DESCRIPTION

Table: Employee

+-------------+---------+

| Column Name | Type |

+-------------+---------+

| empId | int |

| name | varchar |

| supervisor | int |

| salary | int |

+-------------+---------+

empId is the column with unique values for this table.

Each row of this table indicates the name and the ID of an employee in addition to their salary and the id of their manager.

Table: Bonus

+-------------+------+

| Column Name | Type |

+-------------+------+

| empId | int |

| bonus | int |

+-------------+------+

empId is the column of unique values for this table.

empId is a foreign key (reference column) to empId from the Employee table.

Each row of this table contains the id of an employee and their respective bonus.

Write a solution to report the name and bonus amount of each employee with a bonus **less than** 1000.

Return the result table in **any order**.

The result format is in the following example.

Example 1:

Input:

Employee table:

```
+-------+--------+------------+--------+
| empId | name   | supervisor | salary |
+-------+--------+------------+--------+
| 3     | Brad   | null       | 4000   |
| 1     | John   | 3          | 1000   |
| 2     | Dan    | 3          | 2000   |
| 4     | Thomas | 3          | 4000   |
+-------+--------+------------+--------+
```

Bonus table:

```
+-------+-------+
| empId | bonus |
+-------+-------+
| 2     | 500   |
| 4     | 2000  |
+-------+-------+
```

Output:

```
+------+-------+
| name | bonus |
+------+-------+
| Brad | null  |
| John | null  |
| Dan  | 500   |
+------+-------+
```

SOLUTION

Option 1:

- Join two tables using merge() and 'left' join method
- Include nulls in ‘bonus’ using isnull()

```
import pandas as pd

def employee_bonus(employee: pd.DataFrame, bonus: pd.DataFrame) -> pd.DataFrame:
    df = employee.merge(bonus, how = 'left', on = 'empId')
    return df.loc[(df['bonus'] < 1000) | df['bonus'].isnull(), ['name', 'bonus']]
```