**Turkey Vulture Migration Project Report**

**Team members: Winnie Wu, Lei Qin, and Rob Seaberg**

**Project Purpose:** To demonstrate ETL process.

**Work Flow:**

1. **Extract:**

For this project we chose to focus on a theme relating to environmental studies. From data.world we found data from a study which tracked the migration patterns of Turkey Vultures in North and South America ([here](https://data.world/makeovermonday/2018-w-4-turkey-vulture-migration-in-north-and-south-america))(migration path dataset). We chose this set because we saw the potential to link the study results to other sources such as historical climate and population data for what may be interesting insights.

This original data set is an excel file with 220,000 point-in-time location recordings for 19 unique turkey vultures who were tracked by satellite as they migrated between North and South America during 2003 to 2012. After some research, we found more data sets from this same study on movebank.com, a publicly available repository of academically verified animal tracking datasets. Here also were detailed definitions of each data attribute used in our study data. Interesting to us was an additional excel file ([here](https://www.datarepository.movebank.org/discover?query=Cathartes+aura&filtertype=*&filter=&submit_search-filter-controls_add=Add&rpp=20&sort_by=score&order=DESC&location=l2)) which gave detailed information for each vulture tracked, such as: name, life stage, mass, and study site (vulture information dataset). We also used an excel file Vultures Acopian Center USA which listed additional comments, subspecies, and additional birds to the previous file and found from the same source.

Additional sources include the google maps api for converting geographic coordinates into city and state names as well as the openweathermap api to obtain temperature data for each location, as detailed below.

1. **Transform**:
2. Migration path dataset:

Using jupyter notebook, we imported our migratory point-in-time location recordings. Originally, we converted the excel sheet to a csv file, but after a long importing wait, we found that it was faster to import the large file in the original excel form. We then imported the information to a pandas data frame and dropped irrelevant and repetitive columns. We next completed a series of validation checks such as entry counts per column and dropping incomplete entries. However, this data was pretty clean already and no changes were necessary. We also added a column for taxonomical name and filled in all values as "Cathartes aura," since we knew from our second data set that all birds in this study were of this genus and species. The reasoning was that if we wish to compare this data to migratory patterns of other species, then this column will be useful. We also filtered to ensure we only keep turkey vulture data. The primary key for our original data set was changed to “event\_id” because this value indicates a unique recording for each entry. We then dropped entries that had a duplicate event\_id. Lastly, we checked whether animal\_id or tag\_id was unique for turkey vultures, to determine whicth to use as primary key for the other table.

**II.** Vulture information dataset:

For our next data set, we imported it as a csv into a pandas data frame and followed a similar process, such as dropping unnecessary columns and validation. We noted that unfortunately not all birds had a mass listed, but we did not make any changes to this column. Next a connection was set up to a sqlite database. And the primary key for Vulture information dataset is animal\_id.

**III.** City and temperature dataset using API:

Next, we extracted the latitude and longitude points from our migration data set. First, using citypy, we obtained a list of all the unique cities which the birds migrated over. This information can be found in “vulture\_etl\_cities.ipynb”. However, because citypy only gave us the city name, we tried the Google Maps API next. Google was able to give us city and state names which correlated to all our geographic coordinates. From here, we pulled temperature data from the OpenWeatherMap API for each city in our set.

However, because the format of the JSON data from the Google Maps API could differ from city to city and country to country, there was no clear index for “city”, “country”, etc. Therefore, we used the “compound\_code” under “plus\_code”, which was the first key-value pair that could be identified. While this gave a full address, it also did not always provide the exact format we were looking for. To elaborate, in order to add to a dataframe, the compound\_code was split by comma and each section was added in separate dataframe columns. If a compound\_code did not start with a city name, then a non-city name would have been added to the City column. This meant that we probably were unable to extract the max temp for that entry. A second call-out is that the OpenWeatherMap API could have given information for a city with the same name but in the wrong state or country. Finally, we wanted to note that the city and temp information is only provided for a subset of the data; we took every 500th entry in the dataset and ran the Google Maps and OpenWeatherMap APIs for 200 data points.

1. **Load**:

We committed the above tables to our sqlite database. We chose SQL for its ease of use and relational quality. Using sqlite allowed us to easily work on the same database. Our final tables include “vulture\_detail” which lists details about each bird in the study, “migration\_paths” which lists the point-in-time location recordings, and our “city\_df” which lists temperatures by city in the migration paths of the vultures.

**Additional Movebank Information**:

1. How to use Movebank to download data needed: [here](https://www.hawkmountain.org/science/learn-to-use-tracking-maps/page.aspx?id=4515)
2. Additional detail on the vulture study: [here](https://www.makeovermonday.co.uk/week4-2018/)
3. An interactive map of other animal tracking studies: [here](https://www.movebank.org/panel_embedded_movebank_webapp)