**In Cassandra, Can we do ordering?**

Yes, Cassandra supports sorting using the clustering columns.by ORDER BY

In Cassandra, the ORDER BY clause is used to specify the order in which the query results should be returned based on the values of one or more columns. However, it's important to note that the ORDER BY clause in Cassandra has some rules and limitations and behaves differently compared to traditional relational databases.

Here's how the ORDER BY clause works in Cassandra:

* **Clustering Columns:**

Cassandra uses clustering columns to determine the order of rows within a partition. The order is determined by the clustering columns' definitions in the table schema.

* **Partitioning**:

In Cassandra, data is distributed across multiple nodes based on the partition key. Each partition is stored on a separate node. When querying data with the ORDER BY clause, it operates on a single partition.

* **Single Partition Order:**

When the ORDER BY clause is used in a query, it is important to include the partition key columns in the WHERE clause. This ensures that the query operates on a single partition, as ordering is guaranteed only within a partition.

* **Ascending or Descending Order:**

Cassandra allows you to specify the order as either ascending (ASC) or descending (DESC). By default, the order is ascending. You can specify the order for each clustering column individually.

* **Limitations:**

The ORDER BY clause in Cassandra has some limitations. It can only be used on columns that are part of the primary key or indexed columns. Attempting to use the ORDER BY clause on non-indexed columns can result in an error. Cassandra also has limitations on the number of rows it can efficiently order within a partition.

* **Performance Considerations:**

The ORDER BY clause in Cassandra can impact performance, especially when used on large datasets. Ordering requires reading and sorting data within a single partition, which can be resource-intensive. It's important to design the data model and queries carefully to optimize the performance of ordered queries.

* **Clustering rule**

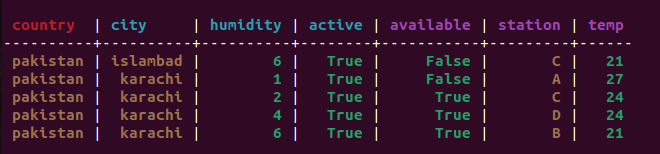
if the table has been defined without any specific CLUSTERING ORDER, then the order is as defined by the clustering columns or the reverse otherwise, the order is defined by the CLUSTERING ORDER option and the reversed one.

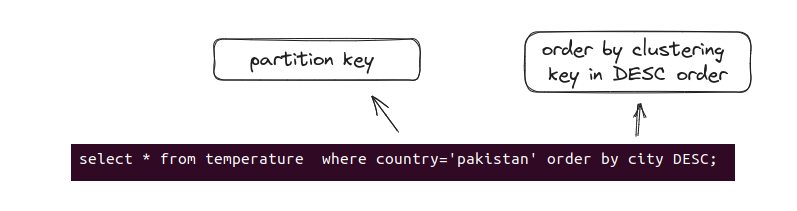
**Syntex:**

| SELECT \* FROM my\_table WHERE partition\_key1 = 'value1'  ORDER BY clustering\_column1 ASC  SELECT \* FROM my\_table WHERE partition\_key1 = 'value1'  ORDER BY clustering\_column1 DESC;; |
| --- |

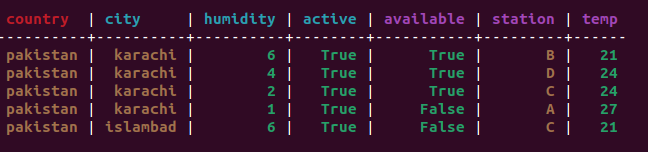
**Example**

Table before implemented ordering





If your query run successfully then the result will be like this



Here you can see we have sorted the whole table according to clustering key which is city and its in Descending order we can also do it in ascending order by ASC

Note: it can only be applied on clustering column and it applied within the partition key