Solución propuesta al ejercicio práctico para la posición de Data Engineer Jr en Deacero.

Citando el problema

```
# Ejercicio práctico para Data Engineer Jr en Deacero.

Debe realizar un fork de este repositorio para desarrollar y entregar su trabajo.

1. Obtén los datos de las siguientes fuentes desde las apis:

| Dataset                  | Url                                                                  |
| ------------------------ | -------------------------------------------------------------------- |
| Lista Pasajeros 2016     | http://analytics.deacero.com/Api/GetApi/ApiPasajeros2016/api_key     |
| Lista Pasajeros 2017     | http://analytics.deacero.com/Api/GetApi/ApiPasajeros2017/api_key     |
| Lista Viajes 2016        | http://analytics.deacero.com/Api/GetApi/ApiVuelos2016/api_key        |
| Lista Viajes 2017        | http://analytics.deacero.com/Api/GetApi/ApiVuelos2017/api_key        |
| Lista Aerolíneas         | http://analytics.deacero.com/Api/GetApi/ApiLineaAerea/api_key        |

  Nota: El api_key válido se proporciona por correo. En caso de estar interesado en aplicar al test puede envia

2. Se tiene un requerimiento de análisis de las fuentes de datos de pasajeros y viajes. Como podrás observar, l
    - Unir cada conjunto de datos en una sola lista.
    - Explicar el proceso realizado.
    - En caso de detectar anomalías generadas por esta unión, deberás indicar el tipo de anomalía que se presen

3. De lo obtenido anteriormente se requiere relacionar las listas de vuelos y pasajeros, que permitan analizar
    - Explicar el proceso que utilizado para unir los pasajeros y los vuelos.
    - Qué tipo de relación y por qué.

4. Ahora se requiere que los datos consolidados de los vuelos y pasajeros se puedan unir con los datos de las L
    - Fecha del viaje
    - Clase
    - Precio
    - Ruta
    - Edad
    - Línea Aérea

   En esta parte deberás indicar:
```

- ¿Qué tipo de proceso consideraste para unir los datos que se piden?
- ¿Qué columnas utilizaste para lograr esa relación?
- ¿Qué tipo de unión utilizaste para unir los datos?
- ¿Qué tipo de proceso utilizaste para dejar únicamente las columnas que se piden?

5. Por último, se requiere el promedio semestral (el primer semestre es de Ene - Jun y el segundo es de Jul - D

Los ejercicios deben realizarse en Python, además de un documento donde se contesten las preguntas y se muestre

Una vez concluido el reto se debe comunicar al correo correspondiente con la liga al repositorio de github fina

Suerte a todos!!! ```

1. Obtener los datos.

Una vez leído el problema a resolver, el primer paso realizado fue la importación de las librerías a utilizar.

```
import requests
import csv
import json
import pandas as pd
from datetime import datetime
```

El primer paso es obtener los datos de las siguientes APIs:

Dataset	Url
Lista Pasajeros 2016	http://analytics.deacero.com/Api/GetApi/ApiPasajeros2016/api_key
Lista Pasajeros 2017	http://analytics.deacero.com/Api/GetApi/ApiPasajeros2017/api_key
Lista Viajes 2016	http://analytics.deacero.com/Api/GetApi/ApiVuelos2016/api_key
Lista Viajes 2017	http://analytics.deacero.com/Api/GetApi/ApiVuelos2017/api_key
Lista Aerolíneas	http://analytics.deacero.com/Api/GetApi/ApiLineaAerea/api_key

Para lo cual se crearon las siguientes variables:

```
URLListaPasajeros2016 = 'http://analytics.deacero.com/Api/GetApi/ApiPasajeros2016/{}'.format('ecfb5fc7
URLListaPasajeros2017 = 'http://analytics.deacero.com/Api/GetApi/ApiPasajeros2017/{}'.format('faabd632
URLListaViajes2016 = 'http://analytics.deacero.com/Api/GetApi/ApiVuelos2016/{}'.format('9ea3b836-6938-
```

```
URLListaViajes2017 = 'http://analytics.deacero.com/Api/GetApi/ApiVuelos2017/{}'.format('fc126260-1cf8-
URLListaAeorilineas = 'http://analytics.deacero.com/Api/GetApi/ApiLineaAerea/{}'.format('1a8d9e13-ce30
```

(La llave de la API fue proporcionada por correo electrónico).

Una vez teniendo el punto de acceso y las llaves se procedió a crear las funciones necesarias para obtener la información.

```
def GetURL(url):
    response = requests.get(url)
    content = response.content.decode("utf8")
    return content

def GetJSONFromURL(url):
    content = GetURL(url)
    js = json.loads(content)
    return js
```

Haciendo uso de dichas funciones podemos obtener la información que buscamos y almacenarla en variables para su posterior uso.

```
RawDataListaPasajeros2016 = GetJSONFromURL(URLListaPasajeros2016)
RawDataListaPasajeros2017 = GetJSONFromURL(URLListaPasajeros2017)
RawListaViajes2016 = GetJSONFromURL(URLListaViajes2016)
RawListaViajes2017 = GetJSONFromURL(URLListaViajes2017)
RawListaAeorilineas = GetJSONFromURL(URLListaAeorilineas)
```

Con esto se puede dar por completado el punto uno. Pero para evitar las llamadas innecesarias al servidor y una posible restricción de solicitudes por parte del servidor se decidió guardar la información en archivos *.csv*.

Para guardar la información se decidió imprimir el tipo de datos con el que se esta trabajando y así determinar la manera optima de crear los archivos necesarios.

```
print(type(RawDataListaPasajeros2016))
print(type(RawDataListaPasajeros2016[0]))
print(RawDataListaPasajeros2016[0])
```

```
<class 'list'>
<class 'dict'>
{'ID_Pasajero': 576, 'Pasajero': 'Danielle Thompson', 'Edad': 60}
```

```
print(type(RawDataListaPasajeros2017))
print(type(RawDataListaPasajeros2017[0]))
print(RawDataListaPasajeros2017[0])
```

```
<class 'list'>
<class 'dict'>
{'ID_Pasajero': 596, 'Pasajero': 'Javier Olson', 'Edad': 71}
```

```
print(type(RawListaViajes2016))
print(type(RawListaViajes2016[0]))
print(RawListaViajes2016[0])
```

```
<class 'list'>
<class 'dict'>
{'Cve_LA': 'SW', 'Viaje': '9/10/2016', 'Clase': 'Economy', 'Precio': 60, 'Ruta': 'DAL-MDW', 'Cve
```

```
print(type(RawListaViajes2017))
print(type(RawListaViajes2017[0]))
print(RawListaViajes2017[0])
```

```
<class 'list'>
<class 'dict'>
{'Cve_LA': 'SW', 'Viaje': '1/4/2017', 'Clase': 'Economy', 'Precio': 86, 'Ruta': 'DAL-ATL', 'Cve_
```

```
print(type(RawListaAeorilineas))
print(type(RawListaAeorilineas[0]))
print(RawListaAeorilineas[0])
```

```
<class 'list'>
<class 'dict'>
{'Code': 'AA', 'Linea_Aerea': 'American Airlines'}
```

Los resultados fueron similares para los cinco conjuntos de datos, a excepción de los cabezales que varía dependiendo del conjunto en cuestión.

- Cabezales para las lista de pasajeros: 'ID_Pasajero', 'Pasajero' y 'Edad'.
- Cabezales para las listas de viajes: 'Cve_LA', 'Viaje', 'Clase', 'Precio', 'Ruta' y 'Cve_Cliente'.
- Cabezales para la lista de aerolíneas: 'Code' y 'Linea_Aerea'.

Una vez teniendo los cabezales de los datos, se procedió a escribirlos en sus archivos *.csv* correspondientes.

```
with open('RawListaPasajeros2016.csv','w', newline='') as csvfile:
  cabezales = ['ID_Pasajero','Pasajero','Edad']
  writer = csv.DictWriter(csvfile, fieldnames=cabezales)
  writer.writeheader()
  for i in RawDataListaPasajeros2016:
    writer.writerow(i)
```

```
with open('RawListaPasajeros2017.csv','w', newline='') as csvfile:
  cabezales = ['ID_Pasajero','Pasajero','Edad']
  writer = csv.DictWriter(csvfile, fieldnames=cabezales)
  writer.writeheader()
  for i in RawDataListaPasajeros2017:
    writer.writerow(i)


with open('RawListaViajes2016.csv','w', newline='') as csvfile:
  cabezales = ['Cve_LA','Viaje','Clase','Precio','Ruta','Cve_Cliente']
  writer = csv.DictWriter(csvfile, fieldnames=cabezales)
  writer.writeheader()
  for i in RawListaViajes2016:
    writer.writerow(i)


with open('RawListaViajes2017.csv','w', newline='') as csvfile:
  cabezales = ['Cve_LA','Viaje','Clase','Precio','Ruta','Cve_Cliente']
  writer = csv.DictWriter(csvfile, fieldnames=cabezales)
  writer.writeheader()
  for i in RawListaViajes2017:
    writer.writerow(i)


with open('RawListaAeorilineas.csv','w', newline='') as csvfile:
  cabezales = ['Code','Linea_Aerea']
  writer = csv.DictWriter(csvfile, fieldnames=cabezales)
  writer.writeheader()
  for i in RawListaAeorilineas:
    writer.writerow(i)
```

Por último, para poder trabajar de manera más rápida se decidió utilizar variables del tipo *pandas.DataFrame*, obtenidas a partir de los archivos recién creados.

```
DFListaPasajeros2016 = pd.read_csv('RawListaPasajeros2016.csv', index_col=0)
DFListaPasajeros2017 = pd.read_csv('RawListaPasajeros2017.csv', index_col=0)
DFListaViajes2016 = pd.read_csv('RawListaViajes2016.csv',index_col=5)
DFListaViajes2017 = pd.read_csv('RawListaViajes2017.csv',index_col=5)
DFListaAeorilineas = pd.read_csv('RawListaAeorilineas.csv')
```

(La selección de las columnas indice tiene su justificación en el punto dos).

2.- Unir los conjuntos de datos.

En el punto dos del ejercicio se nos solicita hacer la unión de los conjuntos de datos una sola lista. Para lo cual se decidió unir las listas de los pasajeros entre sí y las listas de los vuelos entre sí. Esto resulta sencillo ya que los datos contienen las mismas columnas y como también se seleccionaron de antemano las mismas columnas indice lo único que hay que hacer es unir ambas listas con una

Para corroborar la anterior primero obtenemos las primeras filas de cada conjunto de datos a unir.

```
DFListaPasajeros2016.head()
```

ID_Pasajero	Pasajero	Edad
576	Danielle Thompson	60
579	Natalie Cuevas	49
683	John Murray	28
681	Michael Jacobs	24
592	Brian Hunt	40

```
DFListaPasajeros2017.head()
```

ID_Pasajero	Pasajero	Edad
596	Javier Olson	71
625	Monique Ramirez	35
637	Rob Beeghly	29
730	Timothy Moore	21
682	Scot Wooten	72

Una vez comprobado que ambos conjuntos de datos son similares, ya es seguro hacer la concatenación.

```
ListaPasajerosUnificada = pd.concat([DFListaPasajeros2016,DFListaPasajeros2017])
```

Lo que resulta en una lista similar:

```
ListaPasajerosUnificada.head()
```

ID_Pasajero	Pasajero	Edad
576	Danielle Thompson	60
579	Natalie Cuevas	49
683	John Murray	28
681	Michael Jacobs	24
502	Brian Hunt	40

Con ambas listas unidas se procedió a verificar que no tuvieran elementos duplicados, ya que en una lista como esta no es necesario que se repitan los datos.

```
ListaPasajerosUnificada.value_counts()
```

```
Pasajero         Edad
Scot Wooten      72      12
Anemone Ratner   39      10
Shahid Shariari  67       6
Rob Beeghly      29       5
Brandon Harris   68       4
                         ..
Michael Flores   46       1
Michael Jacobs   24       1
Michael Nguyen   66       1
Monique Ramirez  35       1
Aaron Bennett    43       1
Length: 138, dtype: int64
```

Como se puede ver hay cierto valores que se repiten, por lo cual se procede a eliminar dichos datos.

```
ListaPasajerosUnificada.drop_duplicates(keep='first', inplace=True)
```

```
ListaPasajerosUnificada.value_counts()
```

```
Pasajero           Edad
Zachary White      49      1
Erik Wheeler       30      1
Heather Morgan     22      1
Hannah Haley       53      1
Haley Watkins      57      1
                           ..
Monique Ramirez    35      1
Monique Smith      52      1
Mrs. Courtney Day  40      1
Nancy Lomonaco     34      1
Aaron Bennett      43      1
Length: 138, dtype: int64
```

La longitud de los datos nos indica que no se a eliminado ningún dato que no este duplicado, por lo que podemos continuar. Pero para confirmar que el resultado es el esperado se imprimen las primeras columnas:

```
ListaPasajerosUnificada.head()
```

ID_Pasajero	Pasajero	Edad
576	Danielle Thompson	60
579	Natalie Cuevas	49
683	John Murray	28
681	Michael Jacobs	24
592	Brian Hunt	40

El siguiente paso realizado fue guardar estos datos ya unificados en un nuevo archivo *.csv*.

```
ListaPasajerosUnificada.to_csv('Unif.csv')
```

```
# cabezalesListaPasajeros = ['ID_Pasajero','Pasajero','Edad']
# cabezalesListasViajes = ['Cve_LA','Viaje','Clase','Precio','Ruta','Cve_Cliente']
# cabezalesListaAeorilineas = ['Code','Linea_Aerea']
```

Este proceso se repitió con las lista de los viajes. Mostramos las primeras filas de la lista de viajes:

Lista viajes 2016:

```
DFListaViajes2016.head()
```

	Cve_LA	Viaje	Clase	Precio	Ruta

Lista viajes 2017:

553	SW	9/10/2016	Economv	60	DAL-MDW

```
DFListaViajes2017.head()
```

Cve_Cliente	Cve_LA	Viaje	Clase	Precio	Ruta
637	SW	1/4/2017	Economy	86	DAL-ATL
402	SW	1/4/2017	First Class	237	DAL-ATL
191	SW	1/2/2017	Economy	62	DAL-LGA
191	SW	1/2/2017	Economy	81	DAL-MDW
637	AA	1/4/2017	Business	169	DAL-LGA

Las concatenamos:

```
ListaDeViajesUnifacada = pd.concat([DFListaViajes2016,DFListaViajes2017])
```

Y verificamos el resultado

```
ListaDeViajesUnifacada.head()
```

Cve_Cliente	Cve_LA	Viaje	Clase	Precio	Ruta
553	SW	9/10/2016	Economy	60	DAL-MDW
554	AA	6/11/2016	Economy	150	DAL-SLC
554	DA	9/21/2016	Economy	68	DAL-AMA
556	UA	6/28/2016	Business	160	DAL-SLC
557	SW	6/25/2016	Economy	65	DAL-OKC

Verificamos duplicados.

```
ListaDeViajesUnifacada.value_counts()
```

```
Cve_LA  Viaje      Clase        Precio  Ruta
UA      6/28/2016  Business     160     DAL-SLC    1
AM      7/4/2017   Business     122     DAL-AUS    1
        6/22/2016  Economy      207     DAL-LGA    1
```

```
            6/26/2016  Economy      245     DAL-ELP    1
            6/5/2017   First Class  85      DAL-AMA    1
                                                      ..
      KL    6/12/2016  Economy      233     DAL-PHX    1
            6/14/2016  Economy      192     DAL-AUS    1
            6/16/2016  First Class  183     DAL-AUS    1
            6/16/2017  Business     55      DAL-MDW    1
      AA    1/1/2017   Business     115     DAL-LGA    1
      Length: 400, dtype: int64
```

Y por último guardamos los datos en un archivo *.csv*.

```
ListaDeViajesUnifacada.to_csv('ListaDeViajesUnificada.csv')
```

Como la lista de aerolíneas es solo una, no hay necesidad manipularla por ahora.

```
DFListaAeorilineas.head()
```

	Code	Linea_Aerea
0	AA	American Airlines
1	SW	Southwest
2	AM	Aeromexico
3	AV	Avianca
4	KL	KLM

3.- Relacionar lista de vuelos con pasajeros.

Teniendo ya lo solicitado en el punto dos, se procede a realizar el punto tres. El cuál pide relacionar los datos de los pasajeros con los datos de los vuelos.

Como se puede observar, en la información unificada de cada lista podemos encontrar que existe una columna en común, aunque no tengan el mismo nombre. En este caso las columnas que coinciden son las de 'Cve_Cliente' en la lista de viajes y 'ID_Pasajero' en la lista de pasajeros.

```
ListaPasajerosUnificada.head()
```

ID_Pasajero	Pasajero	Edad
576	Danielle Thompson	60
579	Natalie Cuevas	49

```
ListaDeViajesUnifacada.head()
```

Cve_Cliente	Cve_LA	Viaje	Clase	Precio	Ruta
553	SW	9/10/2016	Economy	60	DAL-MDW
554	AA	6/11/2016	Economy	150	DAL-SLC
554	DA	9/21/2016	Economy	68	DAL-AMA
556	UA	6/28/2016	Business	160	DAL-SLC
557	SW	6/25/2016	Economy	65	DAL-OKC

Una vez comprobado que tienen una columna en común, se procede a unir ambas tablas.

```
ViajesPasajerosUnificado = ListaDeViajesUnifacada.join(ListaPasajerosUnificada,how='left',lsuffix='Via
ViajesPasajerosUnificado.head()
```

	Cve_LA	Viaje	Clase	Precio	Ruta	Pasajero	Edad
38	DA	1/2/2017	Economy	84	DAL-AUS	Anemone Ratner	39
38	UA	1/2/2017	Economy	78	DAL-PHX	Anemone Ratner	39
38	SW	1/2/2017	Business	102	DAL-SEA	Anemone Ratner	39
38	SW	1/2/2017	First Class	222	DAL-HOU	Anemone Ratner	39
38	AA	1/2/2017	Economy	104	DAL-OKC	Anemone Ratner	39

En este caso se utilizó una unión por la izquierda para que sea la tabla de pasajeros quien se añada a la tabla de los viajes. Se unieron de esta manera porque se cree que el orden no afectaría al posterior análisis. Con la unión de estas columnas se termina el paso tres.

4.-Consolidar lista de pasajeros, lista de vuelos y aerolíneas.

El paso cuatro nos indica unir la información de los pasajeros y los vuelos con la información de las aerolíneas y en caso de no encontrar relación usar el valor "Otra". También se solicita dejar solo las

siguientes columnas:

- Fecha del viaje.
- Clase.
- Precio.
- Ruta.
- Edad.
- Linea aérea.

Antes de proceder a hacer la unión de las tablas primero hay que corregir el indice en la tabla resultante del paso anterior. Esto para hacer la búsqueda por indice más fácil. Para lo cual hay que restablecer el indicie y renombrar la columna resultante.

```
ViajesPasajerosUnificado.reset_index(inplace=True)
```

```
ViajesPasajerosUnificado.head()
```

	index	Cve_LA	Viaje	Clase	Precio	Ruta	Pasajero	Edad
0	38	DA	1/2/2017	Economy	84	DAL-AUS	Anemone Ratner	39
1	38	UA	1/2/2017	Economy	78	DAL-PHX	Anemone Ratner	39
2	38	SW	1/2/2017	Business	102	DAL-SEA	Anemone Ratner	39
3	38	SW	1/2/2017	First Class	222	DAL-HOU	Anemone Ratner	39
4	38	AA	1/2/2017	Economy	104	DAL-OKC	Anemone Ratner	39

```
ViajesPasajerosUnificado.rename(columns={'index':'ID_Pasajero'}, inplace=True)
```

```
ViajesPasajerosUnificado.head()
```

	ID_Pasajero	Cve_LA	Viaje	Clase	Precio	Ruta	Pasajero	Edad
0	38	DA	1/2/2017	Economy	84	DAL-AUS	Anemone Ratner	39
1	38	UA	1/2/2017	Economy	78	DAL-PHX	Anemone Ratner	39
2	38	SW	1/2/2017	Business	102	DAL-SEA	Anemone Ratner	39
3	38	SW	1/2/2017	First Class	222	DAL-HOU	Anemone Ratner	39
4	38	AA	1/2/2017	Economy	104	DAL-OKC	Anemone Ratner	39

Solo como medida preventiva, guardamos la tabla generada en un nuevo archivo *.csv*.

```
ViajesPasajerosUnificado.to_csv('ViajesUnif.csv')
```

```
DFListaAeorilineas.head()
```

	Code	Linea_Aerea
0	AA	American Airlines
1	SW	Southwest
2	AM	Aeromexico
3	AV	Avianca
4	KL	KLM

Se observa que la columna 'Code' corresponde con la columna 'Cve_LA' por lo que se procede a renombra la columna 'Code'.

```
DFListaAeorilineas.rename(columns={'Code':'Cve_LA'},inplace=True)
```

Comprobamos el resultado.

```
DFListaAeorilineas.head()
```

	Cve_LA	Linea_Aerea
0	AA	American Airlines
1	SW	Southwest
2	AM	Aeromexico
3	AV	Avianca
4	KL	KLM

Nos aseguramos que ambas tablas contengan dichas columnas.

```
ViajesPasajerosUnificado.head()
```

	ID_Pasajero	Cve_LA	Viaje	Clase	Precio	Ruta	Pasajero	Edad

Ya teniendo una columan en común, se procede a hacer la unión.

```
InfoUnificada = ViajesPasajerosUnificado.merge(DFListaAeorilineas,how='left',on='Cve_LA')
```

3	38	SW	1/2/2017	First Class	222	DAL-HOU	Anemone Ratner	39

Nuevamente se hizo unión por la izquierda, ya que el orden no supone un problema.

Comprobamos el resultado.

```
InfoUnificada.head()
```

	ID_Pasajero	Cve_LA	Viaje	Clase	Precio	Ruta	Pasajero	Edad	Linea_Aerea
0	38	DA	1/2/2017	Economy	84	DAL-AUS	Anemone Ratner	39	NaN
1	38	UA	1/2/2017	Economy	78	DAL-PHX	Anemone Ratner	39	NaN
2	38	SW	1/2/2017	Business	102	DAL-SEA	Anemone Ratner	39	Southwest
				First		DAL	Anemone		

Ahora hay que cambiar los elementos vacios (NaN) por la leyenda 'Otra'.

```
InfoUnificada.fillna('Otra',inplace=True)
```

Comprobamos el resultado.

```
InfoUnificada.head()
```

	ID_Pasajero	Cve_LA	Viaje	Clase	Precio	Ruta	Pasajero	Edad	Linea_Aerea
0	38	DA	1/2/2017	Economy	84	DAL-AUS	Anemone Ratner	39	Otra
1	38	UA	1/2/2017	Economy	78	DAL-PHX	Anemone Ratner	39	Otra
2	38	SW	1/2/2017	Business	102	DAL-SEA	Anemone Ratner	39	Southwest
				First		DAL	Anemone		

Guardamos esta tabla como medida preventiva.

```
InfoUnificada.to_csv('infoUnificada.csv')
```

El paso siguiente es quitar las columnas que no se estan solicitando.

```
InfoUnificada.drop(columns=['ID_Pasajero','Cve_LA','Pasajero'], inplace=True)
```

Comprobamos el resultado.

```
InfoUnificada.head()
```

	Viaje	Clase	Precio	Ruta	Edad	Linea_Aerea
0	1/2/2017	Economy	84	DAL-AUS	39	Otra
1	1/2/2017	Economy	78	DAL-PHX	39	Otra
2	1/2/2017	Business	102	DAL-SEA	39	Southwest
3	1/2/2017	First Class	222	DAL-HOU	39	Southwest
4	1/2/2017	Economy	104	DAL-OKC	39	American Airlines

Aquí se nos pide contestar a las siguientes preguntas:

- ¿Qué tipo de proceso consideraste para unir los datos que se piden?
  - En este caso se utilizó el método *merge*.
- ¿Qué columnas utilizaste para lograr esa relación?
  - Aquí se tomaron en cuenta las columnas de la clave de la aerolínea.
- ¿Qué tipo de unión utilizaste para unir los datos?
  - Se utilizó una union por izquierda.
- ¿Qué tipo de proceso utilizaste para dejar únicamente las columnas que se piden?
  - El método *drop* fue el que se utilizó para dejar solo las columnas solicitadas.

5.- Realizar un promedio semestral.

En este paso se nos pide realizar un promedio semestral utilizando las columnas 'Año', 'Clase', 'Ruta y 'Linea Aerea'. Para lo cual se determino que la columna que contiene las fechas de los viajes sería de utilidad, por lo que se procede a trabajar sobre la misma. Para poder manipular las fechas de manera más práctica se utilizará la librería de *datatime*. El primer paso es comprobar el tipo de datos que se tiene en la columna de 'Viaje'.

```
type(InfoUnificada['Viaje'])
```

```
pandas.core.series.Series
```

```
type(InfoUnificada['Viaje'][0])
```

```
str
```

Como el tipo de datos de dicha columna es el de *str* hace falta cambiarlo al tipo de datos *Timestamp* para trabajar de manera más eficiente. Para lo cual se aplica la siguiente función:

```
InfoUnificada['Viaje'] = InfoUnificada['Viaje'].apply(lambda x: datetime.strptime(x,'%m/%d/%Y'))
```

Con esta función tomamos cada valor de la columna 'Viaje', le aplicacmos la función *strptime* de la librería *datatime* para que lo convierta en el tipo de datos esperado.

Comprobamos el resultado

```
type(InfoUnificada['Viaje'][0])
```

```
pandas._libs.tslibs.timestamps.Timestamp
```

Con este tipo de datos podemos obtener fechas de manera mucho más sencilla.

```
InfoUnificada['Viaje'][0].year
```

```
2017
```

Teniendo las columnas con el tipo de datos deseado, se procede a creer las columnas faltantes para hacer el analisis

```
InfoUnificada['Año'] = InfoUnificada['Viaje'].apply(lambda x: x.year)
```

```
InfoUnificada['Semestre'] = InfoUnificada['Viaje'].apply(lambda x: 'Ene - Jun' if (x.month < 7) else '
```

Comprobamos el resultado.

```
InfoUnificada.head()
```

	Viaje	Clase	Precio	Ruta	Edad	Linea_Aerea	Año	Semestre
0	2017-01-02	Economy	84	DAL-AUS	39	Otra	2017	Ene - Jun
1	2017-01-02	Economy	78	DAL-PHX	39	Otra	2017	Ene - Jun
2	2017-01-02	Business	102	DAL-SEA	39	Southwest	2017	Ene - Jun
3	2017-01-02	First Class	222	DAL-HOU	39	Southwest	2017	Ene - Jun
4	2017-01-02	Economy	104	DAL-OKC	39	American Airlines	2017	Ene - Jun

Y por ultimo se realiza el analisis solicitado.

```
UnificadaPorClase = InfoUnificada.groupby(['Clase','Ruta','Linea_Aerea','Año','Semestre']).agg({'Precio
UnificadaPorClase.head(20)
```

Clase	Ruta	Linea_Aerea	Año	Semestre	Precio
Business	DAL-AMA	Aeromexico	2017	Ene - Jun	146.0
				Jul - Dic	204.0
		American Airlines	2016	Ene - Jun	58.0
				Jul - Dic	186.0
		Avianca	2016	Ene - Jun	132.0
			2017	Jul - Dic	206.0
		KLM	2016	Ene - Jun	184.0
			2017	Jul - Dic	121.0
		Southwest	2017	Ene - Jun	163.0
	DAL-ATL	Aeromexico	2016	Ene - Jun	218.0
			2017	Jul - Dic	201.0
		American Airlines	2016	Ene - Jun	102.0