Capstone project AOTA Test Report

1. Unit Test

A unit test is a scripted code-level test, written in Python to verify a small "unit" of functionality.

With this project, we use Python Unit Testing Framework to test the basic functionalities of some two utility classes, **sparkConnector** and **ut_spark.** Here are two python scripts:

1）test_parkconnector.py：to test the only main method of class sparkConnector

Testing method	**Method to be tested**	**Function**
test_get_spark	.get_spark()	Verify that 'get_spark()' method can create a spark session

2）test_utspark.py：to test methoes of class ut_spark

Testing method	**Method to be tested**	**Function**
test_get_sparksession	get_sparksession	Verify that 'get_sparksession' will return a spark session
test_df_read_from_csv	df_read_from_csv	Verify that method 'df_read_from_csv' can read a csv file file and create and populate spark dataframe with the data reads
test_write_read_parquet	df_read_from_parquet df_write_to_file	Verify that 'df_write_to_file' method can correctly write data to a parquet file and 'df_read_from_parquet' can correctly read data from that parquest file

Here are the test results:

```
C:\Users\huang\OneDrive\14-code\sprb-project-code\pyspark_code\capstone_ATOA_cloud>pytest -v
============================================= test session starts =============================================
platform win32 -- Python 3.10.2, pytest-7.0.0, pluggy-1.0.0 -- C:\Python\Python310\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\huang\OneDrive\14-code\sprb-project-code\pyspark_code\capstone_ATOA_cloud
plugins: anyio-3.5.0
collected 4 items

test_parkconnector.py::test_get_spark PASSED                                                          [ 25%]
test_utspark.py::test_get_sparksession PASSED                                                         [ 50%]
test_utspark.py::test_df_read_from_csv PASSED                                                         [ 75%]
test_utspark.py::test_write_read_parquet PASSED                                                       [100%]

============================================== 4 passed in 9.49s ==============================================
SUCCESS: The process with PID 14252 (child process of PID 6020) has been terminated.
SUCCESS: The process with PID 6020 (child process of PID 4352) has been terminated.
SUCCESS: The process with PID 4352 (child process of PID 6036) has been terminated.
```

2. Test on ETL process（also called 'cleanse/transform' process）

   ' ETL process ' is one of the two transformation processes in the system. It will be performed on four data sets: flight, plane, carrier and airport: The logic is as follows:

Dataset	Operations
Flight	.Exclude 'canceled'/'diverted' flights .Add one id column and drop a few columns
Plane	.Filter out the blank records
Airport	Add one column
Carrier	.Nothing needs to be done

ETL on airport and carrier don't do much things . So we will test ETL process on 'plane'. Test on flight ETL will be included the next section.

Test data set: include 6 records. 3 of them are blank

	tailnum	type	manufacturer	issue_date	model	status	aircraft_type	engine_type	year
0	N050AA	None	None	None	None	None	None	None	None
1	N083AA	None	None	None	None	None	None	None	None
2	N084AA	None	None	None	None	None	None	None	None
3	N10156	Corporation	EMBRAER	02/13/2004	EMB-145XR	Valid	Fixed Wing Multi-Engine	Turbo-Fan	2004
4	N102UW	Corporation	AIRBUS INDUSTRIE	05/26/1999	A320-214	Valid	Fixed Wing Multi-Engine	Turbo-Fan	1998
5	N10323	Corporation	BOEING	07/01/1997	737-3TO	Valid	Fixed Wing Multi-Engine	Turbo-Jet	1986

Expected: after ETL process, there are only 3 records in processed layer

Result:
Processed data in 'processed layer':

	tailnum	type	manufacturer	issue_date	model	status	aircraft_type	engine_type	year
0	N10156	Corporation	EMBRAER	02/13/2004	EMB-145XR	Valid	Fixed Wing Multi-Engine	Turbo-Fan	2004
1	N102UW	Corporation	AIRBUS INDUSTRIE	05/26/1999	A320-214	Valid	Fixed Wing Multi-Engine	Turbo-Fan	1998
2	N10323	Corporation	BOEING	07/01/1997	737-3TO	Valid	Fixed Wing Multi-Engine	Turbo-Jet	1986

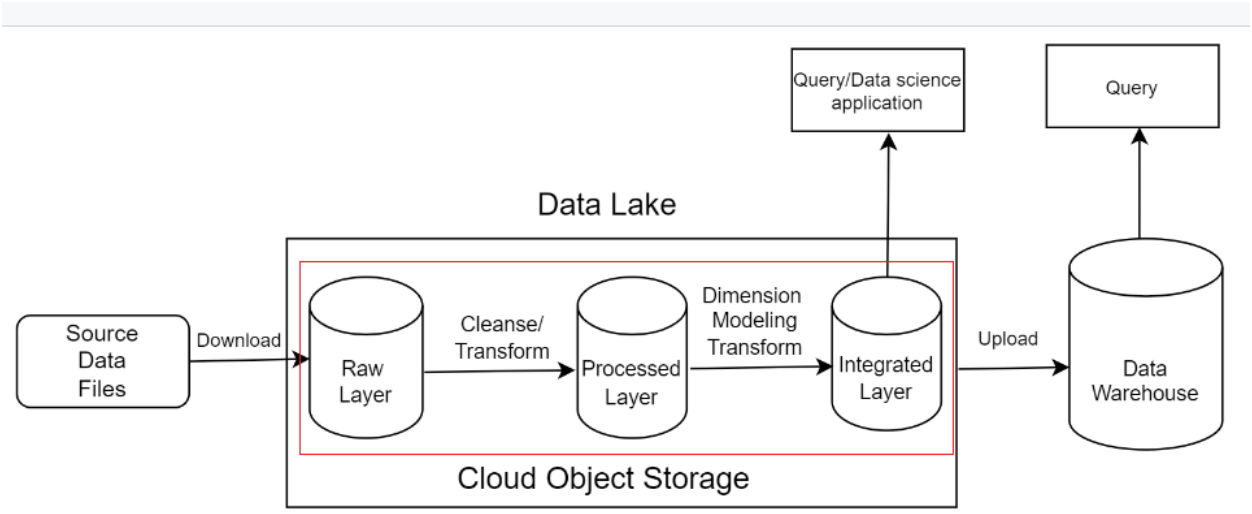
Conclusion: the test results meet expectations.

I attached 'test_etl_plane.ipynb' which has more detailed information.

3. Integration testing for 'flight' data movement

As data movement from 'raw layer' to 'integrated layer' contains almost all the transformation logic, and 'flight' dataset is the major dataset, we design a test case to test the processing of

'flight' from raw layter to integration layer.



I attached a jupyter notebook file ('test_endToEnd-raw-to-integrated.ipynb') which has more detailed information. Here is the introduction of the testing process:

Test dataset:

We populate 10 testing records into the 'flight' table at 'raw layer' ("C:/demo/capstone/raw/2000_test.csv"), as below

	Year	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	FlightNum	...	TaxiIn	TaxiOut	Cancelled	Cancellation
0	2000	1	28	5	1647	1647	1906	1859	HP	154	...	15	11	0	
1	2000	1	29	6	1648	1647	1939	1859	HP	154	...	5	47	0	
2	2000	1	30	7	NA	1647	NA	1859	HP	154	...	0	0	1	
3	2000	1	31	1	1645	1647	1852	1859	HP	154	...	7	14	0	
4	2000	1	1	6	842	846	1057	1101	HP	609	...	3	8	0	
5	2000	1	2	7	849	846	1148	1101	HP	609	...	8	24	0	
6	2000	1	3	1	844	846	1121	1101	HP	609	...	6	27	0	
7	2000	1	1	6	1702	1657	1912	1908	HP	611	...	5	13	0	
8	2000	1	2	7	1658	1657	1901	1908	HP	611	...	3	7	0	
9	2000	1	3	1	1656	1657	1922	1908	HP	611	...	5	20	0	

Expected result:

1. Cleansing and Transformation (ETL process)

This step will read data from 'raw layer', do the cleansing and transformation, and saved the processed results to 'processed layer'. For 'flights', the main cleansing process is to filter out the canceled records.

Expected: 1 record to be filtered out.

2. Dimensional Modeling Transformation (Integration process)
   This step will read data from 'processed layer' , conduct dimension modeling, and finally create two dimension tables dim_date, dim_origin_dest, and one fact table, fact_flight, and save them to the 'integrated layer'.

   Expected:
   1) As the test dataset contains data of 6 different dates (excluding canceled record), **dim_date** should have **6** records.
   2) As the test dataset contains one combination of 'origin' and 'dest', **dim_origin_dest** table should have **1** record
   3) As the test dataset contains 9 uncancelled flights, **'fact_flight'** table should have **9** records.

Result:

(1) Data at 'Processed layer' (after cleansing/transformation).
There are **9** records. One canceled flight record was filtered out.

	Month	DayofMonth	DayOfWeek	DepTime	CRSDepTime	ArrTime	CRSArrTime	UniqueCarrier	FlightNum	TailNum	AirTime	ArrDelay	DepDelay	Origin	D
0	1	28	5	1647	1647	1906	1859	HP	154	N808AW	233	7	0	ATL	P
1	1	29	6	1648	1647	1939	1859	HP	154	N653AW	239	40	1	ATL	P
2	1	31	1	1645	1647	1852	1859	HP	154	N806AW	226	-7	-2	ATL	P
3	1	1	6	842	846	1057	1101	HP	609	N158AW	244	-4	-4	ATL	P
4	1	2	7	849	846	1148	1101	HP	609	N656AW	267	47	3	ATL	P
5	1	3	1	844	846	1121	1101	HP	609	N803AW	244	20	-2	ATL	P
6	1	1	6	1702	1657	1912	1908	HP	611	N652AW	232	4	5	ATL	P
7	1	2	7	1658	1657	1901	1908	HP	611	N807AW	233	-7	1	ATL	P
8	1	3	1	1656	1657	1922	1908	HP	611	N807AW	241	14	-1	ATL	P

(2) Data at 'Integrated layer'

Table	Detail
dim_date	File_path: C:/demo/testaota/integrated/dim_date' Row count: 6

<table>
<tr><td></td><td>

```
+--------+----+-----+----------+---------+
| date_id|Year|Month|DayofMonth|DayOfWeek|
+--------+----+-----+----------+---------+
|20000128|2000|    1|        28|        5|
|20000129|2000|    1|        29|        6|
|20000131|2000|    1|        31|        1|
|20000101|2000|    1|         1|        6|
|20000102|2000|    1|         2|        7|
|20000103|2000|    1|         3|        1|
+--------+----+-----+----------+---------+
```

</td></tr>
<tr><td>dim_origin_dest</td><td>File_path: C:/demo/testaota/integrated/dim_origin_dest
Row count: 1

	orig_dest_id	origin	dest	distance	orig_airport	orig_city	orig_state	orig_country	dest_airport	dest_city	dest_state	dest_country
0	PHX_ATL	ATL	PHX	1587	William B Hartsfield-Atlanta Intl	Atlanta	GA	USA	Phoenix Sky Harbor International	Phoenix	AZ	USA

</td></tr>
<tr><td>fact_flight</td><td>File_path: C:/demo/testaota/integrated/fact_flight
Row count: 9

	month	seq_id	date_id	orig_dest_id	flightNum	carrier_id	plane_id	DepTime	scheduled_deptime	ArrTime	scheduled_arrtime	ArrDelay	year
0	1	4	20000102	PHX_ATL	609	HP	N656AW	849	846	1148	1101	47	2000
1	1	5	20000103	PHX_ATL	609	HP	N803AW	844	846	1121	1101	20	2000
2	1	8	20000103	PHX_ATL	611	HP	N807AW	1656	1657	1922	1908	14	2000
3	1	0	20000128	PHX_ATL	154	HP	N808AW	1647	1647	1906	1859	7	2000
4	1	1	20000129	PHX_ATL	154	HP	N653AW	1648	1647	1939	1859	40	2000
5	1	6	20000101	PHX_ATL	611	HP	N652AW	1702	1657	1912	1908	4	2000
6	1	2	20000131	PHX_ATL	154	HP	N806AW	1645	1647	1852	1859	-7	2000
7	1	3	20000101	PHX_ATL	609	HP	N158AW	842	846	1057	1101	-4	2000
8	1	7	20000102	PHX_ATL	611	HP	N807AW	1658	1657	1901	1908	-7	2000

</td></tr>
</table>

Conclusion: the test results meet expectations.