Evaluate Boolean Expression
SQL Schema
Table Variables:

Column Name	Type
name	varchar
value	int

name is the primary key for this table.
This table contains the stored variables and their values.

Table Expressions:

Column Name	Type
left_operand	varchar
operator	enum
right_operand	varchar

(left_operand, operator, right_operand) is the primary key for this table.
This table contains a boolean expression that should be evaluated.

operator is an enum that takes one of the values ('<', '>', '=')
The values of left_operand and right_operand are guaranteed to be in the Variables table.

Write an SQL query to evaluate the boolean expressions in Expressions table.
Return the result table in any order.
The query result format is in the following example.

Example 1:
Input:
Variables table:

name	value
x	66
y	77

Expressions table:

left_operand	operator	right_operand
x	>	y
x	<	y

```
| x          | =      | y          |
| y          | >      | x          |
| y          | <      | x          |
| x          | =      | x          |
+-------------+---------+--------------+
```

Output:

```
+-------------+---------+--------------+-------+
| left_operand | operator | right_operand |
value |
+-------------+---------+--------------+-------+
| x          | >      | y          | false |
| x          | <      | y          | true  |
| x          | =      | y          | false |
| y          | >      | x          | true  |
| y          | <      | x          | false |
| x          | =      | x          | true  |
```

Program:

```
import pandas as pd
variables_data = {
    "name": ["x", "y"],
    "value": [66, 77]
}
expressions_data = {
    "left_operand": ["x", "x", "x", "y", "y", "x"],
    "operator": [">", "<", "=", ">", "<", "="],
    "right_operand": ["y", "y", "y", "x", "x", "x"]
}
```

```
variables_df = pd.DataFrame(variables_data)
expressions_df =
pd.DataFrame(expressions_data)
merged_df = expressions_df.merge(variables_df,
left_on="left_operand", right_on="name",
suffixes=('', '_left'))
merged_df = merged_df.merge(variables_df,
left_on="right_operand", right_on="name",
suffixes=('', '_right'))
def evaluate_expression(row):
    if row['operator'] == '>':
        return row['value'] > row['value_right']
    elif row['operator'] == '<':
        return row['value'] < row['value_right']
    elif row['operator'] == '=':
        return row['value'] == row['value_right']
merged_df['value'] =
merged_df.apply(evaluate_expression, axis=1)
result_df = merged_df[['left_operand',
'operator', 'right_operand', 'value']]
print(result_df)
```

Output:

```
C:\Users\srika\Desktop\CSA0863\pythonProject\.venv\Scripts\python.exe "C:\Users\srika\Desktop\CSA0863\pythonProject\DAA COADS.PYTHON\program 54.py"
  left_operand operator right_operand  value
0            x        >             y  False
1            x        <             y   True
2            x        =             y  False
3            y        >             x   True
4            y        <             x  False
5            x        =             x   True

Process finished with exit code 0
```

Time complexity:

O(N*M)