**MySQL | PARTITION BY Clause**

A **PARTITION BY**clause is used to partition the rows of table into groups. It is useful when we have to perform a calculation on individual rows of a group using other rows of that group.

* It is always used inside OVER() clause.
* The partition formed by partition clause are also known as **Window**.
* This clause works on windows functions only. Like- RANK(), LEAD(), LAG() etc.
* If this clause is omitted in OVER() clause, then whole table is considered as a single partition.

**Syntax:**  
The syntax for Partition clause is-

Window\_function ( expression )

Over ( partition by expr [order\_clause] [frame\_clause] )

Here, order\_clause and frame\_clause are optional.

**expr** can be column names or built-in functions in MySQL.

But, standard SQL permits only column names in expr.

**Examples:**

Consider, a table “**Hacker**“:

| h\_id | h\_name | challenge\_id | score |
| --- | --- | --- | --- |
| 3 | shubh | 111 | 20 |
| 2 | aayush | 111 | 80 |
| 5 | krithik | 112 | 40 |
| 5 | krithik | 114 | 90 |
| 4 | tushar | 112 | 30 |
| 1 | parth | 112 | 40 |

We have to find the rank of hackers in each challenge. That means we have to list all participated hackers of a challenge along with their rank in that challenge.

**Query:**

select challenge\_id, h\_id, h\_name, score,

dense\_rank() over ( partition by challenge\_id order by score desc )

as "rank", from hacker;

**Explanation:**

In the above query, ***partition by*** clause will partition table into groups that are having same challenge\_id.

***order by*** will arrange the hackers of each partition in descending order by “scores”.

***over()*** clause defines how to partition and order rows of table, which is to be processed by window function rank().

***dense\_rank()*** is a window function, which will assign rank in ordered partition of challenges. If two hackers have same scores then they will be assigned same rank.

**Output:**

| challenge\_id | h\_id | h\_name | score | rank |
| --- | --- | --- | --- | --- |
| 111 | 2 | aayush | 80 | 1 |
| 111 | 3 | shubh | 20 | 2 |
| 112 | 5 | krithik | 40 | 1 |
| 112 | 1 | parth | 40 | 1 |
| 112 | 4 | tushar | 30 | 2 |
| 114 | 5 | krithik | 90 | 1 |

Thus, we get list of all hackers along with their ranks in the individual challenges.