Turismo asiatico en Colombia: Departamentos de hospedaje reportados trimestral 2018-2019

El presente documento exhibe el código utilizado para realizar el mapa coroplético de los departamentos que los visitantes de origen asiatico reportaron como lugar de su hospedaje principal por lo tanto no refleja el flujo turistico de visitantes de origen asiatico a los departamentos ya que por ejemplo la mayoría reporta Bogotá pero algunos visitan varios departamentos. Este ejercicio tuvo como finalidad aplicar conocimientos en Python y aprender sobre gráficas utilizando mapas.

Los datos son de 2018 y 2019 los cuales fueron descargados de los tableros de control en Tableau proporcionados por Migración Colombia. Los datos se encuentran desagregados por infomración demografica de cada grupo de pasajeros que llegaban a Colombia por consiguiente se agrupó por frecuencia trimestral y por nacionalidad de origen.

Datos para 2018: https://public.tableau.com/profile/migraci.n.colombia#!/vizhome/TablasdeSalidas2018/Inicio

Datos para 2019: https://public.tableau.com/profile/migraci.n.colombia#!/vizhome/FlujosMigratorios-2019/Inicio

In [21]:
```
import pandas as pd
import numpy as np
import re

import urllib3
import requests
import csv
import json
from bs4 import BeautifulSoup
from urllib.request import urlopen

urllib3.disable_warnings()
bd_2018= pd.read_csv('Mapa_Nacionalidad_E_Datos_completos_data.csv', sep=";")
bd_2018.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2660449 entries, 0 to 2660448
Data columns (total 24 columns):
 #   Column                          Dtype 
---  ------                          ----- 
 0   País Nacionalidad               object
 1   Centro Regional                 object
 2   Puesto Migratorio               object
 3   Tipo Transporte                 object
 4   Ciudad Hospedaje                object
 5   Entrada Salida                  object
 6   Meses1                          object
 7   Motivo Viaje                    object
 8   Rango Edad                      object
 9   Colombiano Extranjero           object
 10  Departamento Hospedaje          object
 11  Region Nacionalidad             object
 12  Año                             int64 
 13  Categoría Migratoria            object
 14  Departamento1                   object
 15  Entrada Salida (copia)          object
 16  Ocupación                       object
 17  País Destino Procedencia        object
 18  Region Destino                  object
 19  Sexo1                           object
 20  Cantidad de filas (agregadas)   int64 
 21  Femenino                        int64 
 22  Masculino                       int64 
 23  Número de registros             int64 
dtypes: int64(5), object(19)
memory usage: 487.1+ MB
```

In [22]:
```
bd_2019= pd.read_csv('Meses_E_Datos_completos_data .csv', sep=";")
bd_2019.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1318605 entries, 0 to 1318604
Data columns (total 23 columns):
 #   Column                          Non-Null Count    Dtype 
---  ------                          --------------    ----- 
 0   Meses1                          1318605 non-null  object
 1   Entrada Salida                  1318605 non-null  object
 2   Centro Regional                 1318605 non-null  object
 3   Puesto Migratorio               1318605 non-null  object
 4   Tipo Transporte                 1318605 non-null  object
 5   Ciudad Hospedaje                1318605 non-null  object
 6   Motivo Viaje                    1318605 non-null  object
 7   País Nacionalidad               1318604 non-null  object
 8   Rango Edad                      1318605 non-null  object
 9   Colombiano Extranjero           1318605 non-null  object
 10  Departamento Hospedaje          1318605 non-null  object
 11  Region Nacionalidad             1318604 non-null  object
 12  Año                             1318605 non-null  int64 
 13  Categoría Migratoria            1318605 non-null  object
 14  Departamento1                   1318605 non-null  object
 15  Entrada Salida (copia)          1318605 non-null  object
 16  País Destino Procedencia        1318605 non-null  object
 17  Region Destino                  1318605 non-null  object
 18  Sexo1                           1318605 non-null  object
 19  Cantidad de filas (agregadas)   1318605 non-null  int64 
 20  Femenino                        1318605 non-null  int64 
 21  Masculino                       1318605 non-null  int64 
 22  Número de registros             1318605 non-null  int64 
dtypes: int64(5), object(18)
memory usage: 231.4+ MB
```

In [23]:
```
#Utilizando los datos en formato Json del usuario en github john-guerra el cual hizo un estupendo trabajo recopilando en una base la información para hacer los mapas de los departamentos
with urlopen('https://gist.githubusercontent.com/john-guerra/43c7656821069d00dcbc/raw/be6a6e239cd5b5b803c6e7c2ec405b793a9064dd/Colombia.geo.json') as response:
    departamentos = json.load(response)
```

In [24]:
```
#Función para reemplazar las vocales con tilde y así poder cruzar con la base de los mapas
def normalize(s):
    replacements = (
        ("á", "a"),
        ("é", "e"),
        ("í", "i"),
        ("ó", "o"),
        ("ú", "u"),
    )
    for a, b in replacements:
        s = s.replace(a, b).replace(a.upper(), b.upper())
    return s
```

In [25]:
```
#De la base de 2018 agrupo los datos para extraer el total por departamento, mes, rango de edad y sexo.
bd_2018['Departamento Hospedaje']= bd_2018['Departamento Hospedaje'].str.upper() #Los departamentos en la base json están en mayuscula
bd_2018= bd_2018.loc[(bd_2018['Entrada Salida (copia)']=='Entradas')& (bd_2018['Region Nacionalidad']=='Asia')]
bd_2018['Departamento Hospedaje']= bd_2018['Departamento Hospedaje'].apply(normalize) #aplicamos la función para quitar las tildes
#San andres y bogotá en la base Json aparece de diferente forma
bd_2018['Departamento Hospedaje']= bd_2018.apply(lambda x: "ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA"
                                 if x['Departamento Hospedaje'] == 'SAN ANDRES' else ('SANTAFE DE BOGOTA D.C'
                                                                                      if x['Departamento Hospedaje']=='BOGOTA D.C.'
                                                                                      else x['Departamento Hospedaje']), axis=1)
bd_2018AGR= bd_2018.groupby(['Departamento Hospedaje','Año','Meses1', 'Rango Edad']).aggregate({'Femenino':np.sum,
                                                 'Masculino': np.sum, 'Cantidad de filas (agregadas)':np.sum,}).reset_index()


#Repito el mismo procedimiento para 2019.
bd_2019['Departamento Hospedaje']= bd_2019['Departamento Hospedaje'].str.upper()
bd_2019= bd_2019.loc[(bd_2019['Entrada Salida (copia)']=='Entradas')& (bd_2019['Region Nacionalidad']=='Asia')]
bd_2019['Departamento Hospedaje']= bd_2019['Departamento Hospedaje'].apply(normalize)
bd_2019['Departamento Hospedaje']= bd_2019.apply(lambda x: "ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA"
                                 if x['Departamento Hospedaje'] == 'SAN ANDRES' else ('SANTAFE DE BOGOTA D.C'
                                                                                      if x['Departamento Hospedaje']=='BOGOTA D.C.'
                                                                                      else x['Departamento Hospedaje']), axis=1)
bd_2019AGR= bd_2019.groupby(['Departamento Hospedaje','Año','Meses1', 'Rango Edad']).aggregate({'Femenino':np.sum,
                                                 'Masculino': np.sum, 'Cantidad de filas (agregadas)':np.sum,}).reset_index()
```

In [26]:
```
#Unimos ambas bases para tener una consolidada
bd= pd.concat([bd_2018AGR,bd_2019AGR], axis=0)
bd.rename(columns={'Meses1': 'Mes', 'Cantidad de filas (agregadas)': 'Total'}, inplace=True)
bd
```

Out[26]:

	Departamento Hospedaje	Año	Mes	Rango Edad	Femenino	Masculino	Total
0	AMAZONAS	2018	Abril	18-29	7	15	22
1	AMAZONAS	2018	Abril	30-39	6	3	9
2	AMAZONAS	2018	Abril	40-49	3	2	5
3	AMAZONAS	2018	Abril	60-69	0	1	1
4	AMAZONAS	2018	Agosto	18-29	4	5	9
...	...	...	...	...	...	...	...
1561	VALLE DEL CAUCA	2019	Septiembre	70 o Más	4	7	11
1562	VICHADA	2019	Abril	30-39	1	1	2
1563	VICHADA	2019	Diciembre	40-49	0	1	1
1564	VICHADA	2019	Enero	30-39	0	1	1
1565	VICHADA	2019	Noviembre	70 o Más	1	1	2

3135 rows × 7 columns

In [27]:
```
#Lista con los valores único con el fin de añadir información a los departamentos que no registran
depart = list(bd['Departamento Hospedaje'].unique())
depart.append('VAUPES')
meses = list(bd['Mes'].unique())
edad= list(bd['Rango Edad'].unique())
```

In [28]:
```
#Para algunos departamentos no hay información por lo tanto el mapa de Colombia estará incompleto en algunos trimestres
#Por lo tanto, agrego información de cada departamento por cada rango de edad y sexo pero como en turistas le doy valor de 0
#Así en cada trimestre el mapa de Colombia aparecerá completo pero con departamentos con cero visitantes
for i in depart:
    for k in meses:
        for j in edad:
            for a in range(2018,2020):

                df= pd.DataFrame({'Departamento Hospedaje':[i],
                                  'Año':[a],
                                 'Mes': [k],
                                 'Rango Edad': [j]})
                bd= pd.concat([bd,df], axis=0)
```

In [29]:
```
#creo los trimestres y luego agrupo la abse de datos por esos trimestres creados, departamento y rango de edad.
import locale
locale.setlocale(locale.LC_ALL,'es_ES.UTF-8')
bd['Date']= bd['Mes'].map(str) + '-'+ bd['Año'].map(str)
bd['Date']= pd.to_datetime(bd["Date"], infer_datetime_format=True, format="%B-%Y")
bd['Date']= bd['Date'] + pd.offsets.MonthEnd(0)
bd.sort_values(['Date'], ascending=True, inplace=True)
bd['Qtr'] = pd.PeriodIndex(bd['Date'], freq='Q')
bdf= bd.groupby(['Departamento Hospedaje', 'Qtr', 'Rango Edad']).aggregate({'Femenino':np.sum,'Masculino': np.sum, 'Total':np.sum}).reset_index()

#Para san andres cree santa catalina aparte pero con las mimsas cifras e san andres esto por lo que haré para incluir san adnres en el mapa
bdf_sa= bdf[bdf['Departamento Hospedaje'].isin(['ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA'])].reset_index()
bdf_sa['Departamento Hospedaje'] = bdf_sa['Departamento Hospedaje'].replace('ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA', 'SANTA CATALINA')
bdf_sa.drop(['index'],axis = 'columns', inplace=True)
bdf= pd.concat([bdf,bdf_sa], axis=0)
bdf
```

Out[29]:

	Departamento Hospedaje	Qtr	Rango Edad	Femenino	Masculino	Total
0	AMAZONAS	2018Q1	0-17	2.0	1.0	3.0
1	AMAZONAS	2018Q1	18-29	7.0	24.0	31.0
2	AMAZONAS	2018Q1	30-39	7.0	4.0	11.0
3	AMAZONAS	2018Q1	40-49	5.0	8.0	13.0
4	AMAZONAS	2018Q1	50-59	2.0	5.0	7.0
...	...	...	...	...	...	...
51	SANTA CATALINA	2019Q4	30-39	12.0	15.0	27.0
52	SANTA CATALINA	2019Q4	40-49	2.0	7.0	9.0
53	SANTA CATALINA	2019Q4	50-59	6.0	7.0	13.0
54	SANTA CATALINA	2019Q4	60-69	1.0	1.0	2.0
55	SANTA CATALINA	2019Q4	70 o Más	0.0	0.0	0.0

2016 rows × 6 columns

In [30]:
```
#lista con los trimestres con el fin de hacer el loop con los frames
lista_tr = bdf.sort_values('Qtr', ascending = True)
lista_tr = lista_tr['Qtr'].astype(str)
lista_tr = list(lista_tr.unique())
lista_tr
```

Out[30]:
```
['2018Q1',
 '2018Q2',
 '2018Q3',
 '2018Q4',
 '2019Q1',
 '2019Q2',
 '2019Q3',
 '2019Q4']
```

In [31]:
```
#Esto es un json personal que realicé en la página https://geojson.io/ debido a que en el mapa
#la distancia con San andrés es muy lejana y no me permite reescalarlo
#además su tamaño es muy pequeño entonces tuve que hacerlo manualmente a la vez que reescalando santa catalina
#este mapa de san andrés no corresponde con alguna escala o proyección solo es una realización manual con el fin de que se note en el gráfico
with open('map.geojson') as f:
    SA = json.load(f)
```

Gráfico animado

Por último, ya con las bases de datos procedí a realizar el gráfico coroplético animado por rango de edad. Para esto hice un gráfico con seis subplots tres aparte para San Andrés para poder resaltarlo aparte ya que como mencioné anteriormente en el mapa debido a la distancia y a su tamaño no se nota por lo tanto realicé mi propio geoJson para San Andrés a través de la página https://geojson.io/. La base de datos definitiva quedó en trimestres por lo tanto el total de frames de la animación será de 8.

Elegí agrupar los rangos de edad en tres grupos con el fin de que los mapas puedan apreciarse mejor ya que entre más subplots y mapas es más difícil poder observar las diferencias entre los departamentos.Otra decisión subjetiva fue con respecto a el rango ya que dentro de los departamentos de hospedaje reportados Bogotá, Antioquia, Bolivar tienen cifras de miles de turistas por lo que para apreciar mejor los demás departamentos dentro de la escala de colores establecí como limite máximo 100, es decir aquellos departamentos con más de 105 turistas tendrán color azul oscuro. De esa manera se podrá apreciar mejor la esclaa de colores en los demás departamentos cuyos valores son menores de 105 que son la gran mayoria. Sin embargo, esto no implica que el número de turistas que aquellos departamentos recibieron, vuelvo y recalco la información se basa en los departamentos de hospedaje que los turistas reportan más no ello no implica que los turistas no recorran otros departamentos.

In [32]:
```
#importar las librerias para los gráficos
import plotly.express as px
from plotly.subplots import make_subplots
import plotly.graph_objects as go
#Una grafica con dos subplots basica
```

```
fig = make_subplots(rows=2, cols=4, horizontal_spacing=0, print_grid=True, vertical_spacing=0.001,column_widths=[0.05,0.5,0.05,0.4],
                    specs=[[{'rowspan': 2,'type': 'choropleth'},{'rowspan': 2,'type': 'choropleth'},{'type': 'choropleth'}, {'type': 'choropleth'}],
                           [None, None, {'type': 'choropleth'}, {'type': 'choropleth'}]])
```

```
This is the format of your plot grid:
[ (1,1) geo  ] [ (1,2) geo2 ] [ (1,3) geo3 ]  [ (1,4) geo4 ]
[      :     ] [      :     ] [ (2,3) geo5 ]  [ (2,4) geo6 ]
```

In [33]:

```
#Para el primer frame del gráfico correspondiente al primer trimestre de 2018
#Extraigo la información del primer trimestre de 2018
frame1 = bdf[bdf['Qtr'] == '2018Q1'].reset_index()

#Separo la base por rango de edades y la información de San andres por aparte para su mapa propio
#base para el primer fram del grupo de 30 a 49 años
frame1_3049= frame1[(frame1['Rango Edad']== '30-39')|(frame1['Rango Edad']== '40-49')]
frame1_3049= frame1_3049.groupby(['Departamento Hospedaje']).aggregate({'Femenino':np.sum,'Masculino': np.sum,
                                                                        'Total':np.sum}).reset_index()
sa_3049= frame1_3049[(frame1_3049['Departamento Hospedaje']=='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA')|
                     (frame1_3049['Departamento Hospedaje']=='SANTA CATALINA')]
frame1_3049= frame1_3049[(frame1_3049['Departamento Hospedaje']!='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA') &
                     (frame1_3049['Departamento Hospedaje']!='SANTA CATALINA')]

#base para el primer fram del grupo de 0 a 29 años
frame1_029= frame1[(frame1['Rango Edad']== '0-17')|(frame1['Rango Edad']== '18-29')]
frame1_029= frame1_029.groupby(['Departamento Hospedaje']).aggregate({'Femenino':np.sum,'Masculino': np.sum,
                                                                      'Total':np.sum}).reset_index()
sa_029= frame1_029[(frame1_029['Departamento Hospedaje']=='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA')|
                     (frame1_029['Departamento Hospedaje']=='SANTA CATALINA')]

frame1_029= frame1_029[(frame1_029['Departamento Hospedaje']!='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA') &
                     (frame1_029['Departamento Hospedaje']!='SANTA CATALINA')]

#base para el primer fram del grupo de 50 años o más
frame1_5070= frame1[(frame1['Rango Edad']== '50-59')|(frame1['Rango Edad']== '60-69')|(frame1['Rango Edad']== '70 o Más')]
frame1_5070= frame1_5070.groupby(['Departamento Hospedaje']).aggregate({'Femenino':np.sum,'Masculino': np.sum,
                                                                        'Total':np.sum}).reset_index()
sa_5070= frame1_5070[(frame1_5070['Departamento Hospedaje']=='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA')|
                     (frame1_5070['Departamento Hospedaje']=='SANTA CATALINA')]
frame1_5070= frame1_5070[(frame1_5070['Departamento Hospedaje']!='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA') &
                     (frame1_5070['Departamento Hospedaje']!='SANTA CATALINA')]

#procedo a añadir gráfico por gráfico por cada grupo a cada subplot
#custom data con la información por sexo para los hoover text
fig.add_trace(go.Choropleth(geojson=SA, locations=sa_3049['Departamento Hospedaje'], z=sa_3049['Total'],
                            customdata=np.stack((sa_3049['Femenino'], sa_3049['Masculino']), axis=-1),
                            text= sa_3049['Departamento Hospedaje'],
                            hovertemplate='<b>%{text}</b><br><b>Total:%{z}</b><br>Femenino:%{customdata[0]}'+\
                            '<br>Masculino: %{customdata[1]}<extra></extra>',
                           colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                          featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"
                           ),row=1, col=1)

fig.add_trace(go.Choropleth(geojson=departamentos, locations=frame1_3049['Departamento Hospedaje'], z=frame1_3049['Total'],
                            customdata=np.stack((frame1_3049['Femenino'], frame1_3049['Masculino']), axis=-1),
                            text= frame1_3049['Departamento Hospedaje'],
                            hovertemplate='<b>%{text}</b><br><b>Total:%{z}</b><br>Femenino:%{customdata[0]}'+\
                            '<br>Masculino: %{customdata[1]}<extra></extra>',
                           colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                          featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"
                           ),row=1, col=2)

fig.add_trace(go.Choropleth(geojson=SA, locations=sa_029['Departamento Hospedaje'], z=sa_029['Total'],
                            customdata=np.stack((sa_029['Femenino'], sa_029['Masculino']), axis=-1),
                            text= sa_029['Departamento Hospedaje'],
                            hovertemplate='<b>%{text}</b><br><b>Total:%{z}</b><br>Femenino:%{customdata[0]}'+\
                            '<br>Masculino: %{customdata[1]}<extra></extra>',
                           colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                          featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"
                           ),row=1, col=3)

fig.add_trace(go.Choropleth(geojson=departamentos, locations=frame1_029['Departamento Hospedaje'], z=frame1_029['Total'],
                            customdata=np.stack((frame1_029['Femenino'], frame1_029['Masculino']), axis=-1),
                            text= frame1_029['Departamento Hospedaje'],
                            hovertemplate='<b>%{text}</b><br><b>Total:%{z}</b><br>Femenino:%{customdata[0]}'+\
                            '<br>Masculino: %{customdata[1]}<extra></extra>',
                           colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                          featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"
                           ),row=1, col=4)

fig.add_trace(go.Choropleth(geojson=SA, locations=sa_5070['Departamento Hospedaje'], z=sa_5070['Total'],
                            customdata=np.stack((sa_5070['Femenino'], sa_5070['Masculino']), axis=-1),
                            text= sa_5070['Departamento Hospedaje'],
                            hovertemplate='<b>%{text}</b><br><b>Total:%{z}</b><br>Femenino:%{customdata[0]}'+\
                            '<br>Masculino: %{customdata[1]}<extra></extra>',
                           colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                          featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"
                           ),row=2, col=3)

fig.add_trace(go.Choropleth(geojson=departamentos, locations=frame1_5070['Departamento Hospedaje'], z=frame1_5070['Total'],
                            customdata=np.stack((frame1_5070['Femenino'], frame1_5070['Masculino']), axis=-1),
                            text= frame1_5070['Departamento Hospedaje'],
                            hovertemplate='<b>%{text}</b><br><b>Total:%{z}</b><br>Femenino:%{customdata[0]}'+\
                            '<br>Masculino: %{customdata[1]}<extra></extra>',
                           colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                          featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"
                           ),row=2, col=4)
```

In [34]:

```
#personalización de los mapas para que se ajuste a la localización de los datos Jsonn entre otros ajustes
fig.update_layout(
    title_text='<b>2018-2019 evolución trimestral por departamento del turismo asiático en Colombia<b>',
    titlefont=dict({'family': 'Trattatello', 'color': 'black', 'size': 22}),
    geo1 = dict(landcolor = 'lightgray',
        scope='south america', fitbounds="locations", visible=False, projection = go.layout.geo.Projection(
            type = 'equirectangular',
            scale=1)),
    geo2 = dict(landcolor = 'lightgray',
        scope='south america', fitbounds="locations", visible=False, projection = go.layout.geo.Projection(
            type = 'equirectangular',
            scale=1)),
    geo3 = dict(landcolor = 'lightgray',
        scope='south america', fitbounds="locations", visible=False, projection = go.layout.geo.Projection(
            type = 'equirectangular',
            scale=1)),
    geo4 = dict(landcolor = 'lightgray',
        scope='south america', fitbounds="locations", visible=False, projection = go.layout.geo.Projection(
            type = 'equirectangular',
            scale=1)),
    geo5 = dict(landcolor = 'lightgray',
        scope='south america', fitbounds="locations", visible=False, projection = go.layout.geo.Projection(
            type = 'equirectangular',
            scale=1)),
    geo6 = dict(landcolor = 'lightgray',
        scope='south america', fitbounds="locations", visible=False, projection = go.layout.geo.Projection(
            type = 'equirectangular',
            scale=1)),
    height=600, margin={"r":10,"t":50,"l":10,"b":50})

fig.add_annotation(x=0.24, y=1, font= dict({
              'size': 12,
              'color': 'rgb(116, 101, 130)',
                'family': 'Trattatello'}),
            align= 'center', text= 'Total entradas de personas originarias de Asia para 2018Q1: 21125',
                      xanchor='center',xref= 'paper', yanchor='bottom', yref='paper', showarrow=False )
fig.add_annotation(x=0.15, y=0.94, font= dict({'size': 17, 'family': 'Trattatello', 'color': 'black'}),text= 'Viajeros entre 30-49 años',
                      xanchor='center',xref= 'paper', yanchor='bottom', yref='paper', showarrow=False)
fig.add_annotation(x=0.70, y=0.96 , font= dict({'size': 17, 'family': 'Trattatello', 'color': 'black'}),text= 'Viajeros entre 0-29 años',
                      xanchor='center',xref= 'paper', yanchor='bottom', yref='paper', showarrow=False)
fig.add_annotation(x=0.70, y=0.48, font= dict({'size': 17, 'family': 'Trattatello', 'color': 'black'}),text= 'Viajeros entre 50 o más',
                      xanchor='center',xref= 'paper', yanchor='bottom', yref='paper', showarrow=False)
```

In [35]:

```
#Agregar los botones de play & pause a la vez que establecer la duración de cada frame
updatemenus= [{"buttons":[{"args":
                           [None, {"frame": {"duration": 3000,"redraw": True},
                                "fromcurrent": True,
                                   "transition": {"duration": 400, "easing": "linear-in-out"}}],
               "label": "Play", "method": "animate"},
           {"args": [[None], {"frame": {"duration": 0,"redraw": False},
                              "mode": "immediate", "transition": {"duration": 0}}],
               "label": "Pause",
               "method": "animate"}],
              "direction": "left", "pad": {"r": 10, "t": 87}, "showactive": True,
       "type": "buttons", "x": 0.1, "y": 0}]
```

In [36]:

```
#Agregar el slider de tiempo
sliders_dict = {
    "active": 0,
    "yanchor": "top",
    "xanchor": "left",
    "currentvalue": {
        "font": {"size": 17, 'family':'Lithos Pro'},
        "prefix": "Trimestre:",
        "visible": True,
        "xanchor": "right"
    },
    "transition": {"duration": 400, "easing": "linear-in"},
```

```
    "pad": {"b": 10, "t": 50},
    "len": 0.8,
    "x": 0.2,
    "y": 0,
    "steps": []}
```

In [37]:

```
#Hacer los frames
list_of_frames = []
for i in lista_tr:

    #Filtrar por trimestre
    qtr_data = bdf[bdf['Qtr'] == i]
    asia_sum= qtr_data['Total'].sum()
    qtr_3049= qtr_data[(qtr_data['Rango Edad']== '30-39')|(qtr_data['Rango Edad']== '40-49')]
    qtr_3049= qtr_3049.groupby(['Departamento Hospedaje']).aggregate({'Femenino':np.sum,'Masculino': np.sum,
                                                                      'Total':np.sum}).reset_index()
    sa_3049= qtr_3049[(qtr_3049['Departamento Hospedaje']=='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA')|
                  (qtr_3049['Departamento Hospedaje']=='SANTA CATALINA')]
    qtr_3049= qtr_3049[(qtr_3049['Departamento Hospedaje']!='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA') &
                  (qtr_3049['Departamento Hospedaje']!='SANTA CATALINA')]

    qtr_029= qtr_data[(qtr_data['Rango Edad']== '0-17')|(qtr_data['Rango Edad']== '18-29')]
    qtr_029= qtr_029.groupby(['Departamento Hospedaje']).aggregate({'Femenino':np.sum,'Masculino': np.sum,
                                                                    'Total':np.sum}).reset_index()
    sa_029= qtr_029[(qtr_029['Departamento Hospedaje']=='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA')|
                  (qtr_029['Departamento Hospedaje']=='SANTA CATALINA')]
    qtr_029= qtr_029[(qtr_029['Departamento Hospedaje']!='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA') &
                  (qtr_029['Departamento Hospedaje']!='SANTA CATALINA')]

    qtr_5070= qtr_data[(qtr_data['Rango Edad']== '50-59')|(qtr_data['Rango Edad']== '60-69')|
                       (qtr_data['Rango Edad']== '70 o Más')]
    qtr_5070= qtr_5070.groupby(['Departamento Hospedaje']).aggregate({'Femenino':np.sum,'Masculino': np.sum,
                                                                      'Total':np.sum}).reset_index()
    sa_5070= qtr_5070[(qtr_5070['Departamento Hospedaje']=='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA')|
                  (qtr_5070['Departamento Hospedaje']=='SANTA CATALINA')]
    qtr_5070= qtr_5070[(qtr_5070['Departamento Hospedaje']!='ARCHIPIELAGO DE SAN ANDRES PROVIDENCIA Y SANTA CATALINA') &
                  (qtr_5070['Departamento Hospedaje']!='SANTA CATALINA')]


    #Anexarlo a la lista de frames y el texto irá cambiando mostrando el total de entradas desde asia
    list_of_frames.append(go.Frame(
            layout=go.Layout(
            annotations= [{
            'text': "Total entradas de personas originarias de Asia para "+ '{}'.format(i)+': '+ "{}".format(asia_sum),
              'font': {
              'size': 12,
              'color': 'rgb(116, 101, 130)', 'family': 'Trattatello'},
            'showarrow': False,
            'align': 'center',
            'x': 0.24,
            'y': 1,}]),
        data=[go.Choropleth(geojson=SA, locations=sa_3049['Departamento Hospedaje'], z=sa_3049['Total'],
                            colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                        featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"),
              go.Choropleth(geojson=departamentos, locations=qtr_3049['Departamento Hospedaje'], z=qtr_3049['Total'],
                            colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                        featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"),
              go.Choropleth(geojson=SA, locations=sa_029['Departamento Hospedaje'], z=sa_029['Total'],
                            colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                        featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"),
                        go.Choropleth(geojson=departamentos, locations=qtr_029['Departamento Hospedaje'], z=qtr_029['Total'],
                            colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                        featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"),
              go.Choropleth(geojson=SA, locations=sa_5070['Departamento Hospedaje'], z=sa_5070['Total'],
                            colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                        featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas"),
                        go.Choropleth(geojson=departamentos, locations=qtr_5070['Departamento Hospedaje'], z=qtr_5070['Total'],
                            colorscale=[[0,'white'],[0.03,'gold'],[0.14,'tomato'],[0.4,'red'],[0.96,'blue'],[1,'mediumblue']],
                            marker_line_color='black',
                            zmax=105,
                            zmid=28,
                            zmin=-0.2,
                        featureidkey="properties.NOMBRE_DPT",
                            colorbar_title = "Turistas")],
              traces= [0,1,2,3,4,5],name= str(i)))


    # Unir los pasos del slider con cada frame
    slider_step = {"args": [
        [i],
        {"frame": {"duration": 400, "redraw": True},
         "mode": "immediate",
         "transition": {"duration": 400}}
    ],
                   "label": str(i), # Para mostrar el trimestre del frame en el slider
                   "method": "animate"}
    sliders_dict["steps"].append(slider_step)
```

In [38]:

```
fig.update(frames=list(list_of_frames)),
fig.update_layout(updatemenus=updatemenus, sliders= [sliders_dict])
fig.show()
```

A pesar de que el gráfico no refleja la realidad del turismo asiatico en los departamentos de Colombia si es interesante denotar la evolución de Santander cuyo reporte como departamento de hospedaje fue creciendo trimestralmente. También a través del gráfico se puede observar ciertas estacionalidades como por ejemplo la disminución de viajeros que reportan a La Guajira, Cordoba, Tolima y Cauca en los primeros trimestres de 2018 y 2019 y su aumento en los segundos trimestres de ambos años. Caso contrario con Nariño, San andrés, Amazonas, Sucre cuyo aumento es en los primeros trimestres.

Otra particularidad concierne a los rangos de edades, para el Archipielago de San Andrés, Providencia y Santa Catalina el rango de edad más joven es el que más lo reporta como hospedaje y esto se denota debido a que está mas cerca al rango de colores más alto, mientras en los otros dos grupos de mayor edad se tiene menores reportes acercandose a la escala de color más baja. Fue un ejercicio interesante con el cual pude aprender sobre gráficas utilizando mapas.

In [23]:
```
import datapane as dp
table    = dp.DataTable(bdf)
markdown = dp.Markdown("""El presente gráfico exhibe el código utilizado para realizar el mapa coroplético de los departamentos que los visitantes de origen asiatico reportaron como lugar de su hospedaje principal por lo tant
Este ejercicio tuvo como finalidad aplicar conocimientos en Python y aprender sobre gráficas utilizando mapas.

A pesar de que el gráfico no refleja la realidad del turismo asiatico en los departamentos de Colombia si es interesante denotar la evolución de Santander cuyo reporte como departamento de hospedaje fue creciendo trimestralme

Otra particularidad concierne a los rangos de edades, para el Archipielago de San Andrés, Providencia y Santa Catalina el rango de edad más joven es el que más lo reporta como hospedaje y esto se denota debido a que está mas
plot     = dp.Plot(fig)

report = dp.Report(markdown,plot,table)
report.publish(name = '2018-2019 evolución trimestral por departamento del turismo asiático en Colombia')
```

```
Publishing report and associated data - please wait..
Report successfully published at https://datapane.com/u/cristiancamhm/reports/2018-2019-evolucion-trimestral-por-departamento-del-turismo-asiatico-en-colombia/
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: