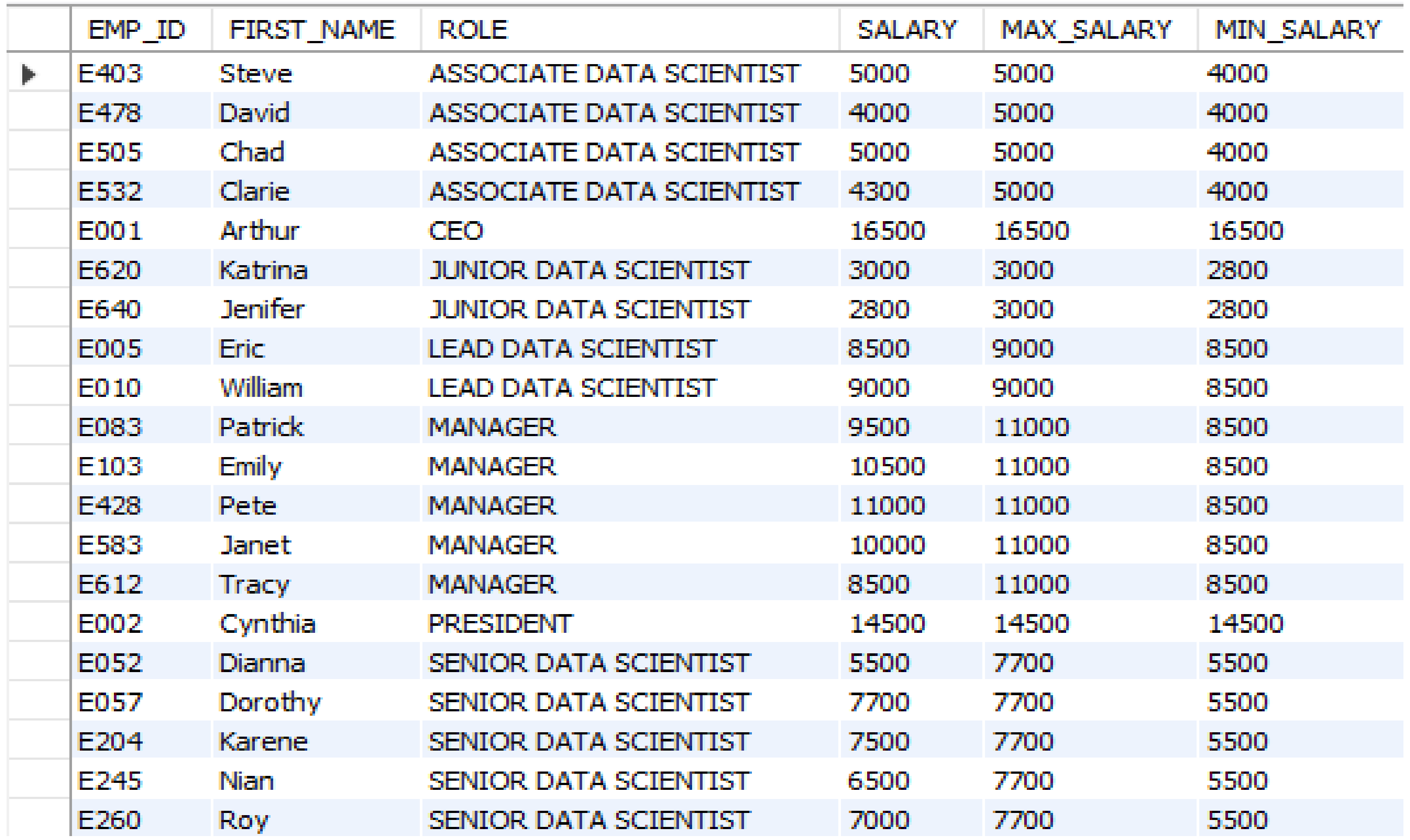
Solution:

```
# SELECT EMP_ID, FIRST_NAME, ROLE, SALARY and calculate minimum, maximum salary of the
employees using PARTITION CLAUSE on the role field, MIN , MAX function.



SELECT     EMP_ID,FIRST_NAME,ROLE,SALARY, MAX(SALARY) OVER (PARTITION BY ROLE)
MAX_SALARY,MIN(SALARY) OVER (PARTITION BY ROLE) MIN_SALARY FROM emp_table
```

By executing this query, the HR can identify the maximum and the minimum salary in a role.

Use Case for MIN and MAX

Output:



EMP_ID	FIRST_NAME	ROLE	SALARY	MAX_SALARY	MIN_SALARY
E403	Steve	ASSOCIATE DATA SCIENTIST	5000	5000	4000
E478	David	ASSOCIATE DATA SCIENTIST	4000	5000	4000
E505	Chad	ASSOCIATE DATA SCIENTIST	5000	5000	4000
E532	Clarie	ASSOCIATE DATA SCIENTIST	4300	5000	4000
E001	Arthur	CEO	16500	16500	16500
E620	Katrina	JUNIOR DATA SCIENTIST	3000	3000	2800
E640	Jenifer	JUNIOR DATA SCIENTIST	2800	3000	2800
E005	Eric	LEAD DATA SCIENTIST	8500	9000	8500
E010	William	LEAD DATA SCIENTIST	9000	9000	8500
E083	Patrick	MANAGER	9500	11000	8500
E103	Emily	MANAGER	10500	11000	8500
E428	Pete	MANAGER	11000	11000	8500
E583	Janet	MANAGER	10000	11000	8500
E612	Tracy	MANAGER	8500	11000	8500
E002	Cynthia	PRESIDENT	14500	14500	14500
E052	Dianna	SENIOR DATA SCIENTIST	5500	7700	5500
E057	Dorothy	SENIOR DATA SCIENTIST	7700	7700	5500
E204	Karene	SENIOR DATA SCIENTIST	7500	7700	5500
E245	Nian	SENIOR DATA SCIENTIST	6500	7700	5500
E260	Roy	SENIOR DATA SCIENTIST	7000	7700	5500

Use Case for AVG and COUNT

Problem Scenario:

The HR of a company wants to identify the average performance of the employee's department-wise and also find the total number of records in a department.

Objective:

You are required to display the employee’s ID, first name, department, and employee rating by calculating the average employee rating and the total number of records in a department using PARTITION BY clause, AVG, and COUNT functions on department and employee rating fields respectively.

Instructions:

Refer to the employee table which is created and perform the above objectives.

Use Case for AVG and COUNT

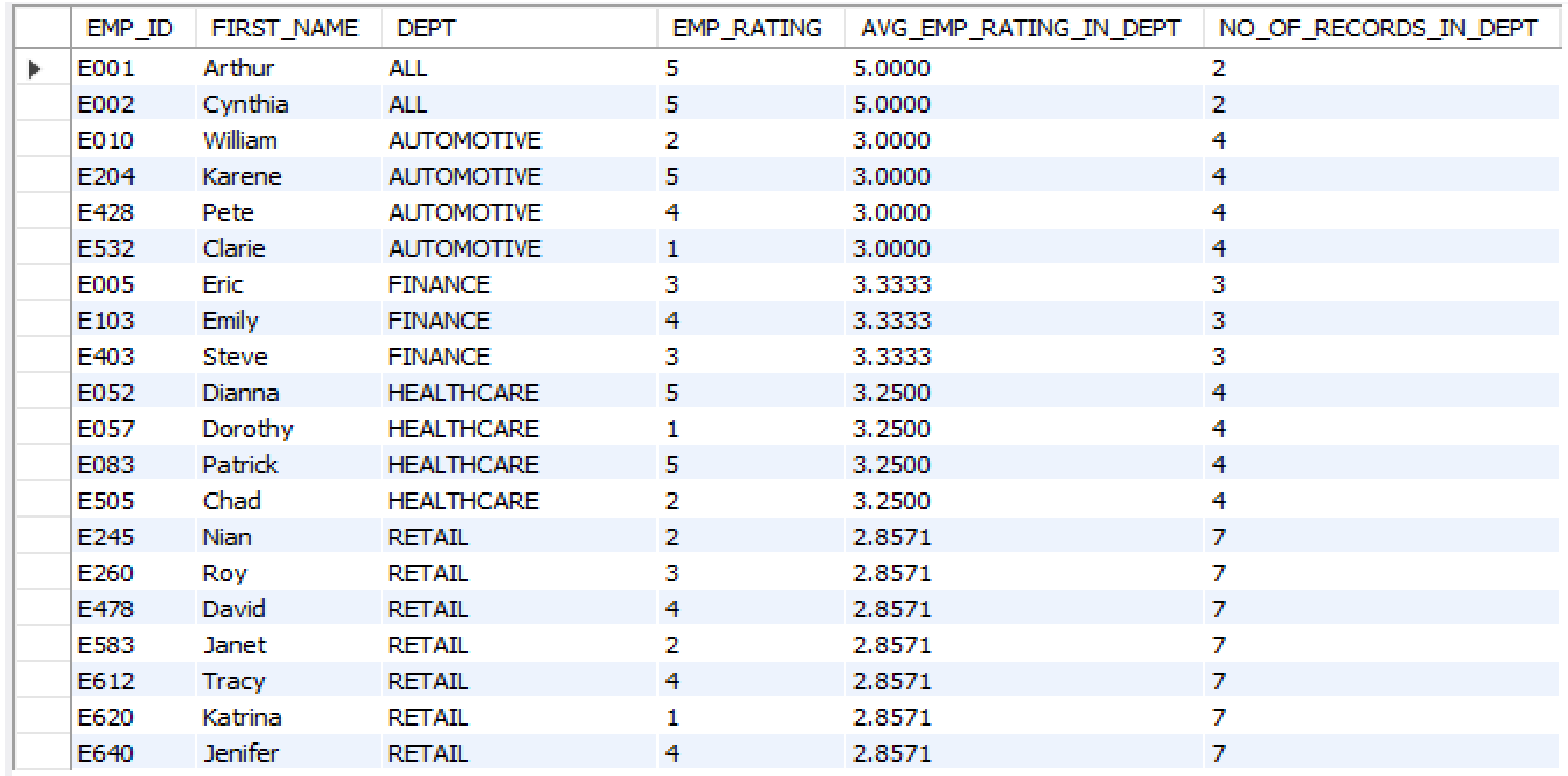
Solution

```
# SELECT EMP_ID, FIRST_NAME,DEPT,EMP_RATING and calculate the average employee rating
and total no of records in a department using AVG,COUNT function.



SELECT   EMP_ID,FIRST_NAME,DEPT,EMP_RATING,  AVG(EMP_RATING)OVER(PARTITION BY DEPT)
AVG_EMP_RATING_IN_DEPT,  COUNT(*) OVER(PARTITION BY DEPT) NO_OF_RECORDS_IN_DEPT FROM
emp_table
```

By executing this query, the HR can identify the average performance of the department and the total number of records in a department.

Use Case for AVG and COUNT

Output:



EMP_ID	FIRST_NAME	DEPT	EMP_RATING	AVG_EMP_RATING_IN_DEPT	NO_OF_RECORDS_IN_DEPT
E001	Arthur	ALL	5	5.0000	2
E002	Cynthia	ALL	5	5.0000	2
E010	William	AUTOMOTIVE	2	3.0000	4
E204	Karene	AUTOMOTIVE	5	3.0000	4
E428	Pete	AUTOMOTIVE	4	3.0000	4
E532	Clarie	AUTOMOTIVE	1	3.0000	4
E005	Eric	FINANCE	3	3.3333	3
E103	Emily	FINANCE	4	3.3333	3
E403	Steve	FINANCE	3	3.3333	3
E052	Dianna	HEALTHCARE	5	3.2500	4
E057	Dorothy	HEALTHCARE	1	3.2500	4
E083	Patrick	HEALTHCARE	5	3.2500	4
E505	Chad	HEALTHCARE	2	3.2500	4
E245	Nian	RETAIL	2	2.8571	7
E260	Roy	RETAIL	3	2.8571	7
E478	David	RETAIL	4	2.8571	7
E583	Janet	RETAIL	2	2.8571	7
E612	Tracy	RETAIL	4	2.8571	7
E620	Katrina	RETAIL	1	2.8571	7
E640	Jenifer	RETAIL	4	2.8571	7

Use Case for SUM

Problem Scenario:

The HR of a company wants to calculate the total employee rating in a department.

Objective:

You are required to display the employee’s Id, first name, department, and employee rating by calculating the total employee rating in a department using PARTITION BY clause and SUM function on the department and the employee rating fields respectively.

Instructions:

Refer to the employee table which is created and perform the above objectives.

Use Case for SUM

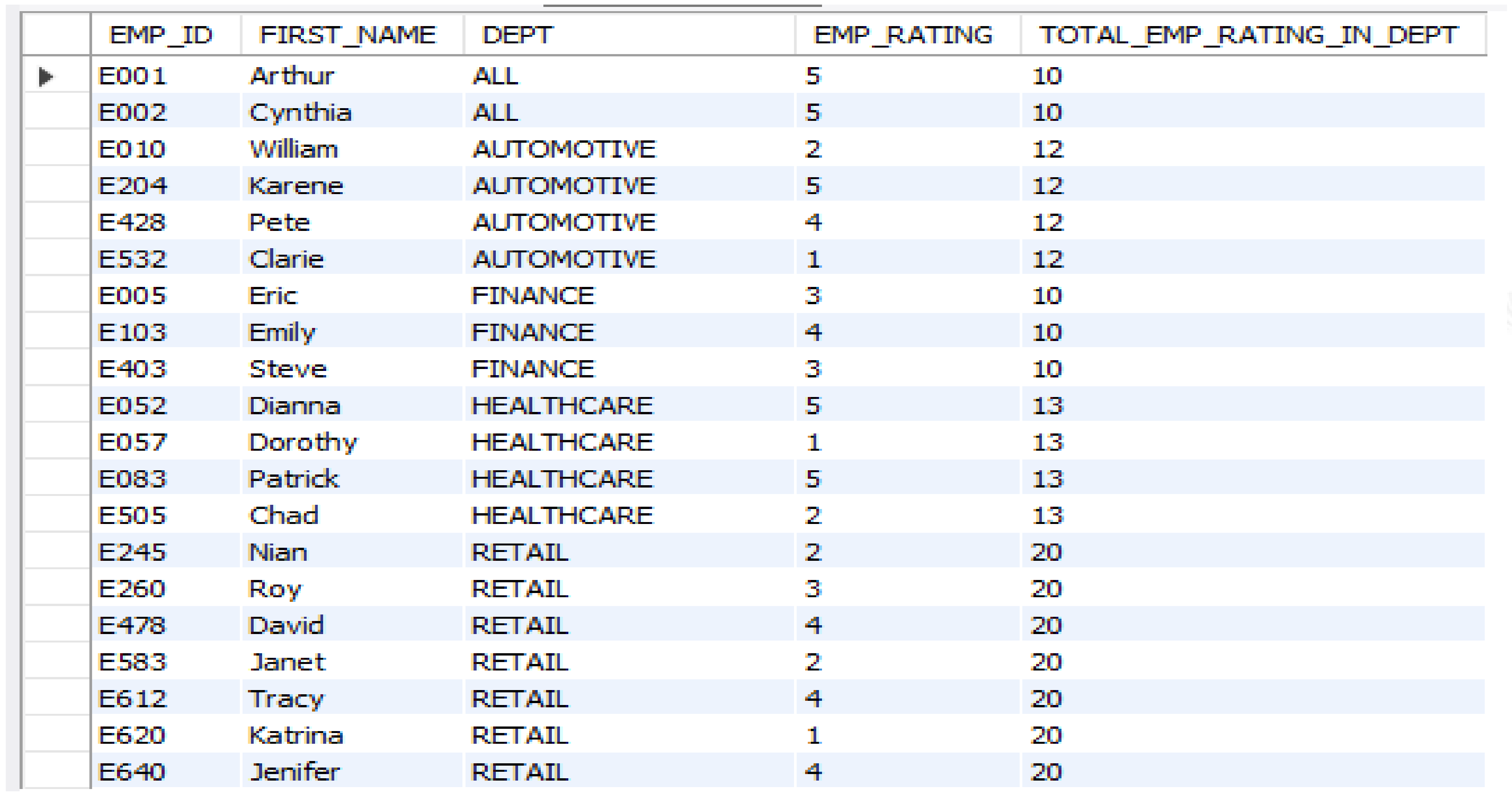
Solution:

```
# SELECT EMP_ID, FIRST_NAME, DEPT,EMP RATING and calculate the total employee rating in a
department using PARTITION CLAUSE on a dept and SUM function.



SELECT   EMP_ID,FIRST_NAME,DEPT,EMP_RATING,  SUM(EMP_RATING) OVER (PARTITION BY DEPT)
TOTAL_EMP_RATING_IN_DEPT FROM emp_table
```

By executing this query, the HR can identify the total employee rating in a department .

Use Case for SUM

Output:



EMP_ID	FIRST_NAME	DEPT	EMP_RATING	TOTAL_EMP_RATING_IN_DEPT
E001	Arthur	ALL	5	10
E002	Cynthia	ALL	5	10
E010	William	AUTOMOTIVE	2	12
E204	Karene	AUTOMOTIVE	5	12
E428	Pete	AUTOMOTIVE	4	12
E532	Clarie	AUTOMOTIVE	1	12
E005	Eric	FINANCE	3	10
E103	Emily	FINANCE	4	10
E403	Steve	FINANCE	3	10
E052	Dianna	HEALTHCARE	5	13
E057	Dorothy	HEALTHCARE	1	13
E083	Patrick	HEALTHCARE	5	13
E505	Chad	HEALTHCARE	2	13
E245	Nian	RETAIL	2	20
E260	Roy	RETAIL	3	20
E478	David	RETAIL	4	20
E583	Janet	RETAIL	2	20
E612	Tracy	RETAIL	4	20
E620	Katrina	RETAIL	1	20
E640	Jenifer	RETAIL	4	20

Assisted Practice: Aggregate Window Functions

Duration: 20 min

Problem Statement: You are required to calculate the total, average, maximum, and minimum salary of the employee by grouping the departments from the employee table.

Assisted Practice: Aggregate Window Functions

Steps to be performed:

Step 1: Creating the employee table and inserting values in it:

CREATE

```
CREATE TABLE lep_7.employee ( emp_id int NOT NULL,    f_name varchar(45) NULL,
l_name varchar(45) NOT NULL,    job_id varchar(45) NOT NULL,    salary
decimal(8,2) NOT NULL,    manager_id int NOT NULL,    dept_id varchar(45) NOT
NULL,    PRIMARY KEY(emp_id));
```

INSERT

```
INSERT INTO lep_7. employee
(emp_id,f_name,l_name,job_id,salary,manager_id,dept_id) VALUES
('103','krishna','gee','125','500000','05','44');
```

Assisted Practice: Aggregate Window Functions

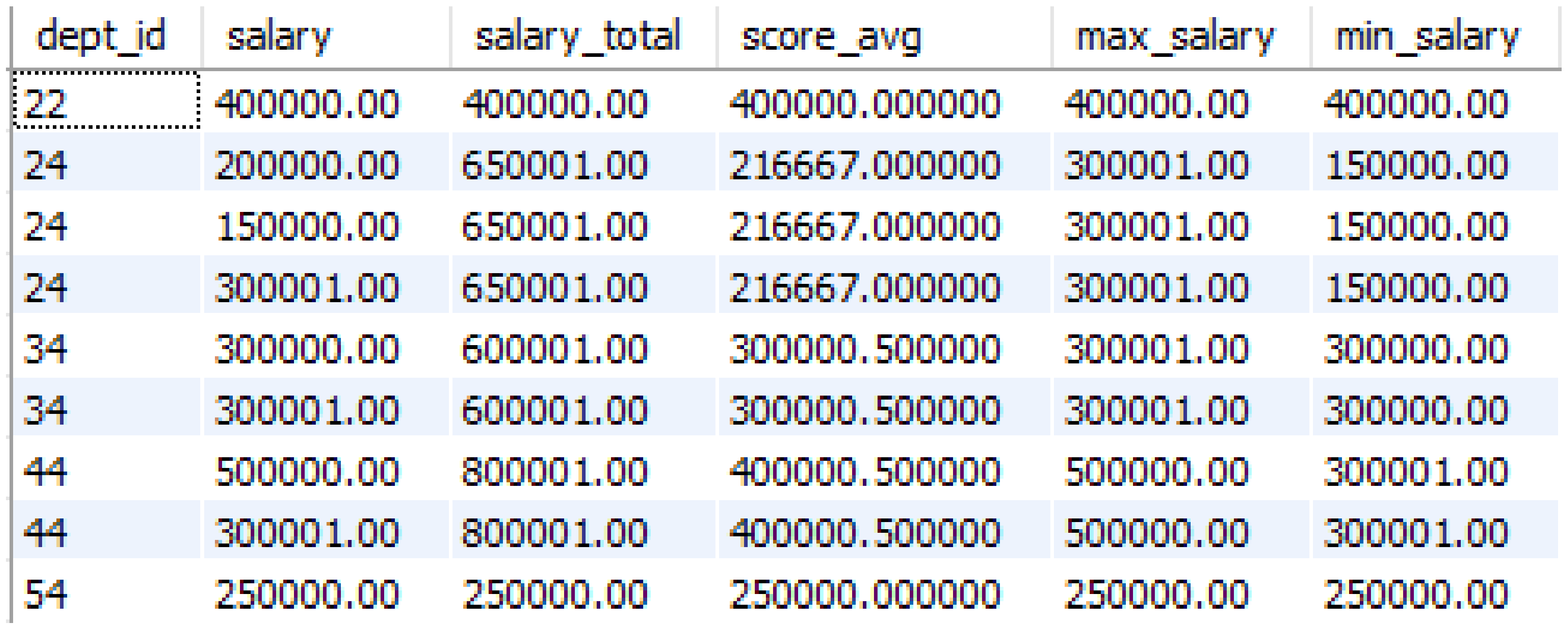
Step 2: Querying to perform aggregate window functions:

QUERY

```
SELECT dept_id,salary,SUM(salary) OVER (PARTITION BY dept_id)  salary_total,
AVG(salary) OVER(PARTITION BY dept_id) score_avg ,          MAX(salary) OVER
(PARTITION BY dept_id) max_salary ,           MIN(salary) OVER (PARTITION BY dept_id)
min_salary         FROM employee
```

Assisted Practice: Aggregate Window Functions

Output:

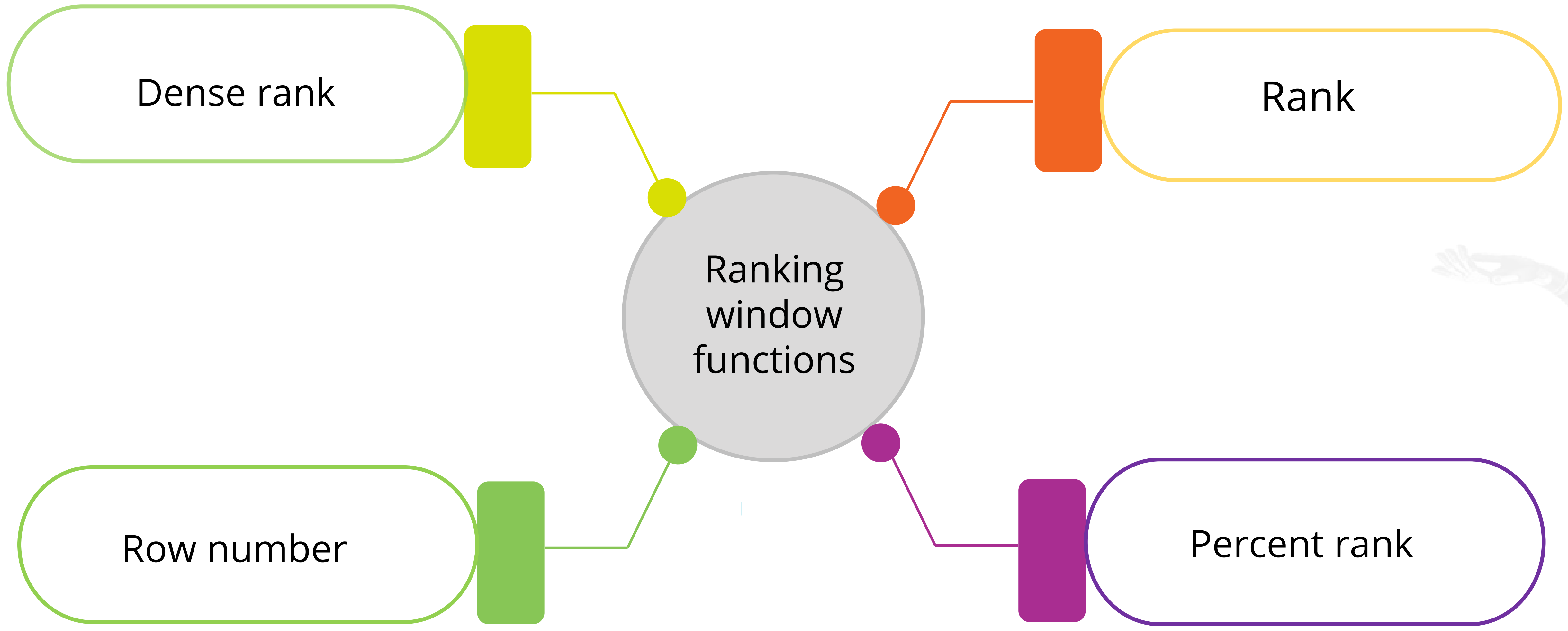


dept_id	salary	salary_total	score_avg	max_salary	min_salary
22	400000.00	400000.00	400000.000000	400000.00	400000.00
24	200000.00	650001.00	216667.000000	300001.00	150000.00
24	150000.00	650001.00	216667.000000	300001.00	150000.00
24	300001.00	650001.00	216667.000000	300001.00	150000.00
34	300000.00	600001.00	300000.500000	300001.00	300000.00
34	300001.00	600001.00	300000.500000	300001.00	300000.00
44	500000.00	800001.00	400000.500000	500000.00	300001.00
44	300001.00	800001.00	400000.500000	500000.00	300001.00
54	250000.00	250000.00	250000.000000	250000.00	250000.00

Ranking Window Functions

Ranking Window Functions and Its Types

Ranking window functions specify the rank for individual fields as per the categorization.



Dense Rank

Definition

- It assigns a rank to every row in a partition based on the ORDER BY clause.
- It assigns the same rank for equal values.
- It has no gaps if two or more rows have a similar rank.

Syntax

```
DENSE_RANK() OVER (

        PARTITION BY
<expression>[{,<expression>...}]

        ORDER BY <expression>
[ASC|DESC], [{,<expression>...}])
```

Rank

Definition

- Rank helps to assign a rank to all rows within every partition.
- The first row of the rank will be 1.
- Same rank for the same value.
- There will be a gap if two or more rows have the same rank.

Syntax

```
RANK() OVER (
               PARTITION BY
<expr1>[{,<expr2>...}]
               ORDER BY <expr1>
[ASC|DESC], [{,<expr2>...}]
)
```

Use Case for Rank and Dense Rank

Problem Scenario:

The HR of a company wants to assign a rank for each employee based on their employee rating.

Objective:

You are required to display the employee's ID, first name, department, and employee rating by assigning a rank to all the employees based on their employee rating using ORDER BY clause, RANK, and DENSE RANK functions on the employee rating field.

Instructions:

Refer to the employee table which is created and perform the above objectives.

Use Case for Rank and Dense Rank

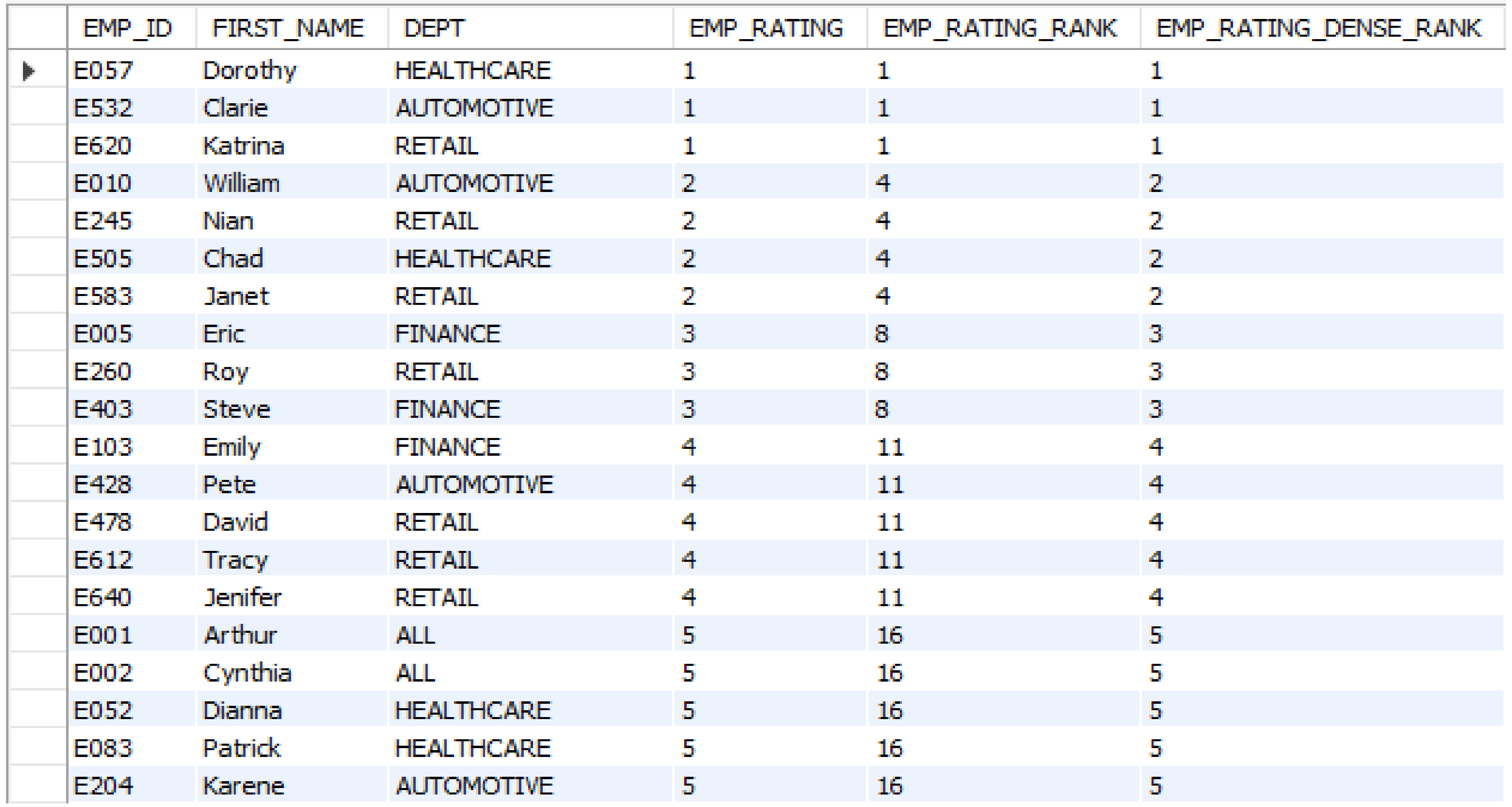
Solution:

```
# SELECT EMP_ID, FIRST_NAME, DEPT,EMP RATING and assign a rank to all the employee
based on their employee rating using Rank and Dense Rank .



SELECT    EMP_ID,FIRST_NAME,DEPT,EMP_RATING,    RANK() OVER (ORDER BY EMP_RATING)
EMP_RATING_RANK,    DENSE_RANK() OVER (ORDER BY EMP_RATING) EMP_RATING_DENSE_RANK FROM
emp_table;
```

By executing this query, the HR can identify the rank of the employees based on their employee rating.

Use Case for Rank and Dense Rank

Output:



EMP_ID	FIRST_NAME	DEPT	EMP_RATING	EMP_RATING_RANK	EMP_RATING_DENSE_RANK
E057	Dorothy	HEALTHCARE	1	1	1
E532	Clarie	AUTOMOTIVE	1	1	1
E620	Katrina	RETAIL	1	1	1
E010	William	AUTOMOTIVE	2	4	2
E245	Nian	RETAIL	2	4	2
E505	Chad	HEALTHCARE	2	4	2
E583	Janet	RETAIL	2	4	2
E005	Eric	FINANCE	3	8	3
E260	Roy	RETAIL	3	8	3
E403	Steve	FINANCE	3	8	3
E103	Emily	FINANCE	4	11	4
E428	Pete	AUTOMOTIVE	4	11	4
E478	David	RETAIL	4	11	4
E612	Tracy	RETAIL	4	11	4
E640	Jenifer	RETAIL	4	11	4
E001	Arthur	ALL	5	16	5
E002	Cynthia	ALL	5	16	5
E052	Dianna	HEALTHCARE	5	16	5
E083	Patrick	HEALTHCARE	5	16	5
E204	Karene	AUTOMOTIVE	5	16	5

Row Number

Definition

- Row number retrieves the unique sequential number of each row for the specified data.
- Similar values will have different ranks.

Syntax

```
ROW_NUMBER() OVER
(<partition_definition>
<order_definition>)
```

Use Case for Row Number

Problem Scenario:

The IT department of a company wants to assign an asset number for each employee based on their employee ID in ascending order.

Objective:

You are required to display the employee's ID, first name, role, and department by assigning a number to each employee in ascending order of their employee ID using ORDER BY clause and ROW NUMBER function on the employee ID field.

Instructions:

Refer to the employee table which is created and perform the above objective.

Use Case for Row Number

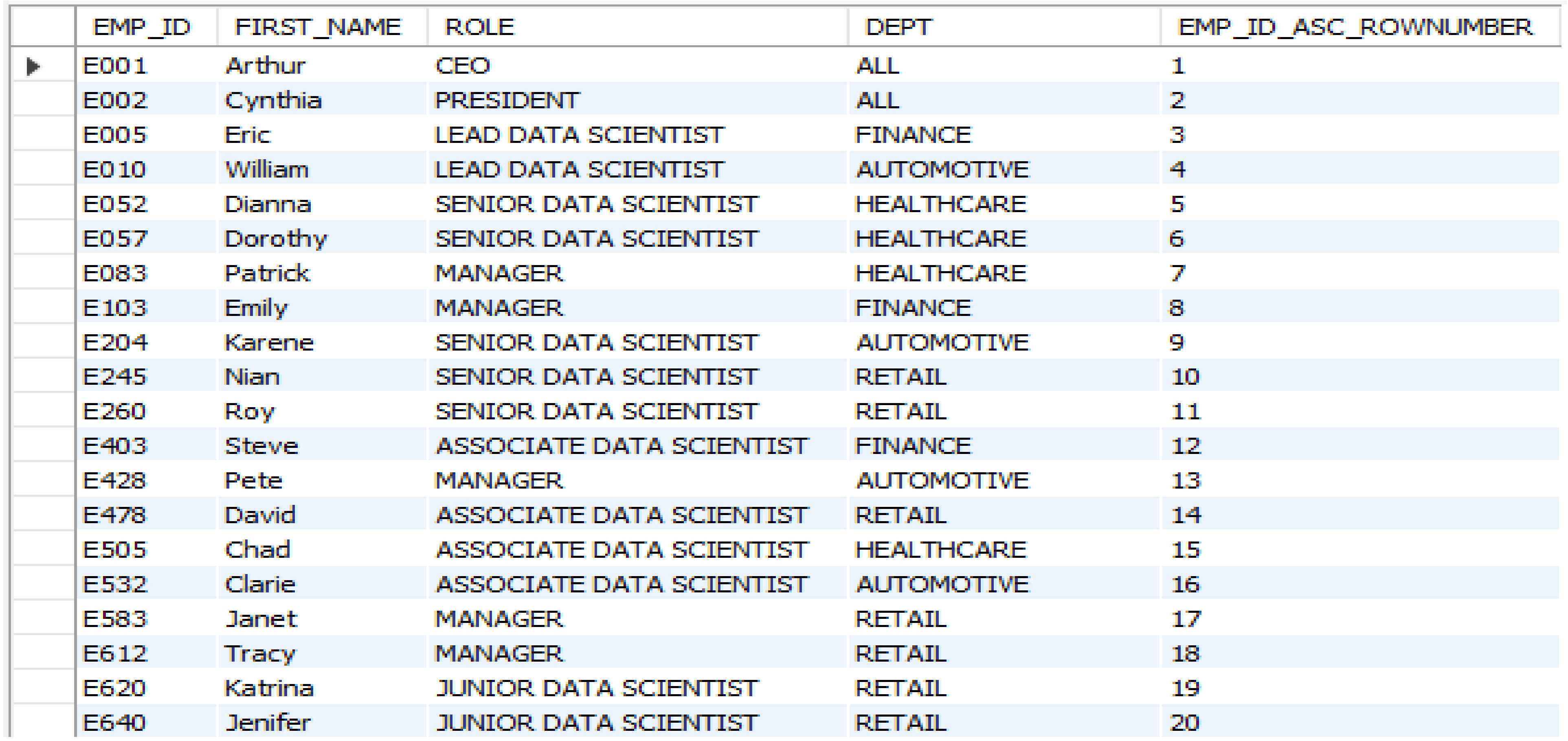
Solution:

```
# SELECT EMP_ID, FIRST_NAME, ROLE, DEPT and assign assetnumber to all the employee in
ascending order of their employee ID.



SELECT EMP_ID,FIRST_NAME,ROLE,DEPT,ROW_NUMBER() OVER(ORDER BY EMP_ID)
EMP_ID_ASC_ROWNUMBER FROM emp_table;
```

By executing this query, the IT department can identify a asset number assigned to each employee in ascending order of their employee ID.

Use Case for Row Number

Output:



EMP_ID	FIRST_NAME	ROLE	DEPT	EMP_ID_ASC_ROWNUMBER
E001	Arthur	CEO	ALL	1
E002	Cynthia	PRESIDENT	ALL	2
E005	Eric	LEAD DATA SCIENTIST	FINANCE	3
E010	William	LEAD DATA SCIENTIST	AUTOMOTIVE	4
E052	Dianna	SENIOR DATA SCIENTIST	HEALTHCARE	5
E057	Dorothy	SENIOR DATA SCIENTIST	HEALTHCARE	6
E083	Patrick	MANAGER	HEALTHCARE	7
E103	Emily	MANAGER	FINANCE	8
E204	Karene	SENIOR DATA SCIENTIST	AUTOMOTIVE	9
E245	Nian	SENIOR DATA SCIENTIST	RETAIL	10
E260	Roy	SENIOR DATA SCIENTIST	RETAIL	11
E403	Steve	ASSOCIATE DATA SCIENTIST	FINANCE	12
E428	Pete	MANAGER	AUTOMOTIVE	13
E478	David	ASSOCIATE DATA SCIENTIST	RETAIL	14
E505	Chad	ASSOCIATE DATA SCIENTIST	HEALTHCARE	15
E532	Clarie	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	16
E583	Janet	MANAGER	RETAIL	17
E612	Tracy	MANAGER	RETAIL	18
E620	Katrina	JUNIOR DATA SCIENTIST	RETAIL	19
E640	Jenifer	JUNIOR DATA SCIENTIST	RETAIL	20

Percent Rank

Definition

- Percent rank helps to evaluate the percentile rank of a value in a partition or result set.
- It returns a value between zero to one.

Syntax

```
PERCENT_RANK()OVER (
   PARTITION BY expr,...
    ORDER BY expr
[ASC|DESC],...)
```

Use Case for Percent Rank

Problem Scenario:

The HR of a company wants to calculate the overall percentile of the employee rating in a department.

Objective:

You are required to display employee’s ID, first name, role, department, and employee rating by calculating the percentile of the employee rating in a department using ORDER BY clause and PERCENT RANK function on an employee rating field.

Instructions:

Refer to the employee table which is created and perform the above objective.

Use Case for Percent Rank

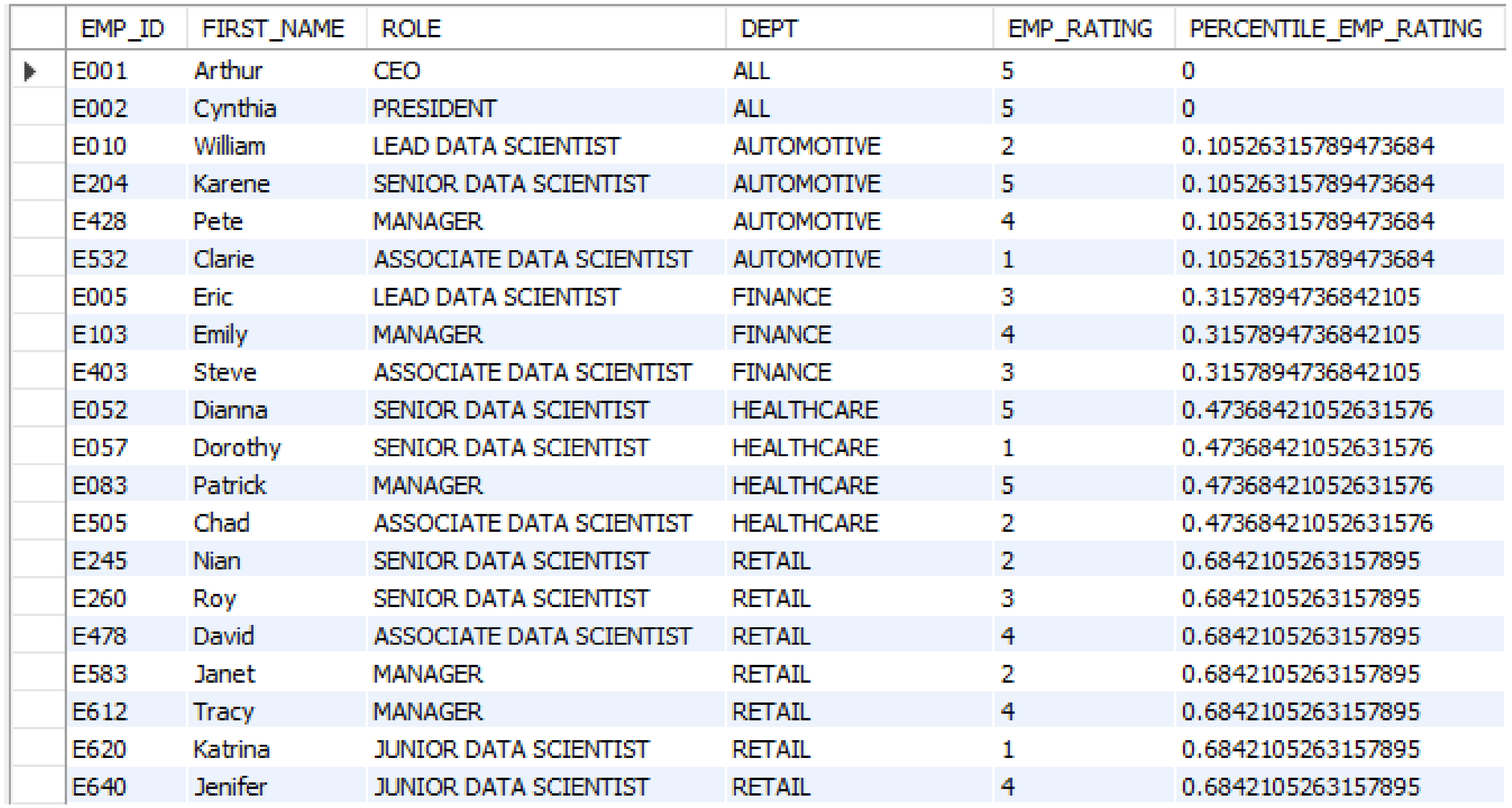
Solution:

```
# SELECT EMP_ID,FIRST_NAME,ROLE,DEPT,EMP_RATING and calculate the percentile of the
employee rating using ODERBY and PERCENT RANK function.



SELECT EMP_ID,FIRST_NAME,ROLE,DEPT,EMP_RATING,PERCENT_RANK() OVER(ORDER BY EMP_RATING)
PERCENTILE_EMP_RATING FROM emp_table;
```

By executing this query, the HR can identify the percentile score of each employee in a department.

Use Case for Percent Rank

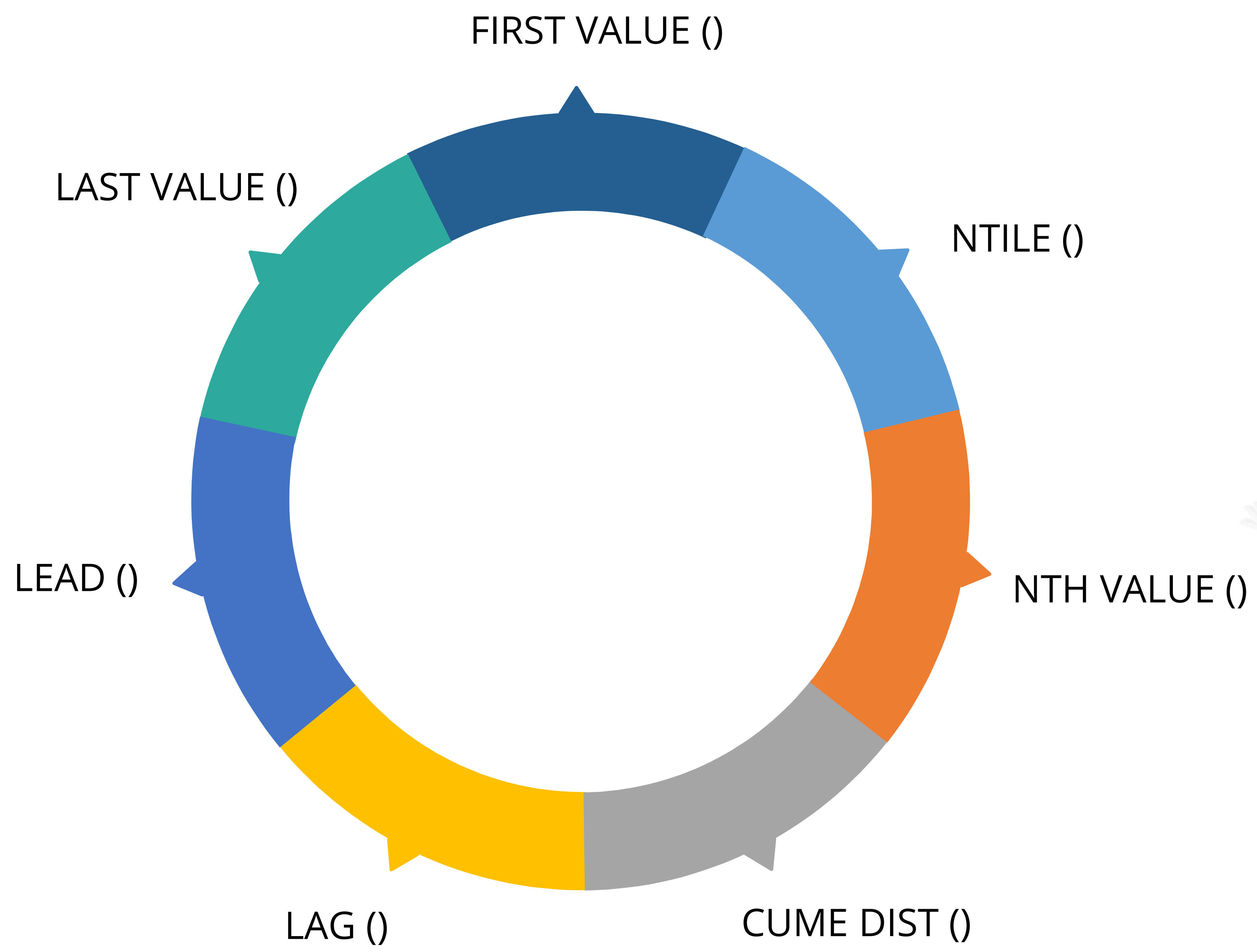
Output:



EMP_ID	FIRST_NAME	ROLE	DEPT	EMP_RATING	PERCENTILE_EMP_RATING
E001	Arthur	CEO	ALL	5	0
E002	Cynthia	PRESIDENT	ALL	5	0
E010	William	LEAD DATA SCIENTIST	AUTOMOTIVE	2	0.10526315789473684
E204	Karene	SENIOR DATA SCIENTIST	AUTOMOTIVE	5	0.10526315789473684
E428	Pete	MANAGER	AUTOMOTIVE	4	0.10526315789473684
E532	Clarie	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	1	0.10526315789473684
E005	Eric	LEAD DATA SCIENTIST	FINANCE	3	0.3157894736842105
E103	Emily	MANAGER	FINANCE	4	0.3157894736842105
E403	Steve	ASSOCIATE DATA SCIENTIST	FINANCE	3	0.3157894736842105
E052	Dianna	SENIOR DATA SCIENTIST	HEALTHCARE	5	0.47368421052631576
E057	Dorothy	SENIOR DATA SCIENTIST	HEALTHCARE	1	0.47368421052631576
E083	Patrick	MANAGER	HEALTHCARE	5	0.47368421052631576
E505	Chad	ASSOCIATE DATA SCIENTIST	HEALTHCARE	2	0.47368421052631576
E245	Nian	SENIOR DATA SCIENTIST	RETAIL	2	0.6842105263157895
E260	Roy	SENIOR DATA SCIENTIST	RETAIL	3	0.6842105263157895
E478	David	ASSOCIATE DATA SCIENTIST	RETAIL	4	0.6842105263157895
E583	Janet	MANAGER	RETAIL	2	0.6842105263157895
E612	Tracy	MANAGER	RETAIL	4	0.6842105263157895
E620	Katrina	JUNIOR DATA SCIENTIST	RETAIL	1	0.6842105263157895
E640	Jenifer	JUNIOR DATA SCIENTIST	RETAIL	4	0.6842105263157895

Miscellaneous Window Functions

Types of Miscellaneous Window Functions



First Value Function

Definition

First value function returns the value of the expression from the first row of the window frame.

Syntax

```
FIRST_VALUE (expression) OVER
( [partition_clause]
[order_clause] [frame_clause]
)
```

Use Case for First Value Function

Problem Scenario:

The HR department of an organization aims to find the employee ID of the employee with the highest experience by sorting their experience in descending order.

Objective:

You are required to display the employee ID, first name, and experience, as well as identify the employee ID of the first employee by sorting the experience in descending order using the ORDER BY clause and first value function on the experience and employee ID fields respectively.

Instructions:

Refer to the employee table which is created and perform the above objective.

Use Case for First Value Function

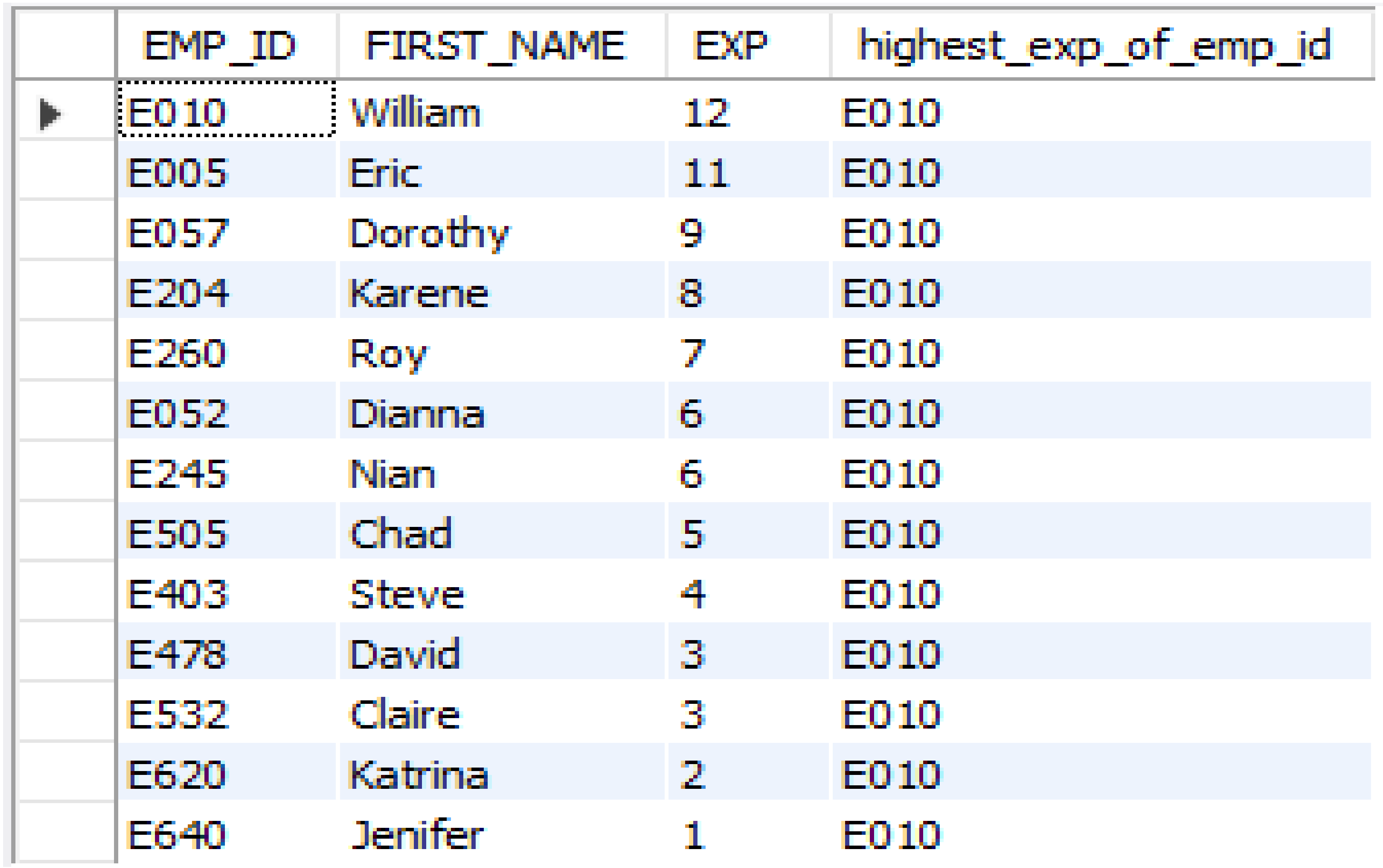
Solution:

```
# SELECT EMP_ID,FIRST_NAME,EXP and determine the highest experience in the EMP_ID based
on descending order of the experience.


SELECT EMP_ID, FIRST_NAME, EXP, FIRST_VALUE(EMP_ID) OVER (ORDER BY EXP DESC)
highest_exp_of_emp_id FROM emp_table;
```

By executing this query, HR can identify the employee ID with the highest experience.

Use Case for First Value Function

Output:



EMP_ID	FIRST_NAME	EXP	highest_exp_of_emp_id
E010	William	12	E010
E005	Eric	11	E010
E057	Dorothy	9	E010
E204	Karene	8	E010
E260	Roy	7	E010
E052	Dianna	6	E010
E245	Nian	6	E010
E505	Chad	5	E010
E403	Steve	4	E010
E478	David	3	E010
E532	Claire	3	E010
E620	Katrina	2	E010
E640	Jenifer	1	E010

NTH Value Function

Definition

The NTH value function acquires a value from the Nth row of an ordered group of rows.

Syntax

```
NTH_VALUE(expression, N)
FROM FIRST
OVER (
partition_clause
order_clause
frame_clause
)
```

Use Case for NTH Value Function

Problem Scenario:

The HR of a company wants to identify the third-highest experience among employees in the company.

Objective:

You are required to display the employee’s ID, first name, and experience by calculating the third-highest experience among employees using ORDER BY clause and NTH value function in descending order of experience field.

Instructions:

Refer to the employee table which is created and perform the following objective.

Use Case for NTH Value Function

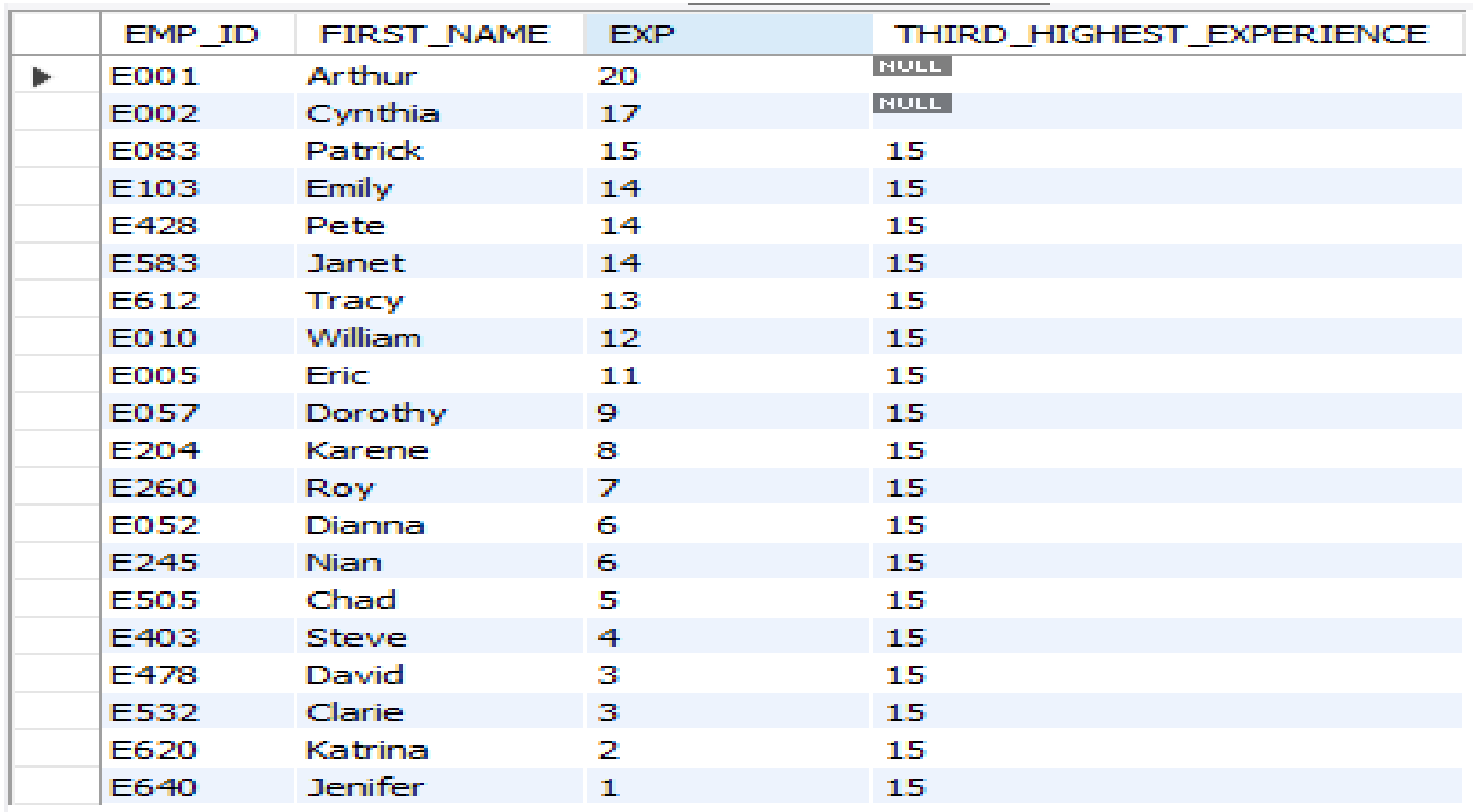
Solution:

```
# SELECT EMP_ID,FIRST_NAME,EXP and calculate the third highest experience among
employees using order by on experience in descending order and N_TH VALUE function



SELECT    EMP_ID,FIRST_NAME,EXP,    NTH_VALUE(EXP,3) OVER(ORDER BY EXP DESC)
THIRD_HIGHEST_EXPERIENCE FROM emp_table
```

By executing this query, the HR can identify the third highest experience of the employee.

Use Case for NTH Value Function

Output:



	EMP_ID	FIRST_NAME	EXP	THIRD_HIGHEST_EXPERIENCE
▶	E001	Arthur	20	NULL
	E002	Cynthia	17	NULL
	E083	Patrick	15	15
	E103	Emily	14	15
	E428	Pete	14	15
	E583	Janet	14	15
	E612	Tracy	13	15
	E010	William	12	15
	E005	Eric	11	15
	E057	Dorothy	9	15
	E204	Karene	8	15
	E260	Roy	7	15
	E052	Dianna	6	15
	E245	Nian	6	15
	E505	Chad	5	15
	E403	Steve	4	15
	E478	David	3	15
	E532	Clarie	3	15
	E620	Katrina	2	15
	E640	Jenifer	1	15

NTILE Function

Definition

NTILE function breaks the rows into a sorted partition in a certain number of groups.

Syntax

```
NTILE(n) OVER (
PARTITION BY
<expression>[{,<expression>..
.}]
ORDER BY <expression>
[ASC|DESC],
[{,<expression>...}]
)
```

Use Case for NTILE Function

Problem Scenario:

The HR of a company wants to sort the employee table in ascending based on their experience in four partitions.

Objective:

You are required to display all the details by sorting the experience into four partitions in ascending order using the ORDER BY clause and NTILE function on an experience field.

Instructions:

Refer to the employee table which is created and perform the above objective.

Use Case for NTILE Function

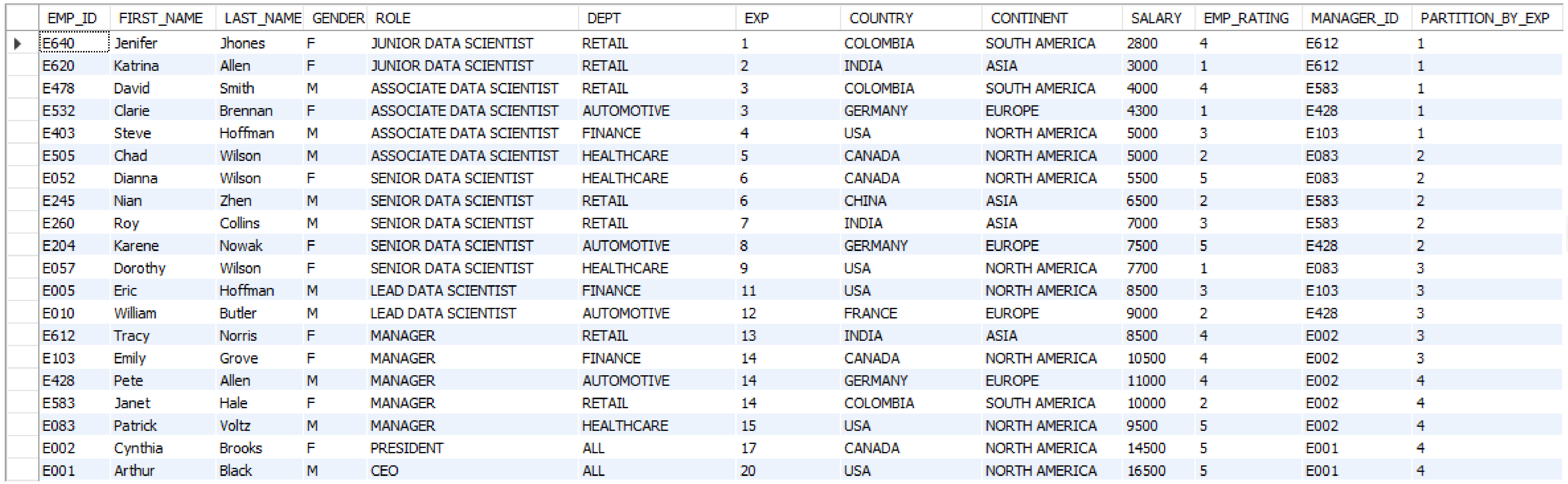
Solution:

```
# SELECT all the details in the employee table by sorting in ascending order of the
experience into four partitions using ORDER BY on EXP and NTILE function.



SELECT   * ,  NTILE(4)OVER(ORDER BY EXP) PARTITION_BY_EXP FROM emp_table
```

By executing this query, the HR can view the experience in ascending order into four partitions.

Use Case for NTILE Function

Output:



EMP_ID	FIRST_NAME	LAST_NAME	GENDER	ROLE	DEPT	EXP	COUNTRY	CONTINENT	SALARY	EMP_RATING	MANAGER_ID	PARTITION_BY_EXP
E640	Jenifer	Jhones	F	JUNIOR DATA SCIENTIST	RETAIL	1	COLOMBIA	SOUTH AMERICA	2800	4	E612	1
E620	Katrina	Allen	F	JUNIOR DATA SCIENTIST	RETAIL	2	INDIA	ASIA	3000	1	E612	1
E478	David	Smith	M	ASSOCIATE DATA SCIENTIST	RETAIL	3	COLOMBIA	SOUTH AMERICA	4000	4	E583	1
E532	Clarie	Brennan	F	ASSOCIATE DATA SCIENTIST	AUTOMOTIVE	3	GERMANY	EUROPE	4300	1	E428	1
E403	Steve	Hoffman	M	ASSOCIATE DATA SCIENTIST	FINANCE	4	USA	NORTH AMERICA	5000	3	E103	1
E505	Chad	Wilson	M	ASSOCIATE DATA SCIENTIST	HEALTHCARE	5	CANADA	NORTH AMERICA	5000	2	E083	2
E052	Dianna	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	6	CANADA	NORTH AMERICA	5500	5	E083	2
E245	Nian	Zhen	M	SENIOR DATA SCIENTIST	RETAIL	6	CHINA	ASIA	6500	2	E583	2
E260	Roy	Collins	M	SENIOR DATA SCIENTIST	RETAIL	7	INDIA	ASIA	7000	3	E583	2
E204	Karene	Nowak	F	SENIOR DATA SCIENTIST	AUTOMOTIVE	8	GERMANY	EUROPE	7500	5	E428	2
E057	Dorothy	Wilson	F	SENIOR DATA SCIENTIST	HEALTHCARE	9	USA	NORTH AMERICA	7700	1	E083	3
E005	Eric	Hoffman	M	LEAD DATA SCIENTIST	FINANCE	11	USA	NORTH AMERICA	8500	3	E103	3
E010	William	Butler	M	LEAD DATA SCIENTIST	AUTOMOTIVE	12	FRANCE	EUROPE	9000	2	E428	3
E612	Tracy	Norris	F	MANAGER	RETAIL	13	INDIA	ASIA	8500	4	E002	3
E103	Emily	Grove	F	MANAGER	FINANCE	14	CANADA	NORTH AMERICA	10500	4	E002	3
E428	Pete	Allen	M	MANAGER	AUTOMOTIVE	14	GERMANY	EUROPE	11000	4	E002	4
E583	Janet	Hale	F	MANAGER	RETAIL	14	COLOMBIA	SOUTH AMERICA	10000	2	E002	4
E083	Patrick	Voltz	M	MANAGER	HEALTHCARE	15	USA	NORTH AMERICA	9500	5	E002	4
E002	Cynthia	Brooks	F	PRESIDENT	ALL	17	CANADA	NORTH AMERICA	14500	5	E001	4
E001	Arthur	Black	M	CEO	ALL	20	USA	NORTH AMERICA	16500	5	E001	4

Cume Dist Function

Definition

The Cume Dist function calculates the cumulative distribution of a number in a group of values.

Syntax

```
CUME_DIST( ) OVER ( [
partition_by_clause ]
order_by_clause )
```

Use Case for Cume Dist Function

Problem Scenario:

The HR of a company wants to sort the employee data based on their experience in ascending order and calculate the cumulative distribution on the employee table.

Objective:

You are required to display the employee's ID, first name, and experience by calculating the cumulative distribution of the experience with the help of ROW NUMBER using ORDER BY, ROW NUMBER, and CUME DIST function on an experience field.

Instructions:

Refer to the employee table which is created and perform the above objective.

Use Case for Cume Dist Function

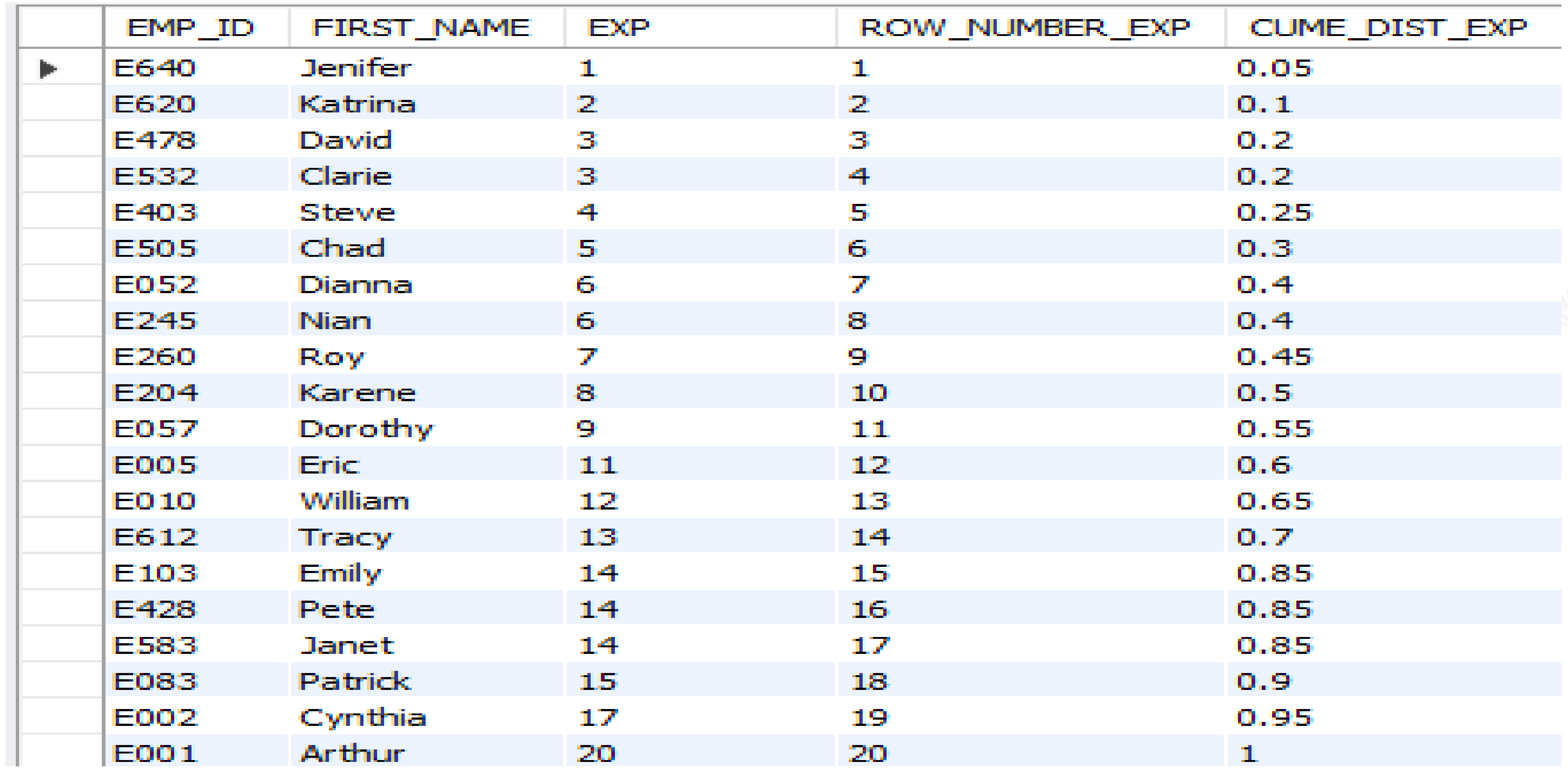
Solution:

```
# SELECT EMP_ID,FIRST_NAME,EXP and calculate the cumulative distribution of the
experience with the help of Row number using ODERBY and ROW_NUMBER, CUME_DIST function.



SELECT  EMP_ID,FIRST_NAME,EXP,     ROW_NUMBER() OVER (ORDER BY EXP) ROW_NUMBER_EXP,
CUME_DIST() OVER (ORDER BY EXP) CUME_DIST_EXP FROM emp_table;
```

By executing this query, the HR can identify the cumulative distribution of the experience with the help of row number .

Use Case for Cume Dist Function

Output:



	EMP_ID	FIRST_NAME	EXP	ROW_NUMBER_EXP	CUME_DIST_EXP
▶	E640	Jenifer	1	1	0.05
	E620	Katrina	2	2	0.1
	E478	David	3	3	0.2
	E532	Clarie	3	4	0.2
	E403	Steve	4	5	0.25
	E505	Chad	5	6	0.3
	E052	Dianna	6	7	0.4
	E245	Nian	6	8	0.4
	E260	Roy	7	9	0.45
	E204	Karene	8	10	0.5
	E057	Dorothy	9	11	0.55
	E005	Eric	11	12	0.6
	E010	William	12	13	0.65
	E612	Tracy	13	14	0.7
	E103	Emily	14	15	0.85
	E428	Pete	14	16	0.85
	E583	Janet	14	17	0.85
	E083	Patrick	15	18	0.9
	E002	Cynthia	17	19	0.95
	E001	Arthur	20	20	1

Lead Function

Definition

Lead function is used to retrieve the values from the next N rows.

Syntax

```
LEAD OVER() ( PARTITION BY
(expr) ORDER BY (expr) )
```

Lag Function

Definition

- Lag function is used to retrieve the values from previous N rows.
- It is the reverse of lead function.

Syntax

```
LAG () OVER ( PARTITION BY
expr,... ORDER BY expr
[ASC|DESC],... )
```

Use Case for Lead and Lag Function

Problem Scenario:

The HR of a company wants to ignore the two lowest and highest experiences of the employees.

Objective:

You are required to display the employee’s ID, first name, experience and sort the employees in ascending order of their experience. Ignore the two lowest experiences using LEAD and two highest experiences using LAG to determine the median of the employee experience.

Instructions:

Refer to the employee table which is created and perform the above objective.

Use Case for Lead and Lag Function

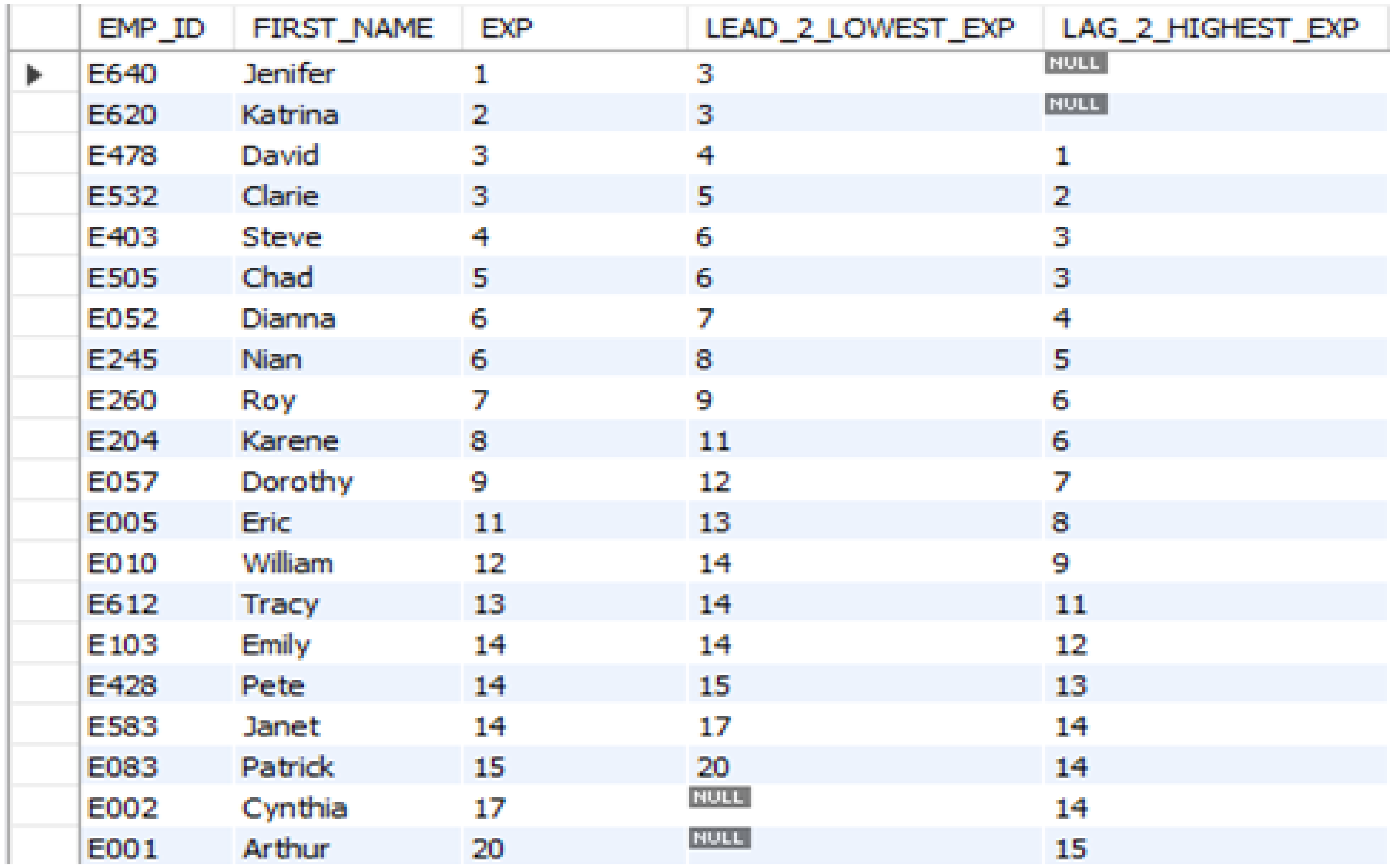
Solution:

```
# SELECT EMP_ID,FIRST_NAME, EXP and ignore the two lowest and highest experience using
LEAD and LAG function



SELECT     EMP_ID,FIRST_NAME,EXP,     LEAD(EXP,2) OVER (ORDER BY EXP) LEAD_2_LOWEST_EXP,
LAG(EXP,2) OVER (ORDER BY EXP) LAG_2_HIGHEST_EXPFROM emp_table
```

By executing this query, the HR can identify the median experience of the employee.

Use Case for Lead and Lag Function

Output:



EMP_ID	FIRST_NAME	EXP	LEAD_2_LOWEST_EXP	LAG_2_HIGHEST_EXP
E640	Jenifer	1	3	NULL
E620	Katrina	2	3	NULL
E478	David	3	4	1
E532	Clarie	3	5	2
E403	Steve	4	6	3
E505	Chad	5	6	3
E052	Dianna	6	7	4
E245	Nian	6	8	5
E260	Roy	7	9	6
E204	Karene	8	11	6
E057	Dorothy	9	12	7
E005	Eric	11	13	8
E010	William	12	14	9
E612	Tracy	13	14	11
E103	Emily	14	14	12
E428	Pete	14	15	13
E583	Janet	14	17	14
E083	Patrick	15	20	14
E002	Cynthia	17	NULL	14
E001	Arthur	20	NULL	15

Last Value Function

Definition

Last value function returns the last value of a specific column in an ordered sequence.

Syntax

```
LAST_VALUE (expression) OVER
( [partition_clause]
[order_clause] [frame_clause]
)
```

Use Case for Last Value Function

Problem Scenario:

The HR of a company wants to determine the last employee ID by sorting the experience in ascending order.

Objective:

You are required to display the employee’s ID, first name, and experience and determine the last employee ID by sorting the experience in ascending order using ORDER BY clause and last value function on the experience and employee ID field respectively.

Instructions:

Refer to the employee table which is created and perform the above objective.

Use Case for Last Value Function

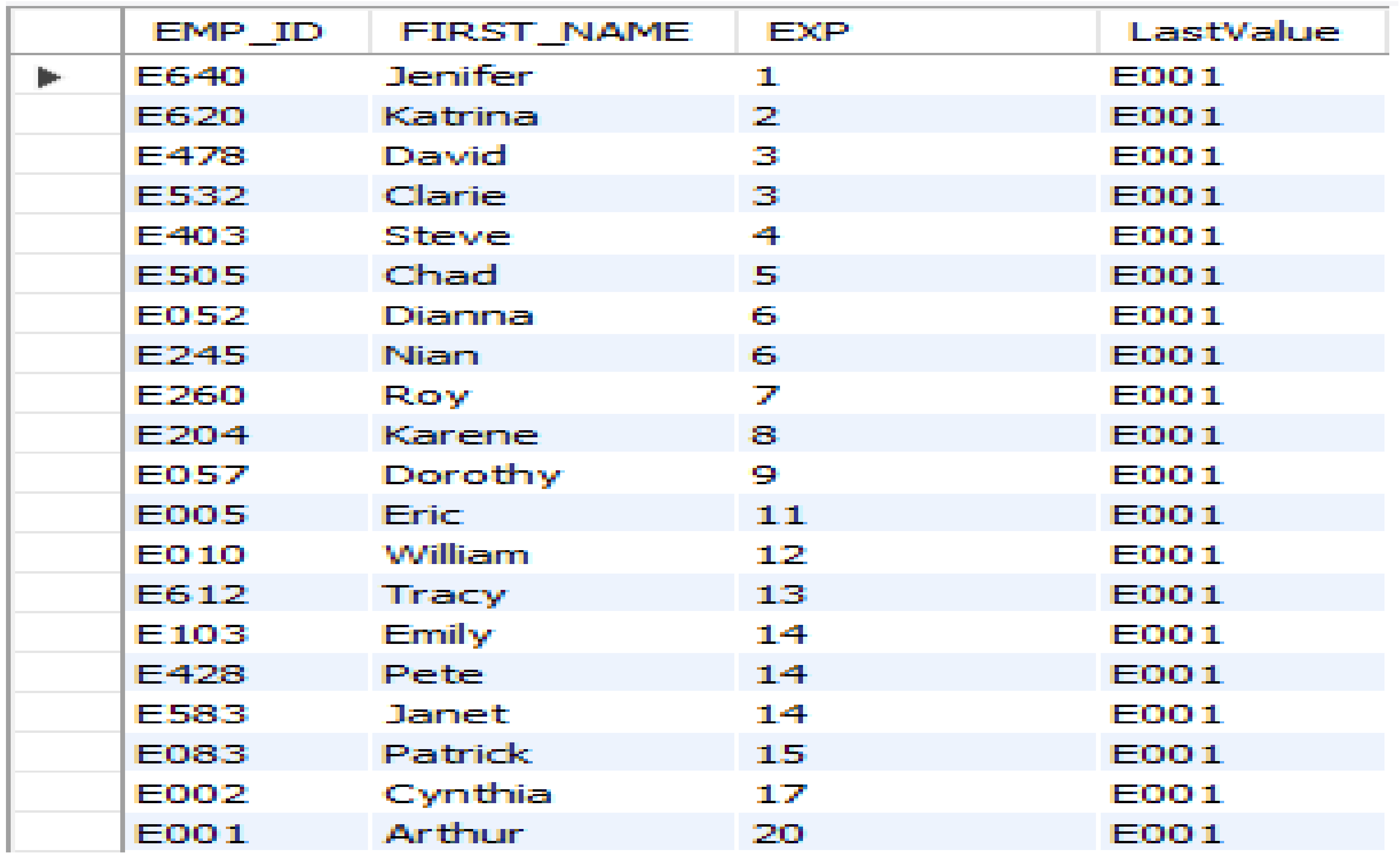
Solution:

```
# SELECT EMP_ID,FIRST_NAME,EXP and determine the last value in the EMP_ID based on
ascending order of the experience.



SELECT    EMP_ID,FIRST_NAME,EXP,    LAST_VALUE(EMP_ID) OVER (ORDER BY EXP         RANGE
BETWEEN          UNBOUNDED PRECEDING AND           UNBOUNDED FOLLOWING ) LastValue
FROM emp_table
```

By executing this query, the HR can identify the last value of the employee ID based on their experience.

Use Case for Last Value Function

Output:



	EMP_ID	FIRST_NAME	EXP	LastValue
▶	E640	Jenifer	1	E001
	E620	Katrina	2	E001
	E478	David	3	E001
	E532	Clarie	3	E001
	E403	Steve	4	E001
	E505	Chad	5	E001
	E052	Dianna	6	E001
	E245	Nian	6	E001
	E260	Roy	7	E001
	E204	Karene	8	E001
	E057	Dorothy	9	E001
	E005	Eric	11	E001
	E010	William	12	E001
	E612	Tracy	13	E001
	E103	Emily	14	E001
	E428	Pete	14	E001
	E583	Janet	14	E001
	E083	Patrick	15	E001
	E002	Cynthia	17	E001
	E001	Arthur	20	E001

Assisted Practice: Ranking and Miscellaneous Window Functions

Duration: 15 min

Problem Statement: You are required to identify the rank and row number and calculate the cumulative distribution and percentile score based on the student score from the marksheet table.

Assisted Practice: Ranking and Miscellaneous Window Functions

Steps to be performed:

Step 1: Creating the marksheet table and inserting values in it:

CREATE

```
CREATE TABLE marksheet (    score INT NOT NULL,    year INT NULL,    class
varchar(45) NULL,    ranking varchar(45) NULL,    s_id INT NOT NULL    );
```

-

INSERT

```
INSERT INTO marksheet (score,year,class,ranking,s_id) VALUES
('989','2014','10','1','1');
```

Assisted Practice: Ranking and Miscellaneous Window Functions

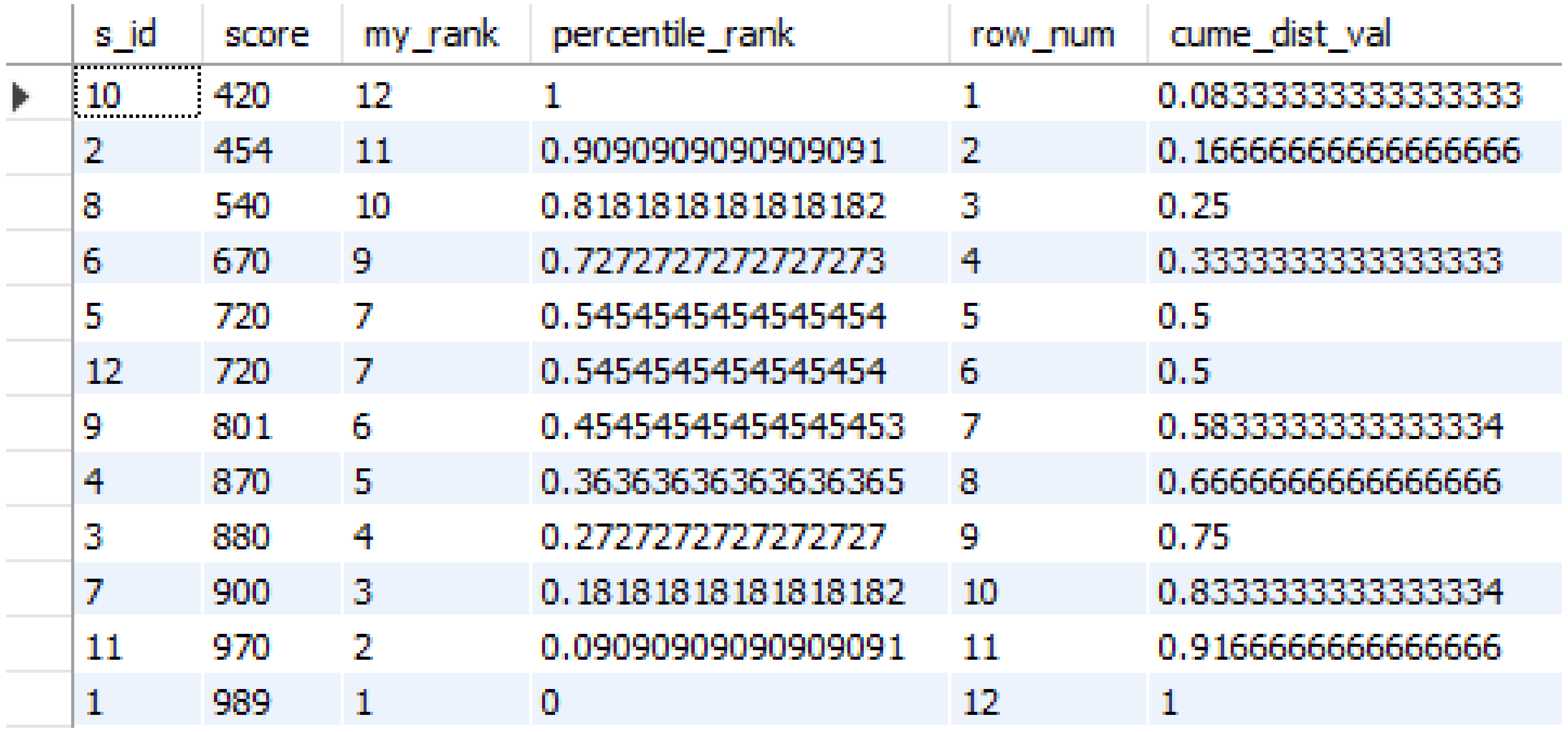
Step 2: Querying to perform window functions:

QUERY

```
SELECT s_id,score, RANK() OVER (ORDER BY score DESC) my_rank ,    PERCENT_RANK()
OVER (ORDER BY score DESC) percentile_rank ,    ROW_NUMBER() OVER (ORDER BY
score) row_num,    CUME_DIST() OVER (ORDER BY score) cume_dist_val FROM
marksheet;
```

Assisted Practice: Ranking and Miscellaneous Window Functions

Output:



s_id	score	my_rank	percentile_rank	row_num	cume_dist_val
10	420	12	1	1	0.08333333333333333
2	454	11	0.9090909090909091	2	0.16666666666666666
8	540	10	0.8181818181818182	3	0.25
6	670	9	0.7272727272727273	4	0.3333333333333333
5	720	7	0.5454545454545454	5	0.5
12	720	7	0.5454545454545454	6	0.5
9	801	6	0.45454545454545453	7	0.5833333333333334
4	870	5	0.36363636363636365	8	0.6666666666666666
3	880	4	0.2727272727272727	9	0.75
7	900	3	0.18181818181818182	10	0.8333333333333334
11	970	2	0.09090909090909091	11	0.9166666666666666
1	989	1	0	12	1

Key Takeaways

- Window functions provide an aggregated value for each row with a unique identity.
- Aggregate window functions perform on a particular set of rows and provide the result in a single row.
- Ranking window functions specify the rank for individual fields as per the categorization.

Knowledge Check

Knowledge Check

1

What is the result of Window Functions ?

A. Aggregate value of each row

B. Group of values

C. Sorted values

D. Divides into partition

Knowledge Check

1

What is the result of Window Functions ?

A. Aggregate value of each row

B. Group of values

C. Sorted values

D. Divides into partition

The correct answer is **A**

Window functions perform various operations on a group of rows and provide an aggregated value for each row.

Knowledge Check

2

Which of the following are the clauses in MySQL ?

A. Row number and Rank

B. Last value and First value

C. Minimum and Maximum

D. Partition, Frame, and Order By

Knowledge Check

2

Which of the following are the clauses in MySQL ?

A. Row number and Rank

B. Last value and First value

C. Minimum and Maximum

D. Partition, Frame, and Order By

The correct answer is **D**

The types of clauses are Partition Clause, Frame Clause, and Order By Clause.

Knowledge Check

3

What is the return type in COUNT() ?

A. DECIMAL

B. FLOAT

C. BIGINT

D. DOUBLE

Knowledge Check

3

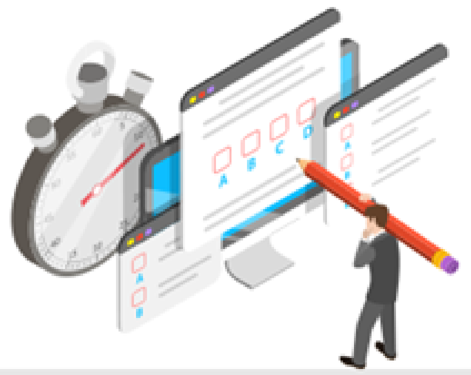
What is the return type in COUNT() ?

A. DECIMAL

B. FLOAT

C. BIGINT

D. DOUBLE



The correct answer is **C**

The return type of COUNT() is BIGINT.

Knowledge Check

4

Which ranking window function returns a value from zero to one ?

A. NTH value

B. Percent rank

C. N title

D. Row number

Knowledge Check

4

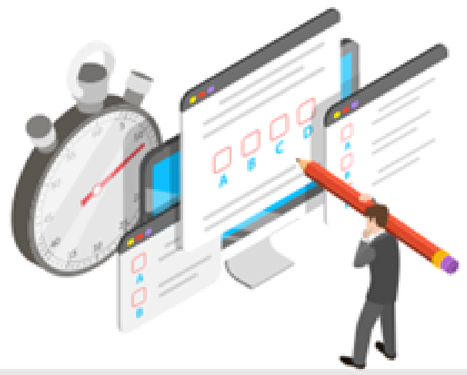
Which ranking window function returns a value from zero to one ?

A. NTH value

B. Percent rank

C. N title

D. Row number



The correct answer is **B**

Percent Rank returns value from zero to one.

Knowledge Check

5

What is the purpose of LEAD Function ?

A. Retrieve the values from next N rows

B. Retrieve the values from previous N rows

C. Retrieve unique sequential number

D. Assign rank to all the rows within every partition

Knowledge Check

5

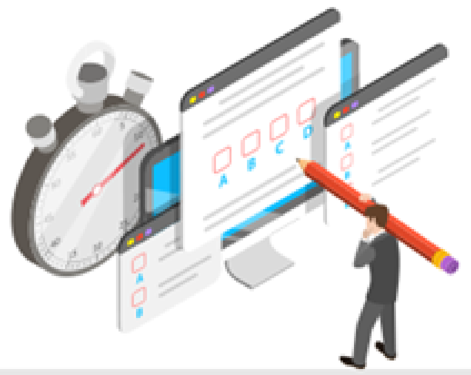
What is the purpose of LEAD Function ?

A. Retrieve the values from next N rows

B. Retrieve the values from previous N rows

C. Retrieve unique sequential number

D. Assign rank to all the rows within every partition



The correct answer is **A**

LEAD function retrieves the values from next N rows.