Airflow with Postgres and Spark

By Dhandapani Yedappalli Krishnamurthi Sep 8, 2025

Step 1: Configure Postgres Connection in Airflow

In Airflow UI → **Admin → Connections** → Add a new connection:

- **Conn Id:** `local_postgres`
- **Conn Type:** `Postgres`
- **Host:** `localhost`
- **Schema:** your DB name
- **Login:** your username
- **Password:** your password
- **Port:** 5432

Levels of Testing in Airflow

1. **DAG validation (static checks)**
    - Does the DAG parse correctly?
    - Are dependencies (`>>`, `<<`) set up correctly?

```
airflow dags list
airflow dags show etl_pipeline --save dag.png
```

Or in PyCharm:

```
from airflow.models import DagBag

def test_dag_integrity():
    dag_bag = DagBag()
    assert len(dag_bag.import_errors) == 0, f"DAG import
errors: {dag_bag.import_errors}"
```

Unit Testing Tasks (without Airflow scheduler)

Each `PythonOperator` wraps a Python function. You can test those functions directly with `pytest` like normal Python code.

Example (`etl_pipeline.py`):

```
def extract():
    # your code
    return True
```

Test in `tests/test_extract.py`:

```
from dags.etl_pipeline import extract

def test_extract_creates_csv(tmp_path):
    result = extract()
    assert result is True
```

Task Instance Testing (simulate Airflow execution)

Airflow provides a way to run a **single task instance** without running the scheduler:

```
airflow tasks test etl_pipeline extract 2025-09-07
airflow tasks test etl_pipeline transform 2025-09-07
airflow tasks test etl_pipeline load 2025-09-07
```

3. This executes the operator as if the scheduler triggered it, using the execution date you provide.

Integration Testing (entire DAG)

Run the DAG manually:

```
airflow dags trigger etl_pipeline
airflow dags state etl_pipeline <dag_run_id>
```

4. Or from PyCharm, you can open **Airflow CLI run configurations** and trigger DAG runs.

End-to-End Validation (data checks)

Since your pipeline loads into Postgres, you can validate results:

```
import psycopg2

def test_people_table_has_rows():
    conn = psycopg2.connect("dbname=airflow user=airflow
password=airflow host=localhost port=5433")
    cur = conn.cursor()
    cur.execute("SELECT COUNT(*) FROM people;")
    count = cur.fetchone()[0]
    assert count > 0
    cur.close()
    conn.close()
```

Summary

- **Unit test** your Python functions (ETL steps).
- **`airflow tasks test`** → run operators without scheduler.
- **`airflow dags trigger`** → integration testing full DAG.
- **SQL/data assertions** → verify final results.

Contents

- **`test_dag_integrity.py`** → checks if DAGs load without errors

- **`test_extract.py`** → unit tests for `extract()` and `transform()` functions
- **`test_postgres_load.py`** → integration test that verifies rows exist in Postgres after running DAG

How to Run Tests

Install pytest inside your Airflow environment:

```
pip install pytest pytest-mock
```

Run tests in PyCharm or CLI:

```
pytest -v tests/
```

To run only integration tests:

```
pytest -m integration
```
