📚 Module 24-27: Boolean Logic & Truth Tables - Complete Notes

🎯 What You'll Learn

Master **Boolean logic** — AND, OR, NOT, truth tables, and short-circuit evaluation.

📖 Concept Explained

Boolean Operators

```
AND: Both must be true
    True AND True = True
    True AND False = False

OR: At least one must be true
    True OR False = True
    False OR False = False

NOT: Flip the value
    NOT True = False
    NOT False = True
```

Truth Table

A	B	A AND B	A OR B	NOT A
T	T	T	T	F
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

💻 Programming Connection

Code Examples

```python
# Example 1: Basic Boolean

print(True and True)    # True
print(True and False)  # False
print(True or False)    # True
print(not True)         # False
```

```python
# Example 2: Practical Conditions

def can_access(user):
    """Complex access control logic"""
    is_admin = user.get('role') == 'admin'
    is_owner = user.get('is_owner', False)
    is_authenticated = user.get('authenticated', False)

    # Must be authenticated AND (admin OR owner)
    return is_authenticated and (is_admin or is_owner)

user1 = {'role': 'admin', 'authenticated': True}
print(can_access(user1))  # True
```

```
# Example 3: Short-Circuit Evaluation

# AND: If first is False, second not evaluated
result = False and expensive_function()  # expensive_function() never runs!

# OR: If first is True, second not evaluated
result = True or expensive_function()    # expensive_function() never runs!

# Safe access pattern
user = None
name = user and user.get('name')  # No error, returns None
```

```
# Example 4: Defaults with OR

name = "" or "Anonymous"      # "Anonymous" (empty string is falsy)
count = 0 or 10               # 10 (0 is falsy - be careful!)
value = None or "default"     # "default"
```

```
# Example 5: all() and any() - Collective AND/OR

conditions = [True, True, True]
print(all(conditions))  # True (AND across list)

conditions = [True, False, True]
print(any(conditions))  # True (OR across list)

# Practical
def validate_form(fields):
    return all(field.strip() for field in fields)  # All non-empty

def has_errors(results):
    return any(r.get('error') for r in results)  # Any error
```

```
# Example 6: De Morgan's Laws

# NOT (A AND B) = (NOT A) OR (NOT B)
# NOT (A OR B) = (NOT A) AND (NOT B)
```

```python
a, b = True, False

# These are equivalent:
print(not (a and b))        # True
print((not a) or (not b))   # True

# These are equivalent:
print(not (a or b))         # False
print((not a) and (not b))  # False
```

🧪 SDET/Testing Application

```python
# SDET Scenario: Request Retry Logic

def should_retry(response, config):
    """Determine if request should be retried"""
    is_retryable = response.status_code in [500, 502, 503]
    has_retries = config['attempt'] < config['max_retries']
    not_cancelled = not config.get('cancelled', False)

    return is_retryable and has_retries and not_cancelled

response = type('Response', (), {'status_code': 503})()
config = {'attempt': 1, 'max_retries': 3}
print(should_retry(response, config))  # True
```

```python
# SDET Scenario: Generate All Test Combinations

from itertools import product

def generate_boolean_test_cases(num_conditions):
    """Generate all True/False combinations"""
    return list(product([True, False], repeat=num_conditions))

# 3 conditions = 8 test cases
cases = generate_boolean_test_cases(3)
print(f"Total cases: {len(cases)}")  # 8
for case in cases:
    print(case)
```

🔑 Key Takeaways

✅ **AND** — All must be true
✅ **OR** — Any can be true
✅ **NOT** — Flip the value
✅ **Short-circuit** — Stop early if result is known

✅ **all()** — AND across collection
✅ **any()** — OR across collection

💾 *Save as:* `Module_24_27_Boolean_Logic.md`

Phase 5 Complete! 🚀