Documentation: Finding the Nth Highest Salary in a DataFrame

Objective

The goal is to determine the Nth highest unique salary from a dataset containing employee salaries. If the Nth highest salary does not exist due to fewer unique salary entries than N, the function should return null.

Function Overview

The function nth_highest_salary(employee: pd.DataFrame, N: int) -> pd.DataFrame is designed to:

- **Input:** Accept a pandas DataFrame representing the employee salary data and an integer N representing the Nth position.
- **Output:** Return a pandas DataFrame with a single entry corresponding to the Nth highest salary. If the Nth salary does not exist, the function returns null.

Input Parameters

1. **employee:**

- ***Type:*** pd.DataFrame
- ***Description:*** A DataFrame containing the salary information of employees. It must have at least two columns:
- ***id:*** The unique identifier for each employee (not used in the calculation but part of the DataFrame schema).
- ***salary:*** Integer values representing the salaries of employees. The column can contain duplicate values.

2. **N:**

- ***Type:*** int
- ***Description:*** The rank of the highest salary to retrieve (e.g., 1 for the highest, 2 for the second-highest, etc.).

Output

- **Type:** pd.DataFrame
- **Description:** A DataFrame with one column labeled "getNthHighestSalary(N)" where N is the input parameter. The DataFrame contains a single row:
- If the Nth highest salary exists, the row contains that salary.
- If there are fewer than N unique salaries or N is non-positive, the row contains None (equivalent to null in SQL-like contexts).

Methodology

1. **Extract Unique Salaries:**
   - Convert the salary column to a set to automatically remove duplicate salary values, ensuring that only unique salaries are considered in subsequent steps.

2. **Sort Unique Salaries:**
   - Convert the set back to a list and sort it in ascending order. Sorting is required to easily access the Nth highest salary using Python's negative indexing feature.

3. **Handle Edge Cases:**
   - *Check for several conditions:*
     - If N is greater than the number of unique salaries, the function should return None.
     - If N is less than or equal to 0, the function should also return None.
     - If the list of unique salaries is empty (no salaries present), return None.

4. **Retrieve the Nth Highest Salary:**
   - If the Nth highest salary exists, use negative indexing to access it (c[-N] where c is the sorted list of unique salaries).

5. **Prepare the Result:**
   - Construct the output DataFrame with the column name formatted as "getNthHighestSalary(N)" and populate it with the retrieved value (None or the Nth highest salary).

Edge Cases and Considerations

- **Empty DataFrame:** If the employee DataFrame has no entries, the function should return None.
- **All Salaries Are Identical:** If all salary values are the same, there will only be one unique salary. Any N greater than 1 should return None.
- **Non-Positive N Values:** If N is 0 or negative, the function should return None since an Nth highest does not conceptually make sense.
- **More Requests Than Unique Salaries:** If the number of unique salaries is less than N, the function should return None.

Example 1: When there are enough unique salaries

Input:

DataFrame with salaries [100, 200, 300]

N = 2

Output:

DataFrame with the value 200 (the 2nd highest salary)

Example 2: When the requested Nth salary does not exist

Input:

DataFrame with salaries [100, 200, 300]

N = 4

Output:

DataFrame with None (since there are only 3 unique salaries)

Performance Considerations

- **Time Complexity:** The function performs a sort operation, which has a time complexity of O(n log n), where n is the number of unique salaries.

- **Space Complexity:** The function uses additional space for storing unique salaries in a set and list. The overall space complexity is O(n) for the auxiliary data structures.

This documentation provides a complete overview of the function's logic, handling of various scenarios, and example usage, ensuring clarity for users and developers working with the salary data.