FTDS // Practical Statistics : Descriptive with SQL

Hacktiv8 DS Curriculum Team

Phase 0 Learning Materials

Hacktiv8 DS
Curriculum
Team

Contents

Objectives

- able to understand Group by, Rank, and Row function on SQL
- able to implement SQL on descriptive statistics problems

We use GROUP BY to group rows into groups. However, It is not mandatory to include an aggregate function in the SELECT clause. However, if you use an aggregate function, it will calculate the summary value for each group.

```
SELECT
        column1,
        column2,
        AGGREGATE_FUNCTION (column3)
FROM
        table1
GROUP BY
        column1,
        column2;
```

Example

```
SELECT
        department_id,
        COUNT(employee_id) headcount
FROM
        employees
GROUP BY
        department_id;
```

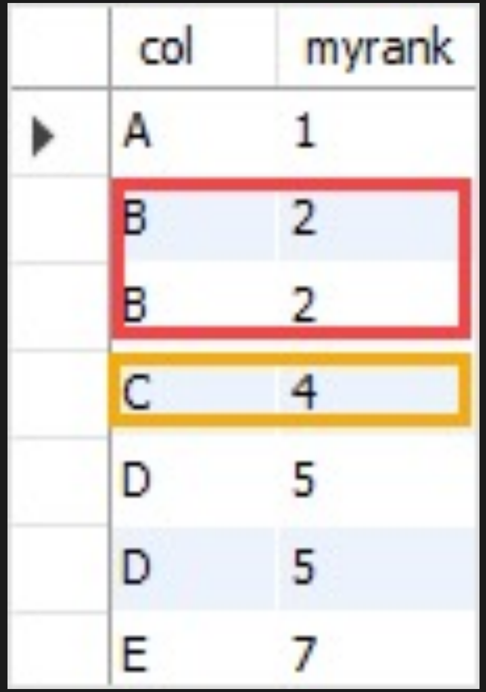
The RANK() function is a window function that assigns a rank to each row in the partition of a result set. The rank of a row is determined by one plus the number of ranks that come before it.

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

- The PARTITION BY clause distributes the rows in the result set into partitions by one or more criteria.
- The ORDER BY clause sorts the rows in each a partition.
- The RANK() function is operated on the rows of each partition and re-initialized when crossing each partition boundary.

Example 1

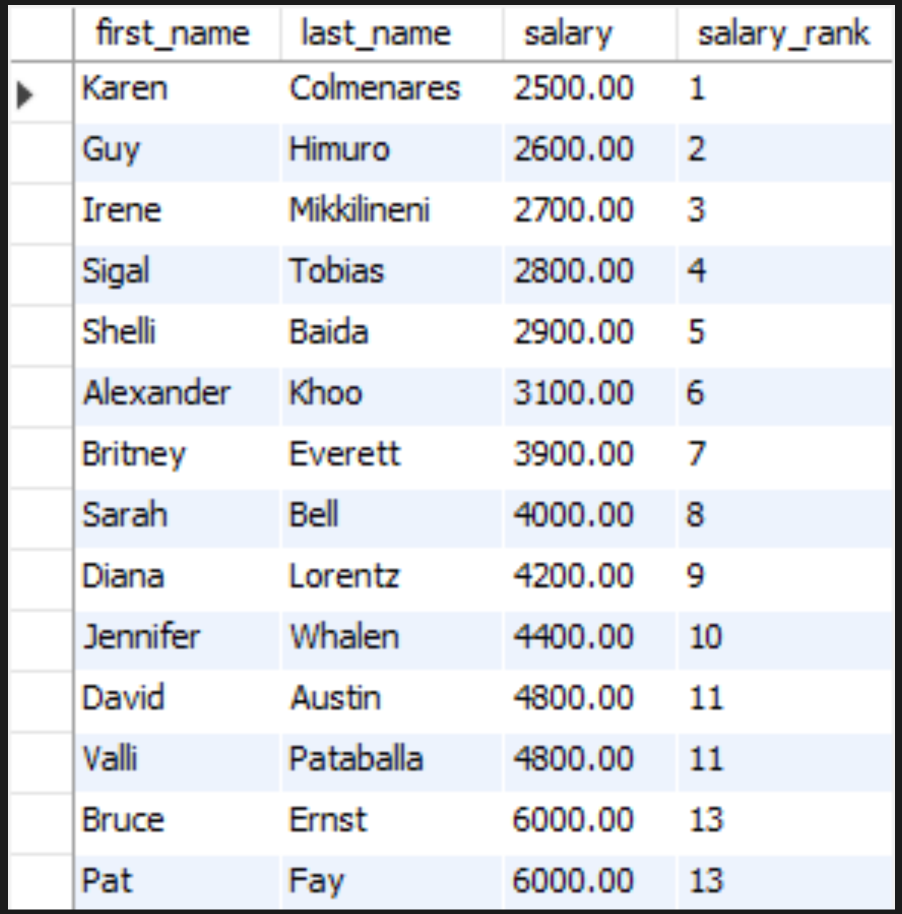
```
SELECT
        col, RANK() OVER (
                ORDER BY col
        ) myrank
FROM t;
```



	col	myrank
▶	A	1
	B	2
	B	2
	C	4
	D	5
	D	5
	E	7

Example 2

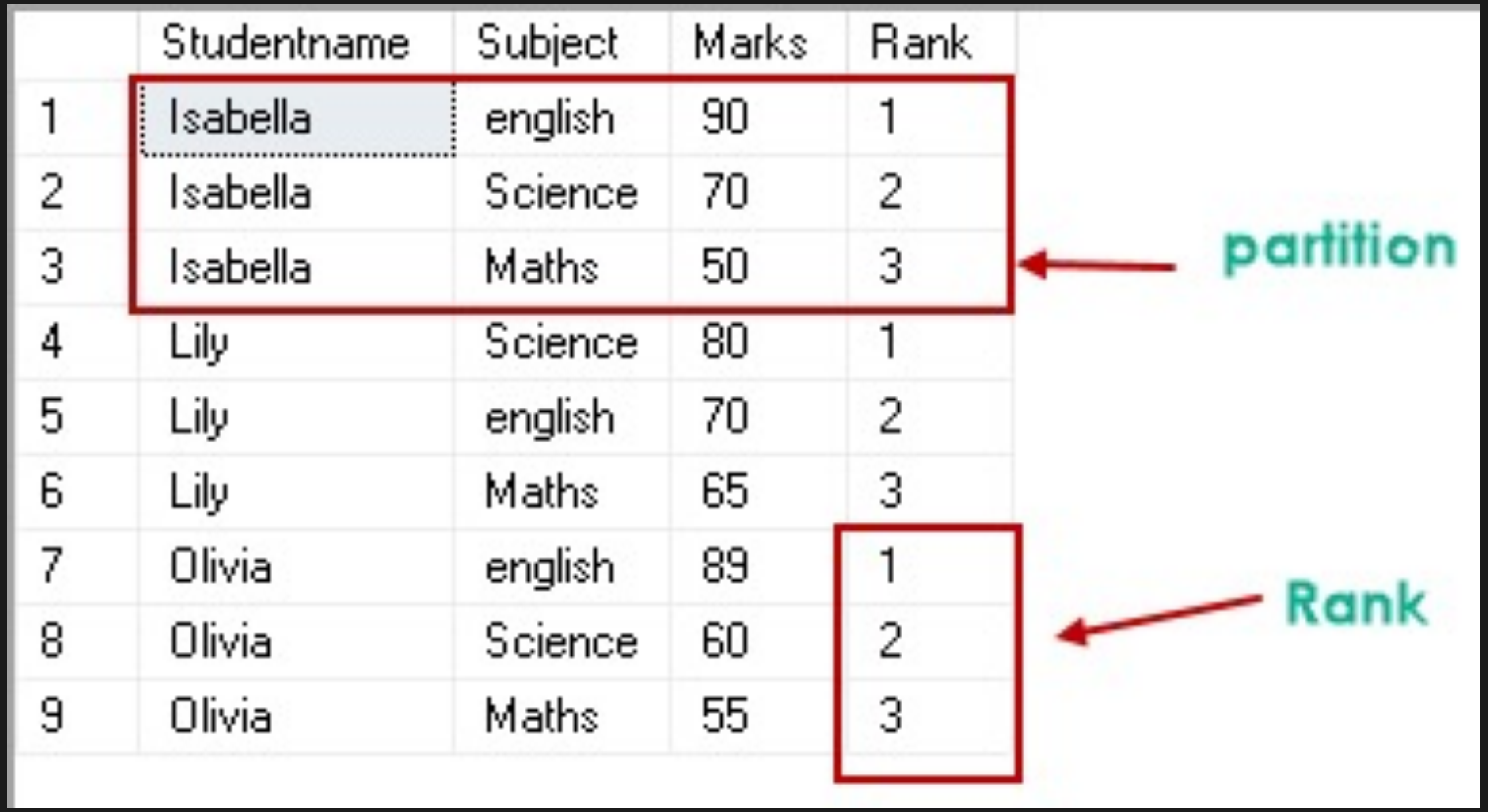
```
SELECT first_name, last_name, salary,
        RANK() OVER (ORDER BY salary) salary_rank
FROM employees;
```



	first_name	last_name	salary	salary_rank
▶	Karen	Colmenares	2500.00	1
	Guy	Himuro	2600.00	2
	Irene	Mikkilineni	2700.00	3
	Sigal	Tobias	2800.00	4
	Shelli	Baida	2900.00	5
	Alexander	Khoo	3100.00	6
	Britney	Everett	3900.00	7
	Sarah	Bell	4000.00	8
	Diana	Lorentz	4200.00	9
	Jennifer	Whalen	4400.00	10
	David	Austin	4800.00	11
	Valli	Pataballa	4800.00	11
	Bruce	Ernst	6000.00	13
	Pat	Fay	6000.00	13

Example 3

```
SELECT Studentname,
       Subject,
       Marks,
       RANK() OVER(
         PARTITION BY Studentname
         ORDER BY Marks DESC) Rank
FROM ExamResult
ORDER BY Studentname, Rank;
```



	Studentname	Subject	Marks	Rank
1	Isabella	english	90	1
2	Isabella	Science	70	2
3	Isabella	Maths	50	3
4	Lily	Science	80	1
5	Lily	english	70	2
6	Lily	Maths	65	3
7	Olivia	english	89	1
8	Olivia	Science	60	2
9	Olivia	Maths	55	3

The ROW_NUMBER() is a window function that assigns a sequential integer number to each row in the query's result set.

```
ROW_NUMBER ( ) OVER (
        [PARTITION BY expr1, expr2, …]
        ORDER BY expr1 [ASC|DESC], expr2, …
)
```

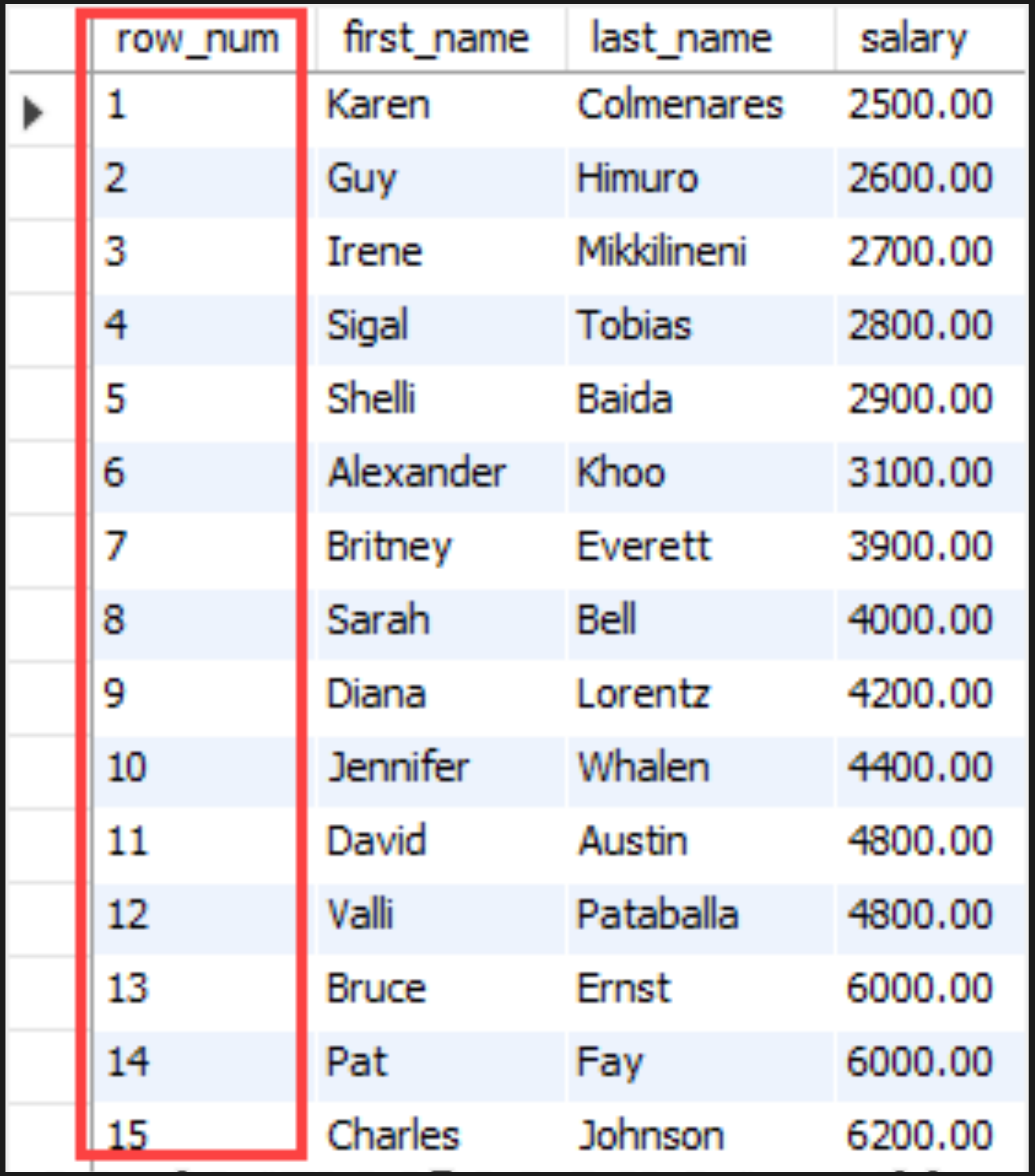
- First, the PARTITION BY clause divides the result set returned from the FROM clause into partitions. The PARTITION BY clause is optional. If you omit it, the whole result set is treated as a single partition.

- Then, the ORDER BY clause sorts the rows in each partition. Because the ROW_NUMBER() is an order sensitive function, the ORDER BY clause is required.

- Finally, each row in each partition is assigned a sequential integer number called a row number. The row number is reset whenever the partition boundary is crossed.

Example

```
SELECT
        ROW_NUMBER() OVER (
                ORDER BY salary
        ) row_num,
        first_name,
        last_name,
        salary
FROM
        employees;
```



row_num	first_name	last_name	salary
1	Karen	Colmenares	2500.00
2	Guy	Himuro	2600.00
3	Irene	Mikkilineni	2700.00
4	Sigal	Tobias	2800.00
5	Shelli	Baida	2900.00
6	Alexander	Khoo	3100.00
7	Britney	Everett	3900.00
8	Sarah	Bell	4000.00
9	Diana	Lorentz	4200.00
10	Jennifer	Whalen	4400.00
11	David	Austin	4800.00
12	Valli	Pataballa	4800.00
13	Bruce	Ernst	6000.00
14	Pat	Fay	6000.00
15	Charles	Johnson	6200.00

Hacktiv8 DS
Curriculum
Team

Hands On

Colab Link — Visit Here