**Technical Report: 3 Fast 3 Furious**

**Topic/DataSources:** State-level Covid data and notable variables all obtained via Kaggle

**Extract, Transform & Load**:

1. We created our first Jupyter Notebook <Obesity.ipynb> and used:
   1. Pandas
   2. Sqlalchemy
   3. numpy
2. We imported the following <csvData.csv> file into our notebook
3. We executed a quick read of this file to determine what columns and data were present
4. Next we pulled some basic statistical details of the data within the file (.describe)
5. We then renamed columns for ‘obesityRate’ and ‘Pop’ to ‘Obesity Rate” and “Population” into a new data frame for a cleaner presentation.
6. We then imported the <raw\_data.csv> file into our notebook
7. Again, we executed a quick read of the file to determine what columns and data were present
8. We then opened PGAdmin and created our Database <project\_db>
9. Within the database we created tables:
   1. First created table: <obesity\_state> with the following columns
      1. ‘State’ – Varchar (Primary Key)
      2. ‘Obesity Rate’ – Decimal
      3. ‘Population’ INT
   2. Second created table: <health\_state> with the following columns
      1. ‘State’ Varchar (Primary Key)
      2. ‘Health Expen’ INT
10. Next in our notebook, we connected to the database where our tables were created
11. We then ran a quick validation to identify the table names for the newly created tables above
12. We had to reformat our ‘Health Expen’ field to remove the “$” character from the field values
13. We then appended the newly formatted values to the <health\_state> SQL table
14. We executed a quick read validation of that field to confirm the formatting was completed correctly
15. Back in our PGAgmin, we then joined the <obesity\_state> and <health\_state> tables on the ‘State’ key from each table
16. We then created a combined table <combined\_df> of the first 2 tables with the following field/attributes:
    1. ‘State’ – Varchar (Primary Key)
    2. ‘Obesity Rate’ – Decimal
    3. ‘Population’ – INT
    4. ‘Health Expen’ – INT
17. We exported the data from the <combined\_df> table to the <State,ObesityRate, Population,Health Expenryan.csv> file
18. Next we created the following table:
    1. <final\_state> with the following columns:
       1. ‘State’ – Varchar (Primary Key)
       2. ‘Tested’ – Varchar
       3. ‘Infected’ – INT
       4. ‘Deaths’ – INT
       5. ‘ICU Beds’ – INT
       6. ‘Age 0-25’ – Decimal
       7. ‘Age 26-54’ – Decimal
       8. ‘Age 55+’ – Decimal
    2. We then joined the <final\_state> and <combined\_df> tables on the ‘State’ key from each table
19. We then created a second notebook <adding\_covid.ipynb> and used:
    1. Pandas
    2. Sqlalchemy
    3. Numpy
20. We imported the following <covid.csv> file into our notebook
21. We executed a quick read of this file to determine what columns and data were present
22. Next we performed a validation to identify all of the Unique values for the ‘State’ field
23. We executed a count of the unique values in the State field to determine if there were 50 states as expected
24. We noticed an extra row for the District of Columbia and we executed a drop command to remove this value (we noted this as Row 7)
25. We ran another read to confirm we were left with 50 rows for the 50 expected States
26. We then recreated a new data frame only pulling in the columns we were interested in:
    1. State
    2. Tested
    3. Infected
    4. Deaths
    5. ICU Beds
    6. Age 0-25
    7. Age 26-54
    8. Age 55+
27. We then imported the <State,ObesityRate,Population,Health Expenryan.csv> file
28. Next we again connected to the SQL <project\_db> database where our tables were created
29. We ran a quick validation to determine what tables existed within the db
30. We appended the values from the newly formatted data frame to the <combined\_df> table
31. We then ran an export from the <final\_state> table of all the data into the <Ryan’s Table.csv> file.
32. Next we started a new notebook <Alex\_work.ipynb> using:
    1. Pandas
    2. Sqlalchemy
33. We imported the following <co-est2019-alldata.csv> file into our notebook
34. We executed a quick read of this file to determine what columns and data were present
35. Next we performed a rename of columns to better align with our date:
    1. STATE > ID
    2. STNAME > State
    3. CTYNAME > City
    4. PPESTIMATE2019 > 2019\_population
36. We created a new data frame only containing the above 4 columns and did a quick read call to validate that the new data frame contained what was intended
37. We then imported the <Healthcare\_per\_+capita\_State-2014.csv> file to our notebook and ran another quick read to confirm what data was present within the file
38. We renamed the columns to better align and dropped the Unnamed column from the data frame:
    1. Location > State
    2. Health Spending per Capita > $$$
39. We reformatted the $$$ column to remove any ‘$’ characters within the sting to have it saved as an INT
40. Next we imported the <us\_state\_vaccinations.csv> file also performing a quick read to validate what data was present
41. We renamed the ‘location’ column to ‘State’
42. We performed some aggregation functions to sum the following column data by State and performed a group-by to present each State as it’s own row:
    1. Total\_vaccinations
    2. Total\_distributed
    3. People\_accinated
    4. People\_fully\_vaccinated\_per\_hundred
    5. Distributed\_per\_hundred
    6. Daily\_vaccinations\_raw
    7. Daily\_vaccinations
    8. Daily\_vaccinations\_per\_million
    9. Share\_doses\_used
43. We did a quick re-display of the data to review what information we had
44. We then did some reformatting of the following columns to make them ‘int64’
    1. Total\_vaccinations
    2. Total\_distrbuted
    3. People\_vaccinated
45. We removed all dates aside from the recent date of ‘2021-06-10 as the data was cumulative and not a day by day calculation so the max date was the only date necessary to obtain the latest figures
46. We created a data frame that isolated the territories and areas that are not one of the 50 US States and dropped them from the data frame
47. We saved the latest data frame to the <state\_vaccinations.csv> table
48. Next we imported the following files <Ryan’s Table.csv> and <state\_vaccinations.csv> to our notebook
49. We did a quick read of the data in the Ryan file and the newly created State\_vaccinations file and create data frames for reference
50. We created a column in the 2nd reference file for ‘partial\_vaccinations’ which was the value of the ‘people\_vaccinated column – the ‘people\_fully\_vaccinated’ column
51. We dropped all columns from the 2nd data frame aside from the following and updated the names where mentioned:
    1. State
    2. People\_fully\_vaccinated
    3. Percent\_doses\_used (formerly share\_doses\_used)
    4. Partial\_vaccinations
52. We also reformated the percent\_doses\_used columns to be displayed as a % (xx.x%) and ran a quick read validation to ensure the formatting and column changes matched what we intended
53. We then connected to PGAdmin
54. In PGAdmin we created 2 tables <ryan> and <vax> where the corresponding data frames were appended to the table (df\_ryan to ryan and df\_vax to vax)
55. We did a quick validation to ensure the columns and data matched what was intended
56. We then created a table that merged the data from the <vax> and <ryan> tables into a single table names <alex> on the ‘State’ field
57. We exported a copy of the alex table to the <alex.csv> file as a final output
58. Our final production database containing our tables above are all considered a relational database – our master table is the <alex> table which is a combination of all tables created above.