**58)** . 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.

CODE:

```
variables = [
    {"name": "x", "value": 66},
    {"name": "y", "value": 77}
]

expressions = [
    {"left_operand": "x", "operator": ">", "right_operand": "y"},
    {"left_operand": "x", "operator": "<", "right_operand": "y"},
    {"left_operand": "x", "operator": "=", "right_operand": "y"},
    {"left_operand": "y", "operator": ">", "right_operand": "x"},
    {"left_operand": "y", "operator": "<", "right_operand": "x"},
    {"left_operand": "x", "operator": "=", "right_operand": "x"}
]

variables_dict = {var['name']: var['value'] for var in variables}

def evaluate_expression(left_operand, operator, right_operand):
    left_value = variables_dict[left_operand]
    right_value = variables_dict[right_operand]
    if operator == '>':
        return left_value > right_value
    elif operator == '<':
        return left_value < right_value
    elif operator == '=':
        return left_value == right_value
    else:
        raise ValueError(f"Unsupported operator: {operator}")

results = []
for expr in expressions:
    result = evaluate_expression(expr['left_operand'], expr['operator'],
expr['right_operand'])
    results.append({
```

```
        "left_operand": expr['left_operand'],
        "operator": expr['operator'],
        "right_operand": expr['right_operand'],
        "value": result
    })

for result in results:
    print(result)
```

OUTPUT:

```
C:\WINDOWS\system32\cmd.

{'left_operand': 'x', 'operator': '>', 'right_operand': 'y', 'value': False}
{'left_operand': 'x', 'operator': '<', 'right_operand': 'y', 'value': True}
{'left_operand': 'x', 'operator': '=', 'right_operand': 'y', 'value': False}
{'left_operand': 'y', 'operator': '>', 'right_operand': 'x', 'value': True}
{'left_operand': 'y', 'operator': '<', 'right_operand': 'x', 'value': False}
{'left_operand': 'x', 'operator': '=', 'right_operand': 'x', 'value': True}
Press any key to continue . . .
```

TIME COMPLEXITY : **O(m+n)**