PROYECTO 03 – DATA ANALYST

Contexto

La **Organización de Aviación Civil Internacional (OACI)**, organismo de la Organización de las Naciones Unidas, quiere investigar en profundidad los accidentes producidos desde inicios del siglo XX. Para ello, les solicita la elaboración de un informe y un dashboard interactivo que recopile tal información.

La OACI únicamente cuenta con un dataset sobre datos de accidentes de aviones, pero insta a la consultora de datos -de la que forman parte- que intente cruzar esta información con otras fuentes de su interés. Esto con el objetivo de obtener mayor claridad y consistencia en los fundamentos del estudio.

Propuesta de trabajo

A raíz de esta solicitud, nuestro Project Manager nos encarga una serie de tareas a cumplir:

- Realizar un EDA con el dataset provisto, junto con un reporte de calidad y diccionario de datos
- Buscar y relacionar información relevante con los eventos
- Crear una base de datos en un motor SQL e ingestar el csv procesado
- Elaborar un dashboard e idear un storytelling con el objetivo de presentarlo ante la OACI
- Adjuntar todo el trabajo en un repositorio de GitHub

ARCHIVOS ADJUNTOS DEL TRABAJO:

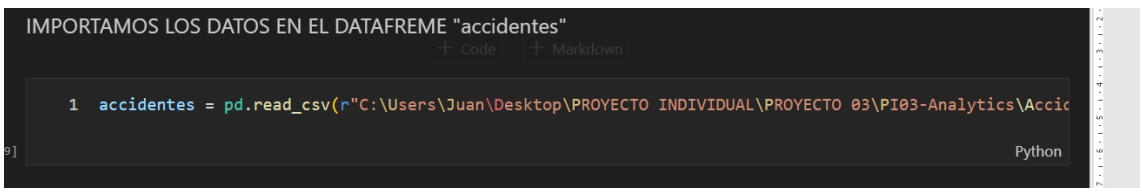
-PROYECTO03.ipynb

-Análisis de calidad, diccionario de datos y sustentación.pdf

-ANALISIS.pbix

-AccidentesAviones.csv

EDA:



Primer importamos los datos en un data frame que llamaremos “accidentes”

```
[40]  1  accidentes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5008 entries, 0 to 5007
Data columns (total 18 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Unnamed: 0             5008 non-null   int64
 1   fecha                  5008 non-null   object
 2   HORA declarada         5008 non-null   object
 3   Ruta                   5008 non-null   object
 4   OperadOR               5008 non-null   object
 5   flight_no              5008 non-null   object
 6   route                  5008 non-null   object
 7   ac_type                5008 non-null   object
 8   registration           5008 non-null   object
 9   cn_ln                  5008 non-null   object
 10  all_aboard             5008 non-null   object
 11  PASAJEROS A BORDO      5008 non-null   object
 12  crew_aboard            5008 non-null   object
 13  cantidad de fallecidos 5008 non-null   object
 14  passenger_fatalities   5008 non-null   object
 15  crew_fatalities        5008 non-null   object
 16  ground                 5008 non-null   object
 17  summary                5008 non-null   object
dtypes: int64(1), object(17)
memory usage: 704.4+ KB
```

Mediante info() conocemos que tenemos 5,008 registros (filas) y 17 columnas. De las 17 columnas, solamente 1 es del tipo entero y las otras 16 son texto. Además, podemos observas que no hay datos nulos (eso es bueno), no obstante, vamos a visualizar el data frame

```
1 accidentes.head(3)
```

[41] Python

	Unnamed: 0	fecha	HORA declarada	Ruta	OperadOR	flight_no	route	ac_type	registration	cn_ln	all_aboard	PASAJEROS A BORDO	crew_ab
0	0	September 17, 1908	1718	Fort Myer, Virginia	Military - U.S. Army	?	Demonstration	Wright Flyer III	?	1	2	1	
1	1	September 07, 1909	?	Juvisy-sur-Orge, France	?	?	Air show	Wright Byplane	SC1	?	1	0	
2	2	July 12, 1912	0630	Atlantic City, New Jersey	Military - U.S. Navy	?	Test flight	Dirigible	?	?	5	0	

```
1 accidentes.head(3)
```

Python

route	ac_type	registration	cn_ln	all_aboard	PASAJEROS A BORDO	crew_aboard	cantidad de fallecidos	passenger_fatalities	crew_fatalities	ground	summary
nstration	Wright Flyer III	?	1	2	1	1	1	1	0	0	During a demonstration flight, a U.S. Army fly...
Air show	Wright Byplane	SC1	?	1	0	1	1	0	0	0	Eugene Lefebvre was the first pilot to ever be...
Test flight	Dirigible	?	?	5	0	5	5	0	5	0	First U.S. dirigible Akron exploded just offsh...

Imprimimos las 3 primeras filas de accidentes, vemos que hay columnas que eran texto pero que contienen números. Además, si bien es cierto que supuestamente no hay datos nulos, estos aparentemente fueron sustituidos por “?”. Por último la columna “Unmamed: 0” está como un índice.

En este paso, dropearemos la columna “Unnamed: 0”, ya que no aporta en el análisis. Luego pasamos a estandarizar los nombres de las columnas y las vamos a renombrar de acuerdo al dato que tiene.

```
1 accidentes.head()
```

	FECHA	HORA	UBICACION	OPERADOR	N.VUELO	TIPO_VUELO	TIPO_NAVE	REGISTRO	CN_LN	TOTAL_A_BORDO	PA
0	September 17, 1908	1718	Fort Myer, Virginia	Military - U.S. Army	?	Demonstration	Wright Flyer III	?	1	2	
1	September 07, 1909	?	Juvisy-sur-Orge, France	?	?	Air show	Wright Byplane	SC1	?	1	
2	July 12, 1912	0630	Atlantic City, New Jersey	Military - U.S. Navy	?	Test flight	Dirigible	?	?	5	
3	August 06, 1913	?	Victoria, British Columbia, Canada	Private	?	?	Curtiss seaplane	?	?	1	
4	September 09, 1913	1830	Over the North Sea	Military - German Navy	?	?	Zeppelin L-1 (airship)	?	?	20	

```
1 accidentes.head()
```
Python

ORDO	TRIPULACION_A_BORDO	CANTIDAD_FALLECIDOS	PASAJEROS_FALLECIDOS	TRIPULACION_FALLECIDOS	FALLECIDOS_EN_SUELO	RESUMEN
1	1	1	1	0	0	During a demonstration flight, a U.S. Army fly...
0	1	1	0	0	0	Eugene Lefebvre was the first pilot to ever be...
0	5	5	0	5	0	First U.S. dirigible Akron exploded just offsh...
0	1	1	0	1	0	The first fatal airplane accident in Canada oc...
?	?	14	?	?	0	The airship flew into a thunderstorm and encou...

Ahora como vimos las columnas ya fueron renombradas de acuerdo al dato que contienen. El siguiente paso es ir estandarizando los datos, empezamos por la columna “UBICACIÓN”, nos damos cuenta que existen ciudades con sus respectos países, pero por ejemplo para cuando está una ubicación de Estados Unidos, sólo están como ciudades y estado. Para el análisis vamos a requerir las ubicaciones por países.

```
[47]
def split_country(x):
    a = x.split(",")[-1]
    return a.replace(" ", "")

accidentes["UBICACION"].apply(split_country)
```

```
0              Virginia
1                France
2             NewJersey
3                Canada
4       OvertheNorthSea
              ...
5003             Alaska
5004            Nigeria
5005            Myanmar
5006        Philippines
5007             Russia
Name: UBICACION, Length: 5008, dtype: object
```

Creamos una función que nos permita obtener la última palabra de la cela, observamos que se guardaron Virginia, France, NewJersey, Canada, entre otros.

```
[48]
accidentes['PAIS'] = accidentes['UBICACION'].apply(split_country)
```

Python

```
[49]
x = "Alabama, Alaska, AmericanSamoa, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, DistrictofColumbia, Florida,
Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Minor OutlyingIslands,
Nebraska, Nevada, NewHampshire, NewJersey, NewMexico, NewYork, NorthCarolina, NorthDakota, NorthernMarianaIslands, Ohio, Oklahoma,
 Rhode Island, South Carolina, South Dakota, Tennessee, Texas, U.S.VirginIslands, Utah, Vermont, Virginia, Washington, West Virgi
x=x.split(", ")
x=pd.Series(x)
```

Python

```
[50]
indices = accidentes['PAIS'].isin(x)
accidentes.loc[indices ,'PAIS'] = 'USA'
accidentes.head()
```

Python

Ahora vemos a cambiar los datos que están como nombres de estado, por el de “USA”, luego imprimos

```
1 indices = accidentes['PAIS'].isin(x)
2 accidentes.loc[indices ,'PAIS'] = 'USA'
3 accidentes.head()
```

[50] Python

:ION_A_BORDO	CANTIDAD_FALLECIDOS	PASAJEROS_FALLECIDOS	TRIPULACION_FALLECIDOS	FALLECIDOS_EN_SUELO	RESUMEN	PAIS
1	1	1	0	0	During a demonstration flight, a U.S. Army fly...	USA
1	1	0	0	0	Eugene Lefebvre was the first pilot to ever be...	France
5	5	0	5	0	First U.S. dirigible Akron exploded just offsh...	USA
1	1	0	1	0	The first fatal airplane accident in Canada oc...	Canada
?	14	?	?	0	The airship flew into a thunderstorm and encou...	OvertheNorthSea

Vemos que ahora ya cambiaron los nombres de los estados por USA

```
1 accidentes['OPERADOR'].str.contains("Military","military").value_counts()
```

[51]

```
False    4246
True      762
Name: OPERADOR, dtype: int64
```

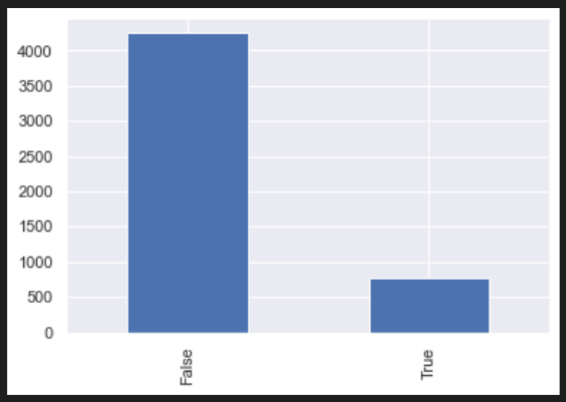
Ahora queremos extraer y conocer cuántos vuelos fueron de clase militar o no. Acá vamos a contar las filas que contengan “Military” y “military”. Nos brinda un resultado de 4,246 como falsos y 762 como verdaderos. Nos damos cuenta que efectivamente podemos usar esta información puede aportar en el análisis, para determinar si fueron vuelos militares o no militares.

```
[53]
1  a = accidentes['OPERADOR'].str.contains("Military")
2
3  accidentes["MILITAR"] = a
4  accidentes["MILITAR"].value_counts()
```

```
False    4246
True      762
Name: MILITAR, dtype: int64
```

```
[54]
1  accidentes['MILITAR'].value_counts().plot(kind='bar')
```

```
<AxesSubplot:>
```



Entonces creamos una variable que contenga los resultados llamada “a”, luego pasamos a guardar los resultados en una nueva columna que llamaremos “MILITAR”, finalmente para imprimir usamos gráficos de barras para apreciar mejor las cantidades y la diferencia.

```
[55]
1  accidentes.head(3)
```

Python

ION_A_BORDO	CANTIDAD_FALLECIDOS	PASAJEROS_FALLECIDOS	TRIPULACION_FALLECIDOS	FALLECIDOS_EN_SUELO	RESUMEN	PAIS	MILITAR
1	1	1	0	0	During a demonstration flight, a U.S. Army fly...	USA	True
1	1	0	0	0	Eugene Lefebvre was the first pilot to ever be...	France	False
5	5	0	5	0	First U.S. dirigible Akron exploded just offsh...	USA	True

Vemos que se creó la nueva columna y se guardaron los resultados correctamente

```python
accidentes['FECHA'] = pd.to_datetime(accidentes['FECHA'])
accidentes["AÑO"] = accidentes['FECHA'].apply(lambda x: x.year)
accidentes["MES"] = accidentes["FECHA"].apply(lambda x: x.month)
accidentes["DIA"] = accidentes["FECHA"].apply(lambda x: x.day)
accidentes.head()
```

Ahora queremos extraer el año, mes y día de la columna “FECHA” y luego imprimimos

```python
accidentes.head()
```

Python

ALLECIDOS	PASAJEROS_FALLECIDOS	TRIPULACION_FALLECIDOS	FALLECIDOS_EN_SUELO	RESUMEN	PAIS	MILITAR	AÑO	MES	DIA
1	1	0	0	During a demonstration flight, a U.S. Army fly...	USA	True	1908	9	17
1	0	0	0	Eugene Lefebvre was the first pilot to ever be...	France	False	1909	9	7
5	0	5	0	First U.S. dirigible Akron exploded just offsh...	USA	True	1912	7	12
1	0	1	0	The first fatal airplane accident in Canada oc...	Canada	False	1913	8	6
14	?	?	0	The airship flew into a thunderstorm and encou...	OvertheNorthSea	True	1913	9	9

Vemos que se crearon nuevas columnas con los datos respectivos para cada una.

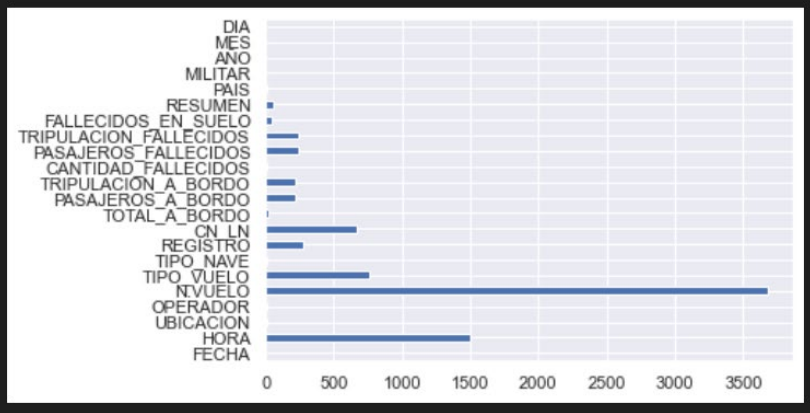
```
v1 =accidentes[accidentes[['FECHA', 'HORA', 'UBICACION', 'OPERADOR', 'N.VUELO', 'TIPO_VUELO',
       'TIPO_NAVE', 'REGISTRO', 'CN_LN', 'TOTAL_A_BORDO', 'PASAJEROS_A_BORDO',
       'TRIPULACION_A_BORDO', 'CANTIDAD_FALLECIDOS', 'PASAJEROS_FALLECIDOS',
       'TRIPULACION_FALLECIDOS', 'FALLECIDOS_EN_SUELO', 'RESUMEN', 'PAIS']]=="?"].count()
v1
```

```
FECHA                      0
HORA                    1504
UBICACION                  5
OPERADOR                  10
N.VUELO                 3682
TIPO_VUELO               762
TIPO_NAVE                 13
REGISTRO                 272
CN_LN                    667
TOTAL_A_BORDO             17
PASAJEROS_A_BORDO        221
TRIPULACION_A_BORDO      219
CANTIDAD_FALLECIDOS        8
PASAJEROS_FALLECIDOS     235
TRIPULACION_FALLECIDOS   235
FALLECIDOS_EN_SUELO       44
RESUMEN                   59
PAIS                       5
MILITAR                    0
AÑO                        0
MES                        0
DIA                        0
dtype: int64
```

Ahora para conocer con qué columnas vamos a trabajar queremos conocer cuántos “?” hay por cada una. Podemos ver que las columnas que no tienen menos de 50 “?” son FECHA, PAIS, OPERADOR, TIPO_NAVE, TOTAL_A_BORDO, CANTIDAD_FALLECIDOS, FALLECIDOS_EN_SUELO, MILITAR, AÑO, MES, DIA.

```
v1.plot(kind='barh')
```

```
<AxesSubplot:>
```



Imprimimos para apreciar mejor las cantidades, por lo antes descrito usaremos esas columnas para el análisis, ya que contienen la menor cantidad de datos faltantes o “?”.

```
accidentes["TOTAL_A_BORDO"] = accidentes["TOTAL_A_BORDO"].replace({ "?": 0})
accidentes["CANTIDAD_FALLECIDOS"] = accidentes["CANTIDAD_FALLECIDOS"].replace({ "?": 0})
accidentes["FALLECIDOS_EN_SUELO"] = accidentes["FALLECIDOS_EN_SUELO"].replace({ "?": 0})

✓ 0.7s
```

Ahora vamos a pasar estas columnas que están como strings a int, no obstante para cambiar el tipo de datos, debemos cambiar los “?” a 0, esto no afectará en el análisis ya que la columna que tiene la mayor cantidad de ese valor es FALLECIDOS_EN_SUELO con 44, no representa ni el 0.88% del total.

```
accidentes['TOTAL_A_BORDO'] = accidentes['TOTAL_A_BORDO'].astype(int)
accidentes['CANTIDAD_FALLECIDOS'] = accidentes['CANTIDAD_FALLECIDOS'].astype(int)
accidentes['FALLECIDOS_EN_SUELO'] = accidentes['FALLECIDOS_EN_SUELO'].astype(int)

✓ 0.8s
```

Ahora los pasamos a enteros para que en power bi podamos analizar correctamente.

```
RangeIndex: 5008 entries, 0 to 5007
Data columns (total 22 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   FECHA                   5008 non-null   datetime64[ns]
 1   HORA                    5008 non-null   object
 2   UBICACION               5008 non-null   object
 3   OPERADOR                5008 non-null   object
 4   N.VUELO                 5008 non-null   object
 5   TIPO_VUELO              5008 non-null   object
 6   TIPO_NAVE               5008 non-null   object
 7   REGISTRO                5008 non-null   object
 8   CN_LN                   5008 non-null   object
 9   TOTAL_A_BORDO           5008 non-null   int32
 10  PASAJEROS_A_BORDO       5008 non-null   object
 11  TRIPULACION_A_BORDO     5008 non-null   object
 12  CANTIDAD_FALLECIDOS     5008 non-null   int32
 13  PASAJEROS_FALLECIDOS    5008 non-null   object
 14  TRIPULACION_FALLECIDOS  5008 non-null   object
 15  FALLECIDOS_EN_SUELO     5008 non-null   int32
 16  RESUMEN                 5008 non-null   object
 17  PAIS                    5008 non-null   object
 18  MILITAR                 5008 non-null   bool
 19  AÑO                     5008 non-null   int64
...
 20  MES                     5008 non-null   int64
 21  DIA                     5008 non-null   int64
dtypes: bool(1), datetime64[ns](1), int32(3), int64(3), object(14)
memory usage: 768.0+ KB
```

Por último verificamos que se cambiaron los tipos de datos, llegamos al final de esta parte que es la preparación para un correcto análisis.

DICCIONARIO DE DATOS:

NOMBRE COLUMNA	CANTIDAD DE DATOS "?"	TIPO	DESCRIPCIÓN	NOMBRE ANTERIOR	¿SE USARÁ PARA ANÁLISIS?
COLUMNA ELIMINADA	0	-	Se eliminó, no aporta	Unnamed: 0	No
FECHA	0	Fecha	Fecha del vuelo	fecha	Sí
HORA	1504	Texto	Hora del vuelo	HORA declarada	No
UBICACIÓN	5	Texto	Ubicación de vuelo	Ruta	Sí
OPERADOR	10	Texto	Nombre del operador de vuelo	OperadOR	Sí
N. VUELO	3682	Texto	N. de vuelo	flight_no	No
TIPO_VUELO	762	Texto	Tipo, clase de vuelo	route	No
TIPO_NAVE	13	Texto	Tipo, modelo de nave	ac_type	Sí
REGISTRO	272	Texto	Código registro	registration	No
CN_LN	667	Texto	-	cn_ln	No
TOTAL_A_BORDO	17	Int	Total personas a bordo de nave	all_aboard	Sí
PASAJEROS_A_BORDO	221	Int	Total pasajeros a bordo en nave	PASAJEROS A BORDO	No
TRIPULACION_A_BORDO	219	Int	Total tripulación a bordo en nave	crew_aboard	No
CANTIDAD_FALLECIDOS	8	Int	Total de fallecidos en nave	cantidad de fallecidos	Sí
PASAJEROS_FALLECIDOS	235	Int	Total de pasajeros fallecidos en nave	Passenger_fatalities	No
TRIPULACION_FALLECIDOS	235	Int	Total de tripulación fallecida en nave	crew_fatalities	No
FALLECIDOS_EN_SUELO	44	Int	Total fallecidos en suelo	ground	Sí
RESUMEN	59	Texto	Descripción del accidente en inglés	Summary	No
PAIS	5	Texto	País del vuelo	-	Sí
MILITAR	0	Bool	Si vuelo fue militar o no	-	Sí
AÑO	0	Int	Año de vuelo	-	Sí
MES	0	Int	Mes de vuelo	-	Sí
DIA	0	Int	Dia de vuelo	-	Sí

CONCLUSIÓN:

Habiendo conocidos los tipos de datos, cantidad y calidad de datos por cada columna, creado columnas para analizar. Se determinó que se usarán 12 del total de 22 columnas, ya que cumplen con los requisitos de tener pocos registros nulos (representados con "?") la mayoría de ellas tiene 0 datos nulos , sin embargo algunas si tienen, aunque no son más de 44 datos de los 5008 que hay por cada columna, esto quiere decir que estos dato nulos representan un 0.8786%, esto quiere decir que no representan ni siquiera el 1%, por lo tanto el análisis que se hará tendrá poco ruido y será bastante preciso. Analizaremos las cantidades de fallecidos según tipo de vuelo (militar o no), por países, por operador, por fecha de vuelo, tipo de nave, total de personas a bordo y si fallecieron en suelo u otra zona.