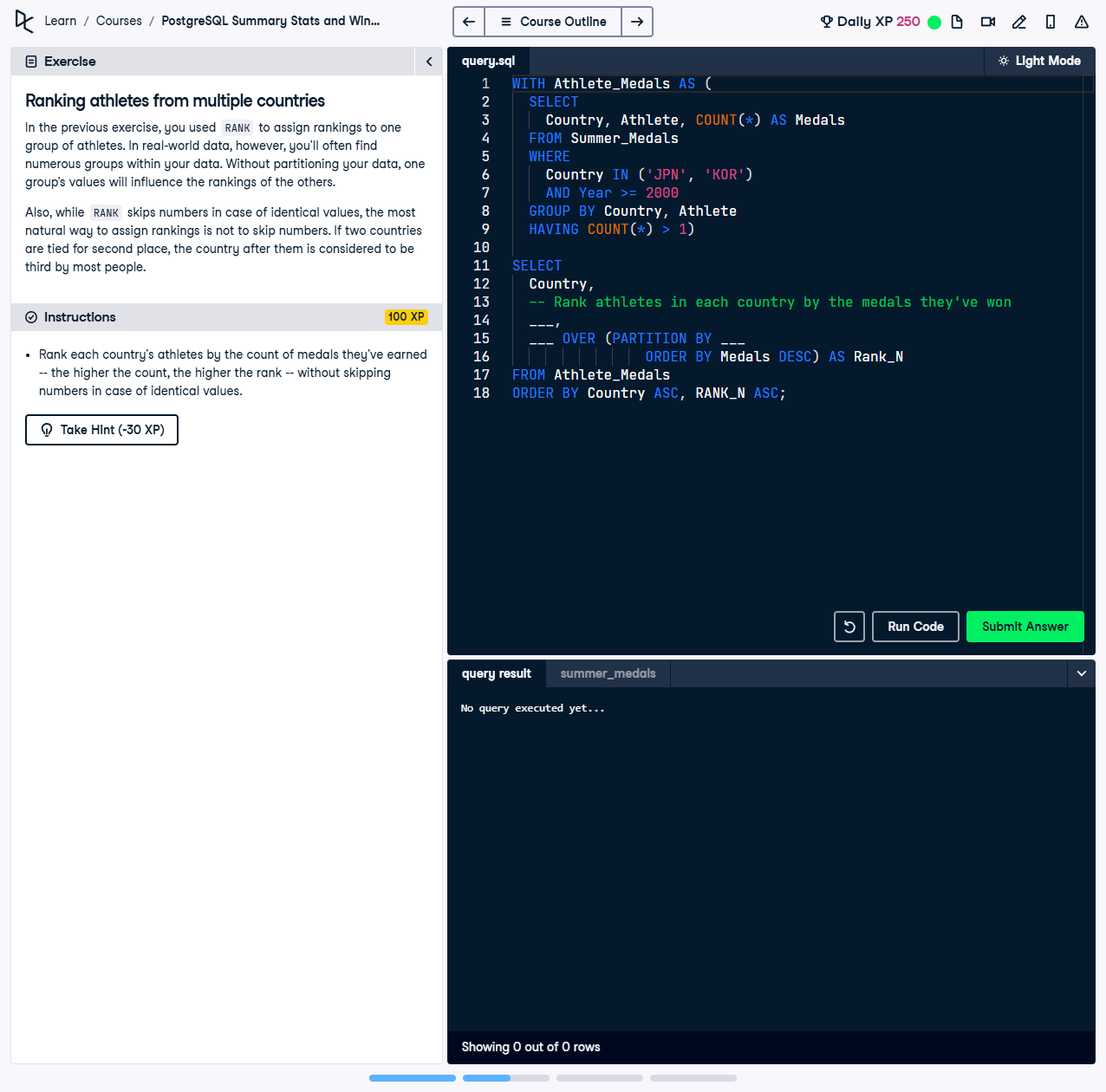
# Ranking Athletes with Dense Rank



In the previous exercise, you used `RANK` to assign rankings to one group of athletes. In real-world data, however, you’ll often find numerous groups within one dataset. Without partitioning your data, one group’s values will influence the rankings of the others. Using `DENSE\_RANK()` ensures rankings do not skip numbers for ties.

This exercise involves using a Common Table Expression (CTE) and the `DENSE\_RANK()` function to rank athletes within multiple countries based on their total medal count. `DENSE\_RANK()` ensures a continuous ranking sequence without skipped numbers for tied values.

## Correct Answer

WITH Athlete\_Medals AS (  
 SELECT  
 Country,  
 Athlete,  
 COUNT(\*) AS Medals  
 FROM Summer\_Medals  
 WHERE   
 Country IN ('JPN', 'KOR') AND  
 Year >= 2000  
 GROUP BY Country, Athlete  
 HAVING COUNT(\*) > 1  
)  
SELECT  
 Country,  
 Athlete,  
 Medals,  
 DENSE\_RANK() OVER (PARTITION BY Country ORDER BY Medals DESC) AS Rank\_N  
FROM Athlete\_Medals  
ORDER BY Country ASC, Rank\_N ASC;

Explanation of the query:

1. `WITH Athlete\_Medals AS (...)`: This CTE calculates the total number of medals won by athletes from specified countries (`JPN` and `KOR`) since 2000. It filters athletes with more than one medal using `HAVING COUNT(\*) > 1`.

2. `DENSE\_RANK() OVER (PARTITION BY Country ORDER BY Medals DESC) AS Rank\_N`: The `DENSE\_RANK()` function ranks athletes within each country separately, based on their medal count in descending order, ensuring no skipped numbers for ties.

3. `ORDER BY Country ASC, Rank\_N ASC`: Ensures that the results are sorted by country and then by rank, presenting a clear and organized output.