DavidGrice_Final_InsertData

December 18, 2019

0.1 Libraries used for cleaning and storing data

[ ]:
```
import sqlite3
import pandas as pd
import numpy as np
import os
```

0.2 Connecting to database

[ ]:
```
path = './AliensExist.db'
conn = sqlite3.connect(path)
curs = conn.cursor()
```

0.3 Functions used for cleaning and storing data

[ ]:
```
def trim_all_columns(df):
    """
    Trim whitespace from ends of each value across all series in dataframe
    """
    trim_strings = lambda x: x.strip() if isinstance(x, str) else x
    return df.applymap(trim_strings)
```

0.4 UFO data cleaning

[ ]:
```
# Import data for the UFO sightings
ufo = pd.read_csv('./FinalFiles/ufo.csv', dtype={"longitude ":"str"})
ufo.head()
```

[ ]:
```
# Rearranging the columns by date/time, geography, duration, comments
ufo_revised = ufo[['date␣
 ↪posted','datetime','shape','city','state','country','latitude','longitude␣
 ↪','duration (seconds)','duration (hours/min)','comments']]
```

```
# Split the original date/time into originial date and original time
# Dropping the column split from, and reordering the columns
# into the categories of date/time, geography, duration, comments
ufo_revised[['Original_Date', 'Original_Time']] = ufo_revised['datetime'].str.
 ↪split(' ', expand=True)
ufo_revised = ufo_revised.drop('datetime',1)
ufo_revised = ufo_revised[['Original_Date','Original_Time','date␣
 ↪posted','shape', 'city', 'state', 'country', 'latitude', 'longitude ',␣
 ↪'duration (seconds)','duration (hours/min)','comments']]

# Split the posted dates into month, day, year, and dropping the split column
# then reordering the columns into the categories listed above.
ufo_revised[['Posted_Month', 'Posted_Day', 'Posted_Year']] = ufo_revised['date␣
 ↪posted'].str.split('/', expand=True)
ufo_revised = ufo_revised.drop('date posted',1)
ufo_revised =␣
 ↪ufo_revised[['Original_Date','Original_Time','Posted_Month','Posted_Day','Posted_Year','sha
 ↪','duration (seconds)','duration (hours/min)','comments']]

# Splitting the original month, day, year and dropping the column split from.
# Then reordering the columns into categories listed above, and finally triming␣
 ↪whitespace
# from the dataframe.
ufo_revised[['Month', 'Day', 'Year']] = ufo_revised['Original_Date'].str.
 ↪split('/', expand=True)
ufo_revised = ufo_revised.drop('Original_Date',1)
ufo_revised =␣
 ↪ufo_revised[['Month','Day','Year','Original_Time','Posted_Month','Posted_Day','Posted_Year'
 ↪','duration (seconds)','duration (hours/min)','comments']]

ufo_revised_trimmed = ufo_revised.apply(lambda x: x.str.strip() if x.dtype ==␣
 ↪"object" else x)
ufo_revised_trimmed = ufo_revised_trimmed.set_index('country')
ufo_revised_trimmed = ufo_revised_trimmed.drop(['gb', 'ca', 'au', 'de'], axis=0)
ufo_revised_trimmed = ufo_revised_trimmed.reset_index()
ufo_revised_trimmed =␣
 ↪ufo_revised_trimmed[['Month','Day','Year','Original_Time','Posted_Month','Posted_Day','Post
 ↪','duration (seconds)','duration (hours/min)','comments']]
ufo_revised_trimmed = ufo_revised_trimmed.fillna(0)

ufo_revised_trimmed['city'] = [str(i).upper() for i in␣
 ↪ufo_revised_trimmed['city']]
ufo_revised_trimmed['state'] = [str(i).upper() for i in␣
 ↪ufo_revised_trimmed['state']]
ufo_revised_trimmed['shape'] = [str(i).upper() for i in␣
 ↪ufo_revised_trimmed['shape']]
```

```
ufo_revised_trimmed['country'] = [str(i).upper() for i in␣
 ↪ufo_revised_trimmed['country']]
ufo_revised_trimmed['comments'] = [str(i).upper() for i in␣
 ↪ufo_revised_trimmed['comments']]

ufo_revised_trimmed
```

0.5 Inserting UFO Sighting data into the database

[ ]:
```
ufo_revised_latitude = ufo_revised_trimmed[['latitude']].drop_duplicates()
ufo_revised_latitude["ID"] = range(0, 0+len(ufo_revised_latitude))
ufo_revised_latitude = ufo_revised_latitude[['ID','latitude']]
for x in ufo_revised_latitude.values:
    curs.execute(""" INSERT INTO tLatitude VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_longitude = ufo_revised_trimmed[['longitude ']].drop_duplicates()
ufo_revised_longitude["ID"] = range(0, 0+len(ufo_revised_longitude))
ufo_revised_longitude = ufo_revised_longitude[['ID','longitude ']]
for x in ufo_revised_longitude.values:
    curs.execute(""" INSERT INTO tLongitude VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_duration = ufo_revised_trimmed[['duration (seconds)', 'duration␣
 ↪(hours/min)']].drop_duplicates()
ufo_revised_duration["ID"] = range(0, 0+len(ufo_revised_duration))
ufo_revised_duration = ufo_revised_duration[['ID','duration (seconds)',␣
 ↪'duration (hours/min)']]
for x in ufo_revised_duration.values:
    curs.execute(""" INSERT INTO tDuration VALUES(
                        ?,?,?
                        ); """, (x[0],x[1],x[2]) )

ufo_revised_comments = ufo_revised_trimmed[['comments']].drop_duplicates()
ufo_revised_comments["ID"] = range(0, 0+len(ufo_revised_comments))
ufo_revised_comments = ufo_revised_comments[['ID','comments']]
for x in ufo_revised_comments.values:
    curs.execute(""" INSERT INTO tUFOCOM VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_shape = ufo_revised_trimmed[['shape']].drop_duplicates()
ufo_revised_shape["ID"] = range(0, 0+len(ufo_revised_shape))
```

```
ufo_revised_shape = ufo_revised_shape[['ID','shape']]
for x in ufo_revised_shape.values:
    curs.execute(""" INSERT INTO tShape VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_basicTable = ufo_revised_trimmed[['Original_Time','city','comments',
                                        'shape','duration (seconds)','duration␣
 ↪(hours/min)']]
ufo_revised_basicTable["ID"] = range(0, 0+len(ufo_revised_basicTable))
ufo_revised_basicTable["COUNT"] = 1
ufo_revised_basicTable =␣
 ↪ufo_revised_basicTable[['ID','Original_Time','city','comments',
                                        'shape','duration (seconds)','duration␣
 ↪(hours/min)','COUNT']]
for x in ufo_revised_basicTable.values:
    curs.execute(""" INSERT INTO tUFOSightingInfo VALUES(
                        ?,?,?,?,?,?,?,?
                        ); """, (x[0],x[1],x[2],x[3],x[4],x[5],x[6],x[7]) )
conn.commit()
```

0.6 Inserting UFO Original Time into database

[ ]:
```
ufo_revised_originalTime = ufo_revised_trimmed[['Original_Time']].
 ↪drop_duplicates()
ufo_revised_originalTime["ID"] = range(0, 0+len(ufo_revised_originalTime))
ufo_revised_originalTime = ufo_revised_originalTime[['ID','Original_Time']]
for x in ufo_revised_originalTime.values:
    curs.execute(""" INSERT INTO tTime VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_originalMonth = ufo_revised_trimmed[['Month']].drop_duplicates()
ufo_revised_originalMonth["ID"] = range(0, 0+len(ufo_revised_originalMonth))
ufo_revised_originalMonth = ufo_revised_originalMonth[['ID','Month']]
for x in ufo_revised_originalMonth.values:
    curs.execute(""" INSERT INTO tOMonth VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_originalDay = ufo_revised_trimmed[['Day']].drop_duplicates()
ufo_revised_originalDay["ID"] = range(0, 0+len(ufo_revised_originalDay))
ufo_revised_originalDay = ufo_revised_originalDay[['ID','Day']]
for x in ufo_revised_originalDay.values:
    curs.execute(""" INSERT INTO tODay VALUES(
```

```
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_originalYear = ufo_revised_trimmed[['Year']].drop_duplicates()
ufo_revised_originalYear["ID"] = range(0, 0+len(ufo_revised_originalYear))
ufo_revised_originalYear = ufo_revised_originalYear[['ID','Year']]
for x in ufo_revised_originalYear.values:
    curs.execute(""" INSERT INTO tOYear VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_originalTimeDay =␣
 ↪ufo_revised_trimmed[['Original_Time','Month','Day','Year']]
ufo_revised_originalTimeDay["ID"] = range(0, 0+len(ufo_revised_originalTimeDay))
ufo_revised_originalTimeDay =␣
 ↪ufo_revised_originalTimeDay[['ID','Original_Time','Month','Day','Year']]
for x in ufo_revised_originalTimeDay.values:
    curs.execute(""" INSERT INTO tUFOOriginalDate VALUES(
                        ?,?,?,?,?
                        ); """, (x[0],x[1],x[2],x[3],x[4]) )
conn.commit()
```

0.7 Inserting UFO Posted Time into database

[ ]:
```
ufo_revised_postedMonth = ufo_revised_trimmed[['Posted_Month']].
 ↪drop_duplicates()
ufo_revised_postedMonth["ID"] = range(0, 0+len(ufo_revised_postedMonth))
ufo_revised_postedMonth = ufo_revised_postedMonth[['ID','Posted_Month']]
for x in ufo_revised_postedMonth.values:
    curs.execute(""" INSERT INTO tPMonth VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_postedDay = ufo_revised_trimmed[['Posted_Day']].drop_duplicates()
ufo_revised_postedDay["ID"] = range(0, 0+len(ufo_revised_postedDay))
ufo_revised_postedDay = ufo_revised_postedDay[['ID','Posted_Day']]
for x in ufo_revised_postedDay.values:
    curs.execute(""" INSERT INTO tPDay VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_postedYear = ufo_revised_trimmed[['Posted_Year']].drop_duplicates()
ufo_revised_postedYear["ID"] = range(0, 0+len(ufo_revised_postedYear))
ufo_revised_postedYear = ufo_revised_postedYear[['ID','Posted_Year']]
for x in ufo_revised_postedYear.values:
```

```
    curs.execute(""" INSERT INTO tPYear VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_postedTimeDay =␣
 ↪ufo_revised_trimmed[['Posted_Month','Posted_Day','Posted_Year']]
ufo_revised_postedTimeDay["ID"] = range(0, 0+len(ufo_revised_postedTimeDay))
ufo_revised_postedTimeDay =␣
 ↪ufo_revised_postedTimeDay[['ID','Posted_Month','Posted_Day','Posted_Year']]
for x in ufo_revised_postedTimeDay.values:
    curs.execute(""" INSERT INTO tUFOPostedDate VALUES(
                        ?,?,?,?
                        ); """, (x[0],x[1],x[2],x[3]) )
conn.commit()
```

0.8 Inserting UFO City/State/Country data into database

[ ]:
```
ufo_revised_city = ufo_revised_trimmed[['city']].drop_duplicates()
ufo_revised_city["ID"] = range(0, 0+len(ufo_revised_city))
ufo_revised_city = ufo_revised_city[['ID','city']]
for x in ufo_revised_city.values:
    curs.execute(""" INSERT INTO tCity VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_cityInfo = ufo_revised_trimmed[['city', 'latitude', 'longitude ']]
ufo_revised_cityInfo["ID"] = range(0, 0+len(ufo_revised_cityInfo))
ufo_revised_cityInfo = ufo_revised_cityInfo[['ID','city', 'latitude',␣
 ↪'longitude ']]
for x in ufo_revised_cityInfo.values:
    curs.execute(""" INSERT INTO tUFOCityInfo VALUES(
                        ?,?,?,?
                        ); """, (x[0],x[1],x[2],x[3]) )

ufo_revised_state = ufo_revised_trimmed[['state']].drop_duplicates()
ufo_revised_state["ID"] = range(0, 0+len(ufo_revised_state))
ufo_revised_state = ufo_revised_state[['ID','state']]
for x in ufo_revised_state.values:
    curs.execute(""" INSERT INTO tState VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

ufo_revised_cityState = ufo_revised_trimmed[['city','state','country']]
ufo_revised_cityState["ID"] = range(0, 0+len(ufo_revised_cityState))
ufo_revised_cityState = ufo_revised_cityState[['ID','city','state','country']]
```

```
for x in ufo_revised_cityState.values:
    curs.execute(""" INSERT INTO tUFOCityStateCountry VALUES(
                        ?,?,?,?
                        ); """, (x[0],x[1],x[2],x[3]) )

ufo_revised_country = ufo_revised_trimmed[['country']].drop_duplicates()
ufo_revised_country["ID"] = range(0, 0+len(ufo_revised_country))
ufo_revised_country = ufo_revised_country[['ID','country']]
for x in ufo_revised_country.values:
    curs.execute(""" INSERT INTO tCountry VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )
conn.commit()
```

0.9 Flight data cleaning

[ ]:
```
# Import data for the flights.
flight = pd.read_csv("./FinalFiles/ifo.csv", dtype={"Dest_airport_lat":"str",
                                                    "Org_airport_lat":"str",
                                                    "Org_airport_long":"str",
                                                    "Dest_airport_long":"str",
                                                    "Passengers":"str",
                                                    "Seats":"str",
                                                    "Flights":"str",
                                                    "Distance":"str",
                                                    "Origin_population":"str",
                                                    "Destination_population":
 ↪"str"})
flight.head()
```

[ ]:
```
# Reorder the columns by flight, distance, and geogaphy.
flight_revised = flight[['Passengers', 'Seats', 'Flights',␣
 ↪'Distance','Fly_date',
                         'Origin_airport', 'Destination_airport',␣
 ↪'Origin_city','Destination_city',
                        'Origin_population', 'Destination_population',␣
 ↪'Org_airport_lat',
                         'Org_airport_long',␣
 ↪'Dest_airport_lat','Dest_airport_long']]

# Split the flight days, drop the columns, and reorder the columns.
flight_revised[['Year', 'Month', 'Day']] = flight_revised['Fly_date'].str.
 ↪split('-', expand=True)
flight_revised = flight_revised.drop('Fly_date',1)
flight_revised['Day'] = flight_revised['Day'].map(lambda x: x.lstrip('0'))
```

```
flight_revised['Month'] = flight_revised['Month'].map(lambda x: x.lstrip('0'))
flight_revised = flight_revised[['Passengers', 'Seats', 'Flights',␣
↪'Distance','Year', 'Month','Day',
                         'Origin_airport', 'Destination_airport',␣
↪'Origin_city','Destination_city',
                        'Origin_population', 'Destination_population',␣
↪'Org_airport_lat',
                         'Org_airport_long',␣
↪'Dest_airport_lat','Dest_airport_long']]

# Splif the origin and destination city/state information, drop the columns,␣
↪reorder the columns
flight_revised[['Origin_City', 'Origin_State']] = flight_revised['Origin_city'].
↪str.split(', ', expand=True)
flight_revised = flight_revised.drop('Origin_city',1)
flight_revised[['Destination_City', 'Destination_State']] =␣
↪flight_revised['Destination_city'].str.split(', ', expand=True)
flight_revised = flight_revised.drop('Destination_city',1)
flight_revised = flight_revised[['Passengers', 'Seats', 'Flights',␣
↪'Distance','Year', 'Month', 'Day',
                         'Origin_airport', 'Destination_airport',␣
↪'Origin_City', 'Origin_State','Destination_City', 'Destination_State',
                        'Origin_population', 'Destination_population',␣
↪'Org_airport_lat',
                         'Org_airport_long',␣
↪'Dest_airport_lat','Dest_airport_long']]

flight_revised['Origin_City'] = [str(i).upper() for i in␣
↪flight_revised['Origin_City']]
flight_revised['Origin_State'] = [str(i).upper() for i in␣
↪flight_revised['Origin_State']]
flight_revised['Destination_City'] = [str(i).upper() for i in␣
↪flight_revised['Destination_City']]
flight_revised['Destination_State'] = [str(i).upper() for i in␣
↪flight_revised['Destination_State']]
flight_revised['Origin_airport'] = [str(i).upper() for i in␣
↪flight_revised['Origin_airport']]
flight_revised['Destination_airport'] = [str(i).upper() for i in␣
↪flight_revised['Destination_airport']]
flight_revised = flight_revised.fillna(0)

# Trim the white space from the columns and view final data.
flight_revised_trimmed = flight_revised.apply(lambda x: x.str.strip() if x.
↪dtype == "object" else x)

flight_revised_trimmed
```

0.10 Inserting Flight Plan info into database

[ ]:
```
flight_revised_passengers = flight_revised_trimmed[['Passengers']].
 ↪drop_duplicates()
flight_revised_passengers["ID"] = range(0, 0+len(flight_revised_passengers))
flight_revised_passengers = flight_revised_passengers[['ID','Passengers']]
for x in flight_revised_passengers.values:
    curs.execute(""" INSERT INTO tFlightPassengers VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_seats = flight_revised_trimmed[['Seats']].drop_duplicates()
flight_revised_seats["ID"] = range(0, 0+len(flight_revised_seats))
flight_revised_seats = flight_revised_seats[['ID','Seats']]
for x in flight_revised_seats.values:
    curs.execute(""" INSERT INTO tFlightSeats VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_flights = flight_revised_trimmed[['Flights']].drop_duplicates()
flight_revised_flights["ID"] = range(0, 0+len(flight_revised_flights))
flight_revised_flights = flight_revised_flights[['ID','Flights']]
for x in flight_revised_flights.values:
    curs.execute(""" INSERT INTO tFlightFlights VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_distance = flight_revised_trimmed[['Distance']].drop_duplicates()
flight_revised_distance["ID"] = range(0, 0+len(flight_revised_distance))
flight_revised_distance = flight_revised_distance[['ID','Distance']]
for x in flight_revised_distance.values:
    curs.execute(""" INSERT INTO tFlightDistance VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_planeInfo =␣
 ↪flight_revised_trimmed[['Origin_airport','Month','Day','Year','Destination_airport','Flight
flight_revised_planeInfo["ID"] = range(0, 0+len(flight_revised_planeInfo))
flight_revised_planeInfo = flight_revised_planeInfo[['ID','Origin_airport',
                                                     'Month','Day','Year',
                                                    ␣
 ↪'Destination_airport','Flights']]
for x in flight_revised_planeInfo.values:
    curs.execute(""" INSERT INTO tFlightPlaneInfo VALUES(
                        ?,?,?,?,?,?,?
                        ); """, (x[0],x[1],x[2],x[3],x[4],x[5],x[6]) )
conn.commit()
```

0.11 Inserting Flight Date info into database

[ ]:
```
flight_revised_month = flight_revised_trimmed[['Month']].drop_duplicates()
flight_revised_month["ID"] = range(0, 0+len(flight_revised_month))
flight_revised_month = flight_revised_month[['ID','Month']]
for x in flight_revised_month.values:
    curs.execute(""" INSERT INTO tFlightMonth VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_day = flight_revised_trimmed[['Day']].drop_duplicates()
flight_revised_day["ID"] = range(0, 0+len(flight_revised_day))
flight_revised_day = flight_revised_day[['ID','Day']]
for x in flight_revised_day.values:
    curs.execute(""" INSERT INTO tFlightDay VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_year = flight_revised_trimmed[['Year']].drop_duplicates()
flight_revised_year["ID"] = range(0, 0+len(flight_revised_year))
flight_revised_year = flight_revised_year[['ID','Year']]
for x in flight_revised_year.values:
    curs.execute(""" INSERT INTO tFlightYear VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_date = flight_revised_trimmed[['Year', 'Month', 'Day']]
flight_revised_date["ID"] = range(0, 0+len(flight_revised_date))
flight_revised_date = flight_revised_date[['ID','Year', 'Month', 'Day']]
for x in flight_revised_date.values:
    curs.execute(""" INSERT INTO tFlightDate VALUES(
                        ?,?,?,?
                        ); """, (x[0],x[1],x[2],x[3]) )
conn.commit()
```

0.12 Inserting Origin Airport/City/State info into database

[ ]:
```
flight_revised_OrigPop = flight_revised_trimmed[['Origin_population']].
↪drop_duplicates()
flight_revised_OrigPop["ID"] = range(0, 0+len(flight_revised_OrigPop))
flight_revised_OrigPop = flight_revised_OrigPop[['ID','Origin_population']]
for x in flight_revised_OrigPop.values:
    curs.execute(""" INSERT INTO tFlightOrigPopulation VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )
```

```
flight_revised_OrigCity = flight_revised_trimmed[['Origin_City']].
 ↪drop_duplicates()
flight_revised_OrigCity["ID"] = range(0, 0+len(flight_revised_OrigCity))
flight_revised_OrigCity = flight_revised_OrigCity[['ID','Origin_City']]
for x in flight_revised_OrigCity.values:
    curs.execute(""" INSERT INTO tFlightOrigCity VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_OrigState = flight_revised_trimmed[['Origin_State']].
 ↪drop_duplicates()
flight_revised_OrigState["ID"] = range(0, 0+len(flight_revised_OrigState))
flight_revised_OrigState = flight_revised_OrigState[['ID','Origin_State']]
for x in flight_revised_OrigState.values:
    curs.execute(""" INSERT INTO tFlightOrigState VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_OrigAirport = flight_revised_trimmed[['Origin_airport']].
 ↪drop_duplicates()
flight_revised_OrigAirport["ID"] = range(0, 0+len(flight_revised_OrigAirport))
flight_revised_OrigAirport = flight_revised_OrigAirport[['ID','Origin_airport']]
for x in flight_revised_OrigAirport.values:
    curs.execute(""" INSERT INTO tFlightOrigAirport VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_OrigAirportLat = flight_revised_trimmed[['Org_airport_lat']].
 ↪drop_duplicates()
flight_revised_OrigAirportLat["ID"] = range(0,␣
 ↪0+len(flight_revised_OrigAirportLat))
flight_revised_OrigAirportLat =␣
 ↪flight_revised_OrigAirportLat[['ID','Org_airport_lat']]
for x in flight_revised_OrigAirportLat.values:
    curs.execute(""" INSERT INTO tFlightOrigAirportLat VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_OrigAirportLong = flight_revised_trimmed[['Org_airport_long']].
 ↪drop_duplicates()
flight_revised_OrigAirportLong["ID"] = range(0,␣
 ↪0+len(flight_revised_OrigAirportLong))
flight_revised_OrigAirportLong =␣
 ↪flight_revised_OrigAirportLong[['ID','Org_airport_long']]
for x in flight_revised_OrigAirportLong.values:
```

```
    curs.execute(""" INSERT INTO tFlightOrigAirportLong VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_OrigCityInfo =␣
 ↪flight_revised_trimmed[['Origin_City','Origin_State',
                                                  ␣
 ↪'Origin_airport','Org_airport_lat','Org_airport_long']].drop_duplicates()
flight_revised_OrigCityInfo["ID"] = range(0, 0+len(flight_revised_OrigCityInfo))
flight_revised_OrigCityInfo = flight_revised_OrigCityInfo[['ID',
                                                           ␣
 ↪'Origin_City','Origin_State',
                                                           ␣
 ↪'Origin_airport','Org_airport_lat',
                                                            'Org_airport_long']]
for x in flight_revised_OrigCityInfo.values:
    curs.execute(""" INSERT INTO tFlightOrigAirportInfo VALUES(
                        ?,?,?,?,?,?
                        ); """, (x[0],x[1],x[2],x[3],x[4],x[5]) )
conn.commit()
```

0.13 Inserting Destination Airport/City/State info into database

[ ]:
```
flight_revised_DestPop = flight_revised_trimmed[['Destination_population']].
 ↪drop_duplicates()
flight_revised_DestPop["ID"] = range(0, 0+len(flight_revised_DestPop))
flight_revised_DestPop = flight_revised_DestPop[['ID','Destination_population']]
for x in flight_revised_DestPop.values:
    curs.execute(""" INSERT INTO tFlightDestPopulation VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_DestCity = flight_revised_trimmed[['Destination_City']].
 ↪drop_duplicates()
flight_revised_DestCity["ID"] = range(0, 0+len(flight_revised_DestCity))
flight_revised_DestCity = flight_revised_DestCity[['ID','Destination_City']]
for x in flight_revised_DestCity.values:
    curs.execute(""" INSERT INTO tFlightDestCity VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_DestState = flight_revised_trimmed[['Destination_State']].
 ↪drop_duplicates()
flight_revised_DestState["ID"] = range(0, 0+len(flight_revised_DestState))
flight_revised_DestState = flight_revised_DestState[['ID','Destination_State']]
```

```
for x in flight_revised_DestState.values:
    curs.execute(""" INSERT INTO tFlightDestState VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_DestAirport = flight_revised_trimmed[['Destination_airport']].
 ↪drop_duplicates()
flight_revised_DestAirport["ID"] = range(0, 0+len(flight_revised_DestAirport))
flight_revised_DestAirport =␣
 ↪flight_revised_DestAirport[['ID','Destination_airport']]
for x in flight_revised_DestAirport.values:
    curs.execute(""" INSERT INTO tFlightDestAirport VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_DestAirportLat = flight_revised_trimmed[['Dest_airport_lat']].
 ↪drop_duplicates()
flight_revised_DestAirportLat["ID"] = range(0,␣
 ↪0+len(flight_revised_DestAirportLat))
flight_revised_DestAirportLat =␣
 ↪flight_revised_DestAirportLat[['ID','Dest_airport_lat']]
for x in flight_revised_DestAirportLat.values:
    curs.execute(""" INSERT INTO tFlightDestAirportLat VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_DestAirportLong = flight_revised_trimmed[['Dest_airport_long']].
 ↪drop_duplicates()
flight_revised_DestAirportLong["ID"] = range(0,␣
 ↪0+len(flight_revised_DestAirportLong))
flight_revised_DestAirportLong =␣
 ↪flight_revised_DestAirportLong[['ID','Dest_airport_long']]
for x in flight_revised_DestAirportLong.values:
    curs.execute(""" INSERT INTO tFlightDestAirportLong VALUES(
                        ?,?
                        ); """, (x[0],x[1]) )

flight_revised_DestCityInfo =␣
 ↪flight_revised_trimmed[['Destination_airport','Destination_City','Destination_State',
                                                    ␣
 ↪'Dest_airport_lat','Dest_airport_long']].drop_duplicates()
flight_revised_DestCityInfo["ID"] = range(0, 0+len(flight_revised_DestCityInfo))
flight_revised_DestCityInfo =␣
 ↪flight_revised_DestCityInfo[['ID','Destination_City',
                                                          ␣
 ↪'Destination_State','Destination_airport',
```

```
                                                                    ␣
 ↪'Dest_airport_lat','Dest_airport_long']]
for x in flight_revised_DestCityInfo.values:
    curs.execute(""" INSERT INTO tFlightDestAirportInfo VALUES(
                        ?,?,?,?,?,?
                        ); """, (x[0],x[1],x[2],x[3],x[4],x[5]) )
conn.commit()
curs.execute("PRAGMA foreign_keys=ON")
conn.close()
```

[ ]: