Here’s a concise document addressing each review point, explaining how I fixed it, what the new code does, and why it’s better:

GitHub Link: <https://github.com/ankita391989/PEI-Task>

**Unit Test Review: Fixes and Improvements**

1. Tests Written as Validations

\*\*Review\*\*

Tests were only validating outputs, not truly testing logic.

\*\*Fix\*\*

- Rewrote tests to use controlled input data and assert expected outputs.

- Example: Instead of just checking if a value is non-negative, the test now sets up a DataFrame with known values and checks the exact result.

- Ensures the function works as intended for specific scenarios.

- Catches logic errors, not just data issues.

-----------------------------------------------------------------------------------------------------------------------------

2. Lack of Data Setup in Tests

\*\*Review\*\*

Tests did not set up input data, so it was unclear why a test should pass or fail.

\*\*Fix\*\*

- Added fixtures and explicit DataFrame setups in each test.

- Example:

```python

@pytest.fixture

def sample\_orders\_df():

return pd.DataFrame({'quantity\_sold': [1, 2, -1]})

```

- Tests are now reproducible and independent of external data.

-Makes the purpose and expected outcome of each test clear

-----------------------------------------------------------------------------------------------------------------------------

3. Tests Not Really Testing Anything

\*\*Review\*\*

Some tests (e.g., for missing columns) did not assert real errors or outcomes.

\*\*Fix\*\*

- Updated tests to use `pytest.raises` to assert that the correct exceptions are raised when expected.

- Example:

def test\_error\_on\_missing\_column\_rename():

    import pandas as pd

    df = pd.DataFrame({"A": [1], "B": [2]})

    with pytest.raises(KeyError):

        df.rename(columns={"C": "D"}, errors="raise")

AND

def test\_error\_on\_missing\_column\_in\_join(sample\_customers\_df, customer\_metrics, missing\_col):

    import pyspark

    with pytest.raises(pyspark.sql.utils.AnalysisException):

        sample\_customers\_df.join(customer\_metrics, on=missing\_col, how='left').count()

AND

@pytest.mark.parametrize("agg\_col", ['NonExistentColumn', 'AnotherMissing'])

def test\_error\_on\_invalid\_agg\_column(sample\_orders\_df, agg\_col):

    import pyspark

    with pytest.raises(pyspark.sql.utils.AnalysisException):

        sample\_orders\_df.groupby('customer\_id').agg(F.sum(agg\_col)).count()

- Confirms that error handling works as intended.

- Prevents silent failures or false positives.

-----------------------------------------------------------------------------------------------------------------------------

\*\*4. Assertion Logic Duplicates Production Logic\*\*

\*\*Review\*\*

Tests repeated the same logic as the code under test, so they didn’t catch errors.

\*\*Fix\*\*

- Tests now use hardcoded expected outputs instead of duplicating logic.

- Example:

```python

assert cleaned\_name == "John Doe"

```

- Ensures the test is independent and will fail if the production logic is incorrect.

-----------------------------------------------------------------------------------------------------------------------------

### \*\*5. Incomplete Column Name Sanitization\*\*

\*\*Review\*\*

Sanitization only replaced spaces, not all special characters or cases.

\*\*Fix\*\*

- Implemented a comprehensive sanitization function:

```python

def sanitize\_column(col):

import re

col = col.strip().lower()

col = re.sub(r'[\s\-]+', '\_', col)

col = re.sub(r'[^a-z0-9\_]', '', col)

return col

```

- Applied this to all DataFrames immediately after loading.

-----------------------------------------------------------------------------------------------------------------------------

6. Date Parsing Logic Issues

\*\*Review\*\*

Using a fixed format (`dd/mm/yyyy`) could cause parsing errors with inconsistent data.

\*\*Fix\*\*

- Used flexible date parsing (`d/M/yyyy`) to handle both single and double-digit days/months.

- Standardized date columns immediately after loading.

- Prevents parsing errors and ensures date operations work reliably.

--------------------------------------------------------------------------------------------------------------------------

7. Duplicate Tests in 3b

\*\*Review\*\*

Schema validation and column name validation were tested separately, causing duplication.

\*\*Fix\*\*

- Removed redundant column name validation tests if schema validation already covers them.

- Reduces test maintenance.

- Keeps the test suite focused and efficient.

-----------------------------------------------------------------------------------------------------------------------------