CIDM-5310 – Summary

This particular project covers the ability to pull in multiple data sources. The data sources accessed ended up being the following (I removed a few from the original list):

- Commercial Aviation - Involuntary Denied Boarding
- Commercial Aviation Mishandled Baggage and Mishandled Wheelchairs
- Airline Quarterly Financial Review - Majors
- Contiguous State City-Pair Markets That Average At Least 10 Passengers Per Day
- Consumer Airfare Report: Detailed Fare Information For Highest and Lowest Fare Markets Under 750 Miles

I was a little ambitious when it came to the last two items. I created the database structures for them and pulled the data in; however, I didn't get a chance to incorporate them as much as wanted to. For each of the items listed, I did pull the data in for a single year. Then I cross-loaded them into tables (in some cases, I normalized the data, such as airliner names and airports). I also added a table that will track the last time the data was pulled; if it was today, then it won't pull again. This way, we're not constantly pulling in data from the source.

I used several pandas data frame functions to accomplish this project. Specifically, I used grouping and massaging before inserting the data for "Mishandled Baggage and Mishandled Wheelchairs" and "Involuntary Denied Boarding." I also used some merging and column manipulation in different area's to facilitate the data analysis.

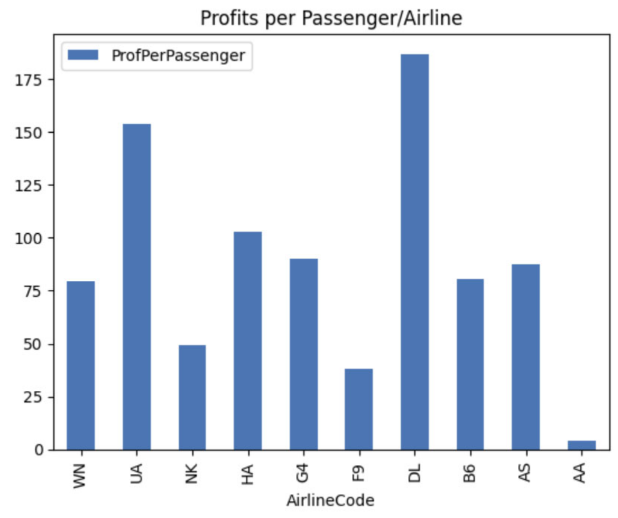
I also used the SQL functionality of the pandas to read and insert data into the database. I used SQL Alchemy to structure the data as well as models that I used to transform the raw data from the API. This allowed me to more efficiently fit the data into a consumable format.

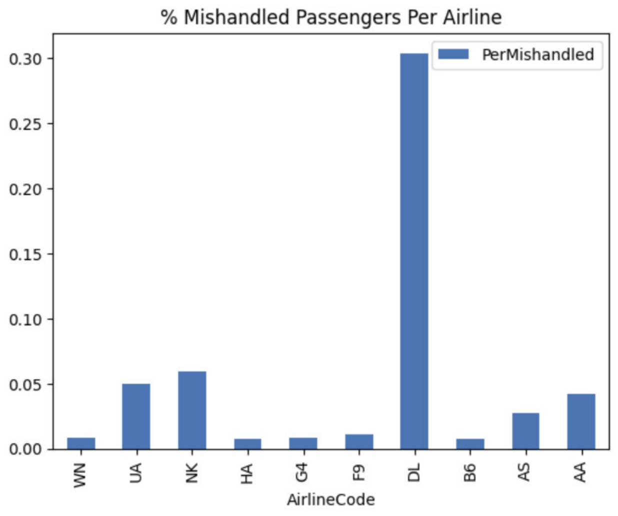
In the Analysis notebook, I created several scenarios to display the data that was queried from the tables. The primary work in the notebook was some merging and grouping that I applied to attempt to see if my points hypothesis that poor customer support leads to lower profits. I found some very interesting results. Specifically, Delta had the largest profit per customer but also the highest incidents of passenger mishandling. On the flip side, lost baggage didn't seem to follow this as starkly, and the data was shaped very similarly to the one for involuntary boarding. The graphs for these can be found in the Appendix.

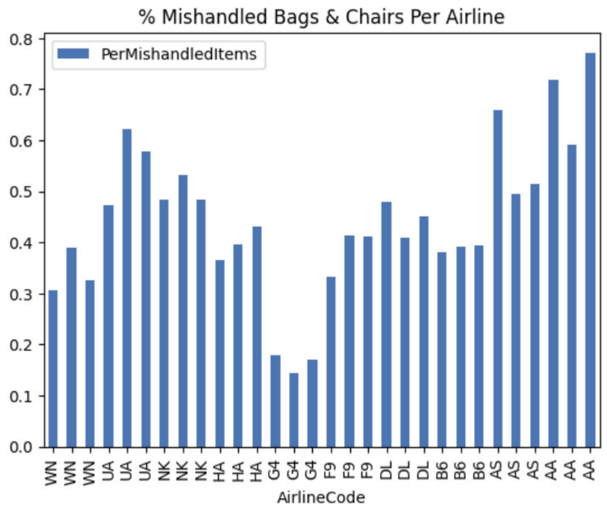
When loading the project onto my production system, I did have some issues. When running it locally, I simply ran a SQL lite server in memory only. When I introduced it to MySQL (MariaDB, actually), I did run into some difficulties. I was unaware of some adapters that needed to be set up just to get Python and the DB to communicate with each other. After a bit of Googling, I got things up and running successfully. I could also cross-validate that the data was getting inserted into the database via phpMyAdmin (see the Appendix for screenshots).

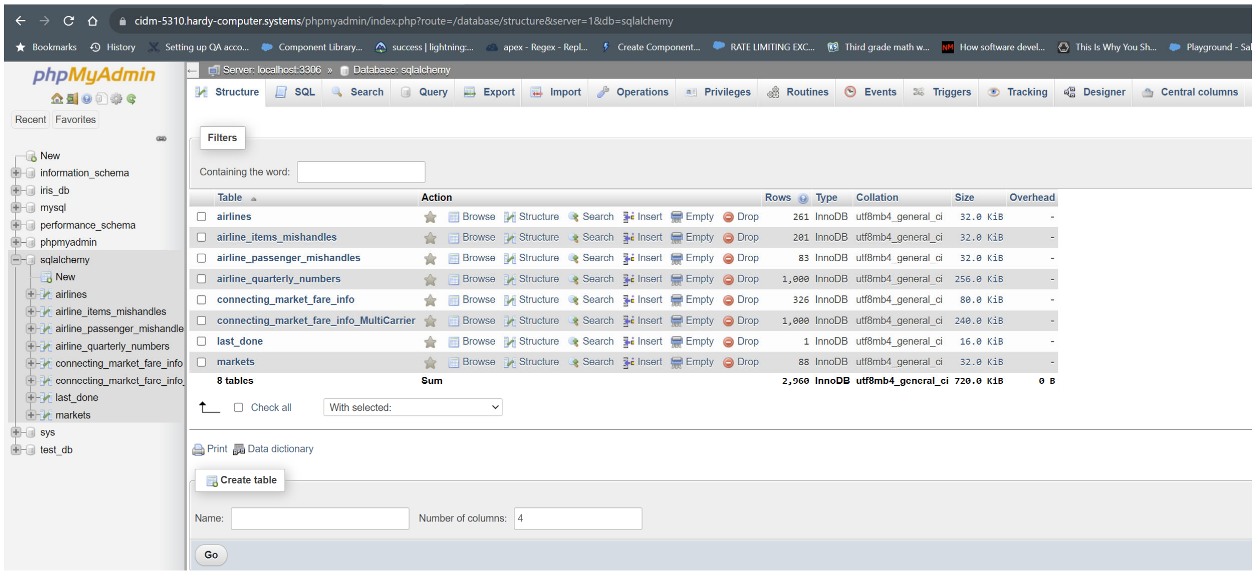
I believe this was a fairly good use of pandas as it allowed for multi-level data analysis that led to some interesting results. I believe it demonstrates a lot of the power of the platform as well. I did very little manipulation of the data via SQL; the majority of it was handled within data frames. The VPS setup also helped me dust off some skills that had been fallow for a while (took me a bit of time to get the key-based authentication setup on my machine for example).

Appendix


Profits per Passenger/Airline
ProfPerPassenger
175
150
125
100
75
50
25
0
WN
UA
NK
HA
G4
F9
DL
B6
AS
AA
AirlineCode


% Mishandled Passengers Per Airline
PerMishandled
0.30
0.25
0.20
0.15
0.10
0.05
0.00
WN
UA
NK
HA
G4
F9
DL
B6
AS
AA
AirlineCode


% Mishandled Bags & Chairs Per Airline
PerMishandledItems
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
WN
WN
WN
UA
UA
UA
NK
NK
NK
HA
HA
HA
G4
G4
G4
F9
F9
F9
DL
DL
DL
B6
B6
B6
AS
AS
AS
AA
AA
AA
AirlineCode



cidm-5310.hardy-computer.systems/phpmyadmin/index.php?route=/database/structure&server=1&db=sqlalchemy

Server: localhost:3306 » Database: sqlalchemy

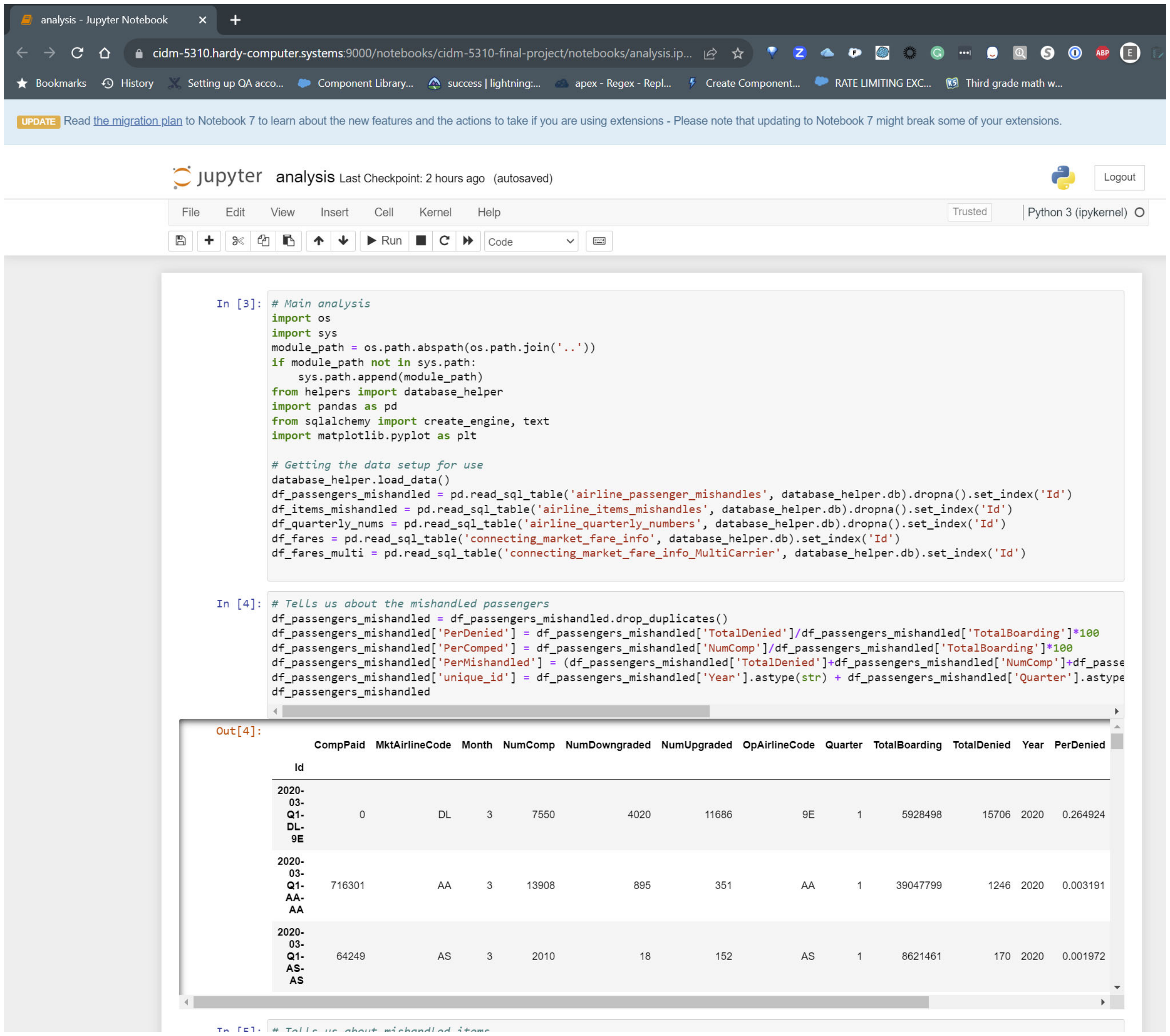
Filters

Containing the word:

	Table	Action	Rows	Type	Collation	Size	Overhead
☐	airlines	Browse Structure Search Insert Empty Drop	261	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	airline_items_mishandles	Browse Structure Search Insert Empty Drop	201	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	airline_passenger_mishandles	Browse Structure Search Insert Empty Drop	83	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	airline_quarterly_numbers	Browse Structure Search Insert Empty Drop	1,000	InnoDB	utf8mb4_general_ci	256.0 KiB	-
☐	connecting_market_fare_info	Browse Structure Search Insert Empty Drop	326	InnoDB	utf8mb4_general_ci	80.0 KiB	-
☐	connecting_market_fare_info_MultiCarrier	Browse Structure Search Insert Empty Drop	1,000	InnoDB	utf8mb4_general_ci	240.0 KiB	-
☐	last_done	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	markets	Browse Structure Search Insert Empty Drop	88	InnoDB	utf8mb4_general_ci	32.0 KiB	-
	8 tables	Sum	2,960	InnoDB	utf8mb4_general_ci	720.0 KiB	0 B

Create table

Name: Number of columns: 4



cidm-5310.hardy-computer.systems:9000/notebooks/cidm-5310-final-project/notebooks/analysis.ip...

UPDATE Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions.

jupyter analysis Last Checkpoint: 2 hours ago (autosaved)

```
In [3]: # Main analysis
import os
import sys
module_path = os.path.abspath(os.path.join('..'))
if module_path not in sys.path:
    sys.path.append(module_path)
from helpers import database_helper
import pandas as pd
from sqlalchemy import create_engine, text
import matplotlib.pyplot as plt

# Getting the data setup for use
database_helper.load_data()
df_passengers_mishandled = pd.read_sql_table('airline_passenger_mishandles', database_helper.db).dropna().set_index('Id')
df_items_mishandled = pd.read_sql_table('airline_items_mishandles', database_helper.db).dropna().set_index('Id')
df_quarterly_nums = pd.read_sql_table('airline_quarterly_numbers', database_helper.db).dropna().set_index('Id')
df_fares = pd.read_sql_table('connecting_market_fare_info', database_helper.db).set_index('Id')
df_fares_multi = pd.read_sql_table('connecting_market_fare_info_MultiCarrier', database_helper.db).set_index('Id')
```

```
In [4]: # Tells us about the mishandled passengers
df_passengers_mishandled = df_passengers_mishandled.drop_duplicates()
df_passengers_mishandled['PerDenied'] = df_passengers_mishandled['TotalDenied']/df_passengers_mishandled['TotalBoarding']*100
df_passengers_mishandled['PerComped'] = df_passengers_mishandled['NumComp']/df_passengers_mishandled['TotalBoarding']*100
df_passengers_mishandled['PerMishandled'] = (df_passengers_mishandled['TotalDenied']+df_passengers_mishandled['NumComp']+df_passe
df_passengers_mishandled['unique_id'] = df_passengers_mishandled['Year'].astype(str) + df_passengers_mishandled['Quarter'].astype
df_passengers_mishandled
```

Out[4]:

	CompPaid	MktAirlineCode	Month	NumComp	NumDowngraded	NumUpgraded	OpAirlineCode	Quarter	TotalBoarding	TotalDenied	Year	PerDenied
Id												
2020-03-Q1-DL-9E	0	DL	3	7550	4020	11686	9E	1	5928498	15706	2020	0.264924
2020-03-Q1-AA-AA	716301	AA	3	13908	895	351	AA	1	39047799	1246	2020	0.003191
2020-03-Q1-AS-AS	64249	AS	3	2010	18	152	AS	1	8621461	170	2020	0.001972