Carrer d'illueca, 28
03206 Elx, Espanya
03013224.secret@gva.es
(+34) 966.912.260

EDA Bomberos San Francisco

El alcalde de San Francisco ha recibido algunas críticas sobre el departamento de bomberos, ya que hay unos vecinos que dicen que cada vez que han requerido de sus servicios han tardado mucho en atender sus peticiones.

El jefe del departamento, sabiendo de la fortaleza de su equipo requiere tus servicios para demostrar que los vecinos se equivocan y que, en su gran mayoría, la atención es buena.

Para ello, tenemos disponibles los datos de las llamadas atendidas por los bomberosde la ciudad de San Francisco (https://data.sfgov.org/Public-Safety/Fire-Department- Calls-for-Service/nuek-vuh3).

Para resolver este problema se pide:

1.- Carga los datos en un almacenamiento distribuido (HDFS / S3).
2.- Realiza un proceso EDA sobre los datos (mediante Pandas o Spark).
3.- Crea un cuadro mandos.

Adrian Espi Peña
Inteligencia Artificial y Big Data
05/06/2021

Pautas de Entrega:

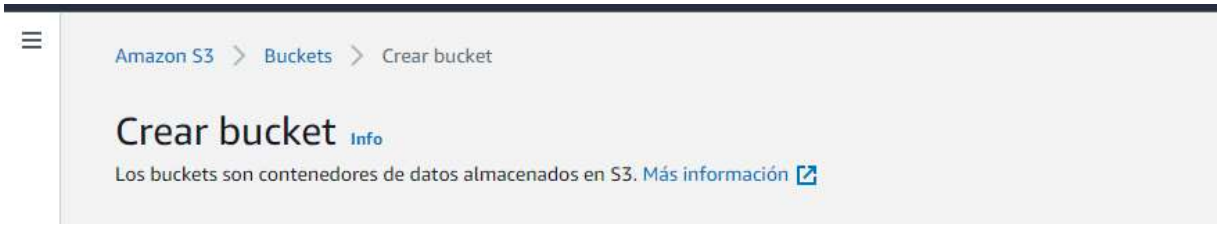
Como entregable sólo os vamos a pedir el código fuente/datasets y un pequeño documento donde nos contéis qué habéis hecho, las decisiones que habéis tomado y el resultado que habéis obtenido (no tiene por qué ser un resultado exitoso, el trayecto puede aportar tanto valor como el destino).

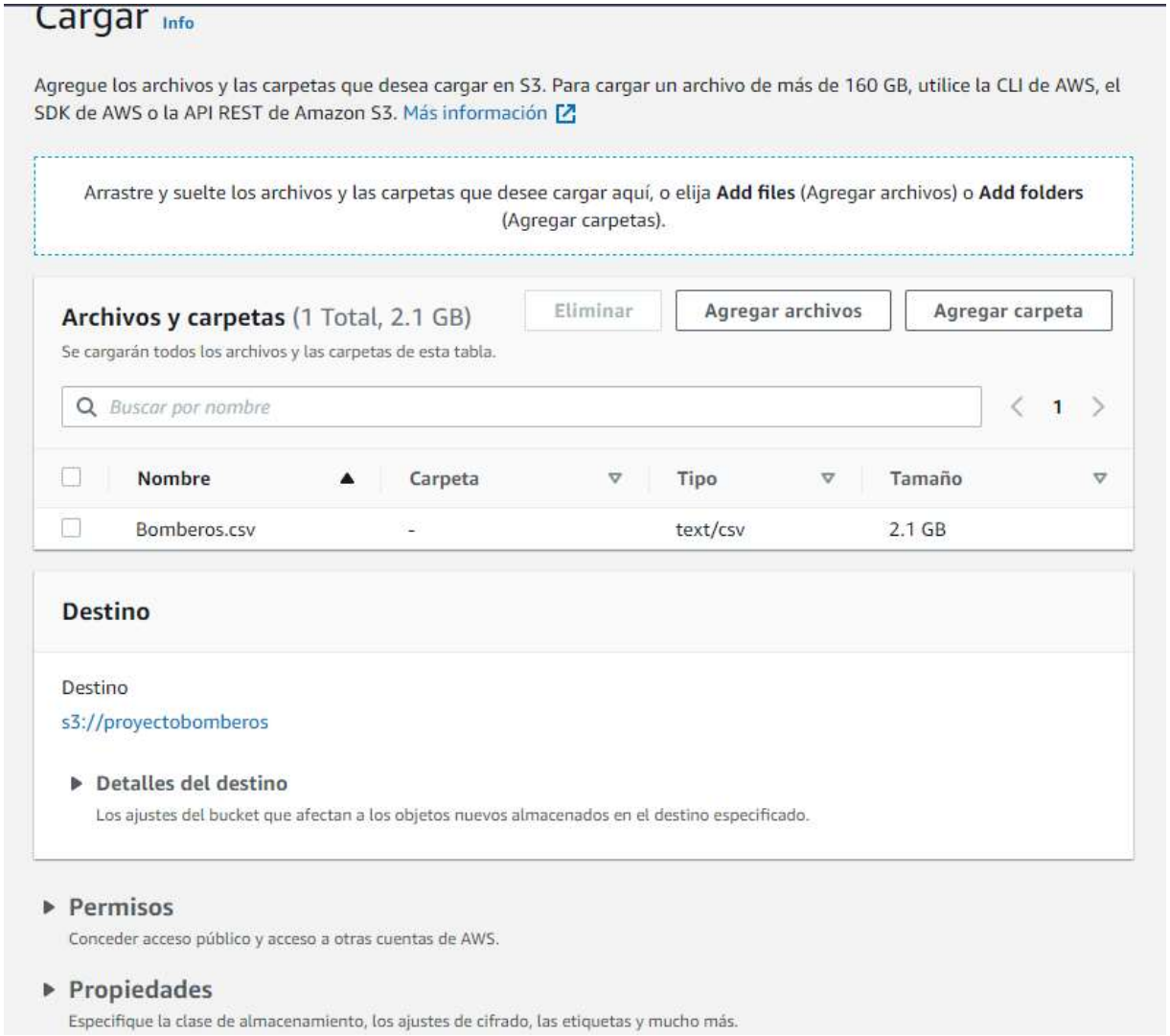
Realización del proyecto

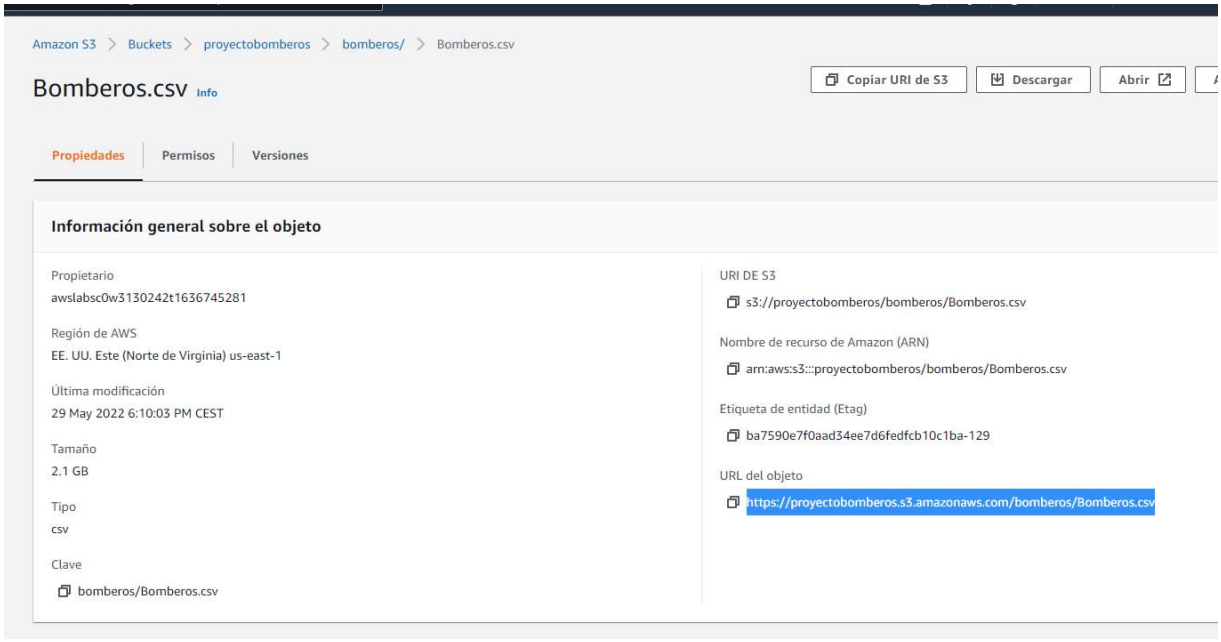
Carga y Almacenamiento:

He creado un **bucket en amazon S3**, para almacenar el csv, leerlo desde Spark, realizar la limpieza o transformaciones, de nuevo volver a almacenar los resultados en AWS S3, y leerlo desde PowerBI para realizar un cuadro o visualización.

Creación de Bucket (AWS S3):

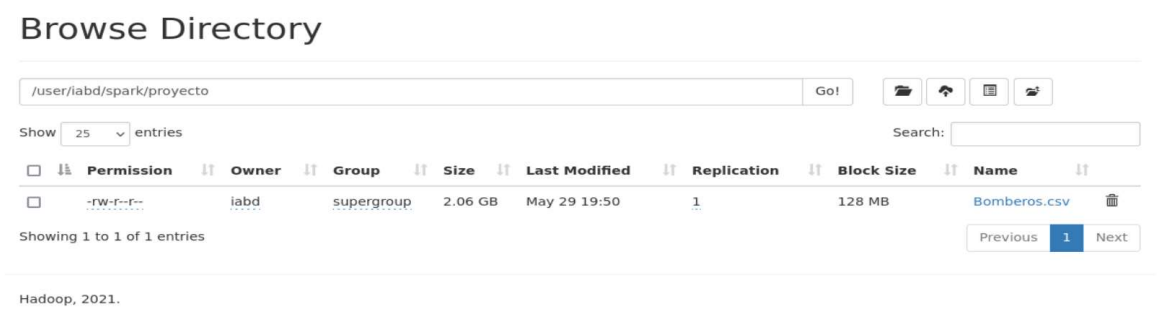






Otra opción podría ser almacenar los datos **HDFS**, también realice varias pruebas por si tenía algún problema con S3.

hdfs dfs -mkdir /spark/proyecto
hdfs dfs -mkdir /spark/proyecto/Resultado
hdfs dfs -put Bomberos.csv /user/iabd/spark/proyecto



Transformaciones y Limpieza de datos (Spark):

Lanzamos el Lab para usar el S3 (donde tenemos el CSV con los datos)
https://awsacademy.instructure.com/courses/9086/modules/items/846890

Configuramos las credenciales, para poder acceder…
Cargamos los datos con Pyspark, examinamos los datos para más tarde elaborar un esquema, y cargar los tipos de datos correctamente.

```
In [2]: #Esquema Cabeceras Español
from pyspark.sql.types import StructType, StructField, StringType, IntegerType, DoubleType, FloatType, DateType, Time
esquema = StructType([
    StructField("NumeroLlamada", IntegerType(), False),
    StructField("IDUnidad", StringType(), False),
    StructField("NumeroIncidente", IntegerType(), False),
    StructField("TipoLlamada", StringType(), False),
    StructField("FechaLlamada", StringType(), True),
    StructField("WatchDate", StringType(), True),
    StructField("ReceivedDtTm", StringType(), True),
    StructField("Entry DtTm", StringType(), True),
    StructField("Dispatch DtTm", StringType(), True),
    StructField("Response DtTm", StringType(), True),
    StructField("OnSceneDtTm", StringType(), True),
    StructField("TransportDtTm", StringType(), True),
    StructField("HospitalDtTm", StringType(), True),
    StructField("CallFinalDisposition", StringType(), True),
    StructField("AvailableDtTm", StringType(), True),
    StructField("Direccion", StringType(), False),
    StructField("Ciudad", StringType(), False),
    StructField("CP", IntegerType(), False),
    StructField("Battalion", StringType(), False),
    StructField("AreaEstacion", IntegerType(), False),
    StructField("Box", IntegerType(), False),
    StructField("PrioridadOriginal", StringType(), False),
    StructField("Prioridad", StringType(), False),
    StructField("PrioridadFinal", StringType(), False),
    StructField("UnidadALS", BooleanType(), False),
    StructField("GrupoTipoLlamada", StringType(), False),
    StructField("NumeroAlarmas", IntegerType(), False),
    StructField("TipoUnidad", StringType(), False),
    StructField("Unit sequence in call dispatch", IntegerType(), False),
    StructField("DistritoPrevencionIncendios", IntegerType(), False),
    StructField("DistritoSupervisor", IntegerType(), False),
    StructField("Vecindario", StringType(), True),
    StructField("IDfila", StringType(), False),
    StructField("Ubicación", StringType(), False),
    StructField("Retraso", FloatType(), True)
])
```

Ya que tratamos de evaluar si los bomberos de San Francisco son "lentos", vamos a establecer un tiempo que si al superarlo podríamos decir que son lentos… creo que 10 minutos sería adecuado…

Antes de proceder con la limpieza de los datos voy a realizar 2 comprobaciones o aproximaciones, con una simple regla de 3, obtenemos el total de las actuaciones, obtenemos las actuaciones que han tardado mas de 10 minutos y obtenemos que el 75% de las actuaciones han tardado mas de 10 minutos.

Por lo que una primera aproximación con estos datos, obtendríamos como resultado que los bomberos de San Francisco son "lentos".

```
In [12]: #Prueba para la presentacion
total = df.select("Retraso").where(col("Retraso") > 0).count()

In [13]: #Prueba para la presentacion
mayor10 = df.select("Retraso").where(col("Retraso") > 10).count()

In [14]: #Prueba para la presentacion
(mayor10 * 100) / total

Out[14]: 73.80061199401672
```

Comprobamos la cantidad de Actuaciones distintas, filtramos por las actuaciones que han realizado los bomberos y volvemos a comprobar la cantidad de actuaciones.

```
In [7]: #Antes de Limpiar los datos vamos a comprobar cuantas actuaciones DISTINTAS se han realizado
        df.select("NumeroLlamada").distinct().count()

Out[7]: 2693286
```

```
In [11]: #Nos quedamos con las llamadas a las que han acudido los Bomberos
         dfBom = df.select("*").where(df.CallFinalDisposition=='Fire')
         dfBom.select("NumeroLlamada").distinct().count()

Out[11]: 372897
```

Los datos a tratar se han reducido considerablemente.

Se realiza una primera limpieza eliminando las columnas que creemos que no son necesarias para nuestro proyecto.

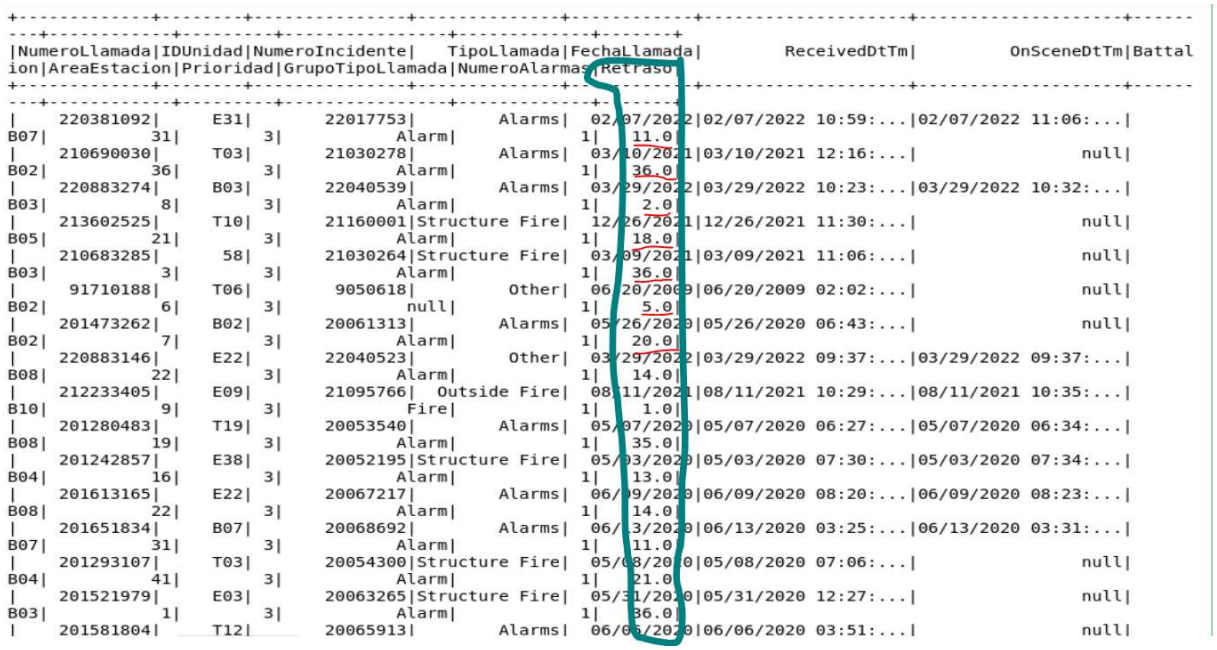
```
#Eliminamos columnas no necesarias
#Antes de esto eliminar estas columnas hemos comprobado:
"""from pyspark.sql.functions import col
dftipollamada = dfRespuestaporBomberos.select("PrioridadOriginal","Prioridad", "PrioridadFinal").where(dfRespuestapor
dftipollamada.show(35, truncate=False)"""
dfLimpio = dfBom.drop("IDfila", "Ciudad", "PrioridadOriginal","Prioridad", "CallFinalDisposition")
dfLimpio = dfLimpio.drop("WatchDate", "Entry DtTm", "Dispatch DtTm", "Response DtTm", "AvailableDtTm", "TransportDtTm
dfLimpio = dfLimpio.drop("DistritoPrevencionIncendios", "DistritoSupervisor", "Vecindario" , "Direccion", "CP", "Ubic
dfLimpio = dfLimpio.drop("TipoUnidad", "Unit sequence in call dispatch", "Box")
dfLimpio = dfLimpio.withColumnRenamed("PrioridadFinal", "Prioridad")
dfLimpio.show()
#dfLimpio.count()
```

Ahora tenemos los datos que creemos que son más importantes estructurados…

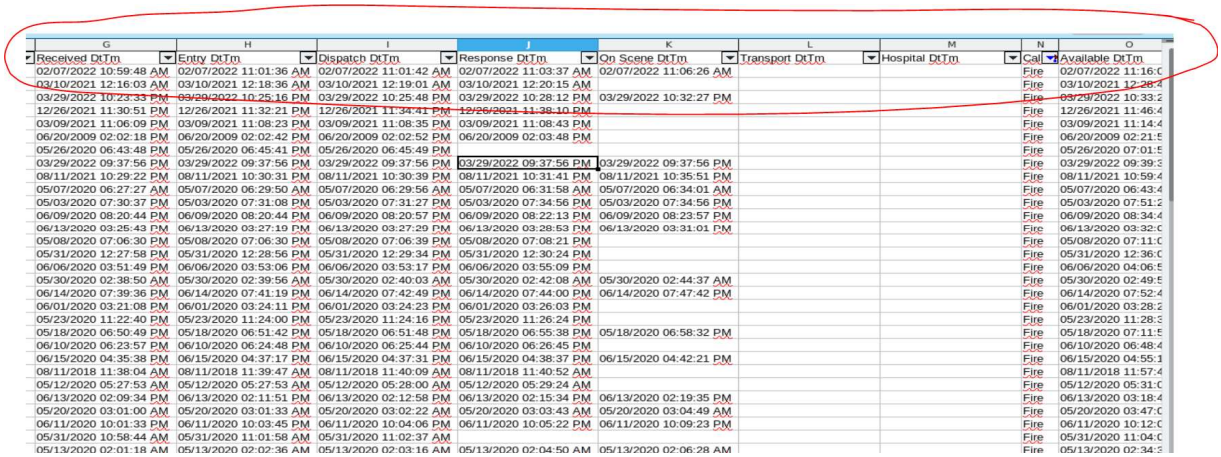
```
In [13]: dfLimpio.printSchema()

root
 |-- NumeroLlamada: integer (nullable = true)
 |-- IDUnidad: string (nullable = true)
 |-- NumeroIncidente: integer (nullable = true)
 |-- TipoLlamada: string (nullable = true)
 |-- FechaLlamada: string (nullable = true)
 |-- ReceivedDtTm: string (nullable = true)
 |-- OnSceneDtTm: string (nullable = true)
 |-- Battalion: string (nullable = true)
 |-- AreaEstacion: integer (nullable = true)
 |-- Prioridad: string (nullable = true)
 |-- GrupoTipoLlamada: string (nullable = true)
 |-- NumeroAlarmas: integer (nullable = true)
 |-- Retraso: float (nullable = true)
```

Voy a centrarme en tratar de recalcular el tiempo o Retraso… al final del CSV hay un campo llamado Analysis Neighborhoods -> que he renombrado con el nombre **Retraso**



NumeroLlamada	IDUnidad	NumeroIncidente	TipoLlamada	FechaLlamada	ReceivedDtTm	OnSceneDtTm	Battalion	AreaEstacion	Prioridad	GrupoTipoLlamada	NumeroAlarmas	Retraso
220381092	E31	22017753	Alarms	02/07/2022	02/07/2022 10:59:...	02/07/2022 11:06:...	B07	31	3	Alarm	1	11.0
210690030	T03	21030278	Alarms	03/10/2021	03/10/2021 12:16:...	null	B02	36	3	Alarm	1	36.0
220883274	B03	22040539	Alarms	03/29/2022	03/29/2022 10:23:...	03/29/2022 10:32:...	B03	8	3	Alarm	1	2.0
213602525	T10	21160001	Structure Fire	12/26/2021	12/26/2021 11:30:...	null	B05	21	3	Alarm	1	18.0
210683285	58	21030264	Structure Fire	03/09/2021	03/09/2021 11:06:...	null	B03	3	3	Alarm	1	36.0
91710188	T06	9050618	Other	06/20/2009	06/20/2009 02:02:...	null	B02	6	3	null	1	5.0
201473262	B02	20061313	Alarms	05/26/2020	05/26/2020 06:43:...	null	B02	7	3	Alarm	1	20.0
220883146	E22	22040523	Other	03/29/2022	03/29/2022 09:37:...	03/29/2022 09:37:...	B08	22	3	Alarm	1	14.0
212233405	E09	21095766	Outside Fire	08/11/2021	08/11/2021 10:29:...	08/11/2021 10:35:...	B10	9	3	Fire	1	1.0
201280483	T19	20053540	Alarms	05/07/2020	05/07/2020 06:27:...	05/07/2020 06:34:...	B08	19	3	Alarm	1	35.0
201242857	E38	20052195	Structure Fire	05/03/2020	05/03/2020 07:30:...	05/03/2020 07:34:...	B04	16	3	Alarm	1	13.0
201613165	E22	20067217	Alarms	06/09/2020	06/09/2020 08:20:...	06/09/2020 08:23:...	B08	22	3	Alarm	1	14.0
201651834	B07	20068692	Alarms	06/13/2020	06/13/2020 03:25:...	06/13/2020 03:31:...	B07	31	3	Alarm	1	11.0
201293107	T03	20054300	Structure Fire	05/08/2020	05/08/2020 07:06:...	null	B04	41	3	Alarm	1	21.0
201521979	E03	20063265	Structure Fire	05/31/2020	05/31/2020 12:27:...	null	B03	1	3	Alarm	1	36.0
201581804	T12	20065913	Alarms	06/06/2020	06/06/2020 03:51:...	null						



Received DtTm	Entry DtTm	Dispatch DtTm	Response DtTm	On Scene DtTm	Transport DtTm	Hospital DtTm	Cal	Available DtTm
02/07/2022 10:59:48 AM	02/07/2022 11:01:36 AM	02/07/2022 11:01:42 AM	02/07/2022 11:03:37 AM	02/07/2022 11:06:26 AM			Fire	02/07/2022 11:16:0
03/10/2021 12:16:03 AM	03/10/2021 12:18:36 AM	03/10/2021 12:19:01 AM	03/10/2021 12:20:15 AM				Fire	03/10/2021 12:28:4
03/29/2022 10:23:33 PM	03/29/2022 10:25:16 PM	03/29/2022 10:25:48 PM	03/29/2022 10:28:12 PM	03/29/2022 10:32:27 PM			Fire	03/29/2022 10:33:2
12/26/2021 11:30:51 PM	12/26/2021 11:32:21 PM	12/26/2021 11:34:41 PM	12/26/2021 11:38:10 PM				Fire	12/26/2021 11:46:4
03/09/2021 11:06:09 PM	03/09/2021 11:08:23 PM	03/09/2021 11:08:35 PM	03/09/2021 11:08:43 PM				Fire	03/09/2021 11:14:4
06/20/2009 02:02:18 PM	06/20/2009 02:02:42 PM	06/20/2009 02:02:52 PM	06/20/2009 02:03:48 PM				Fire	06/20/2009 02:21:5
05/26/2020 06:43:48 PM	05/26/2020 06:45:41 PM	05/26/2020 06:45:49 PM					Fire	05/26/2020 07:01:5
03/29/2022 09:37:56 PM	03/29/2022 09:37:56 PM	03/29/2022 09:37:56 PM	03/29/2022 09:37:56 PM	03/29/2022 09:37:56 PM			Fire	03/29/2022 09:39:3
08/11/2021 10:29:22 PM	08/11/2021 10:30:31 PM	08/11/2021 10:30:39 PM	08/11/2021 10:31:41 PM	08/11/2021 10:35:51 PM			Fire	08/11/2021 10:59:4
05/07/2020 06:27:27 AM	05/07/2020 06:29:50 AM	05/07/2020 06:29:56 AM	05/07/2020 06:31:58 AM	05/07/2020 06:34:01 AM			Fire	05/07/2020 06:43:4
05/03/2020 07:30:37 PM	05/03/2020 07:31:08 PM	05/03/2020 07:31:27 PM	05/03/2020 07:34:56 PM	05/03/2020 07:34:56 PM			Fire	05/03/2020 07:51:2
06/09/2020 08:20:44 PM	06/09/2020 08:20:44 PM	06/09/2020 08:20:57 PM	06/09/2020 08:22:13 PM	06/09/2020 08:23:57 PM			Fire	06/09/2020 08:34:4
06/13/2020 03:25:43 PM	06/13/2020 03:27:19 PM	06/13/2020 03:27:29 PM	06/13/2020 03:28:53 PM	06/13/2020 03:31:01 PM			Fire	06/13/2020 03:32:0
05/08/2020 07:06:30 PM	05/08/2020 07:06:30 PM	05/08/2020 07:06:39 PM	05/08/2020 07:08:21 PM				Fire	05/08/2020 07:11:0
05/31/2020 12:27:58 PM	05/31/2020 12:28:56 PM	05/31/2020 12:29:34 PM	05/31/2020 12:30:24 PM				Fire	05/31/2020 12:36:0
06/06/2020 03:51:49 PM	06/06/2020 03:53:06 PM	06/06/2020 03:53:17 PM	06/06/2020 03:55:09 PM				Fire	06/06/2020 04:06:5
05/30/2020 02:38:50 AM	05/30/2020 02:39:56 AM	05/30/2020 02:40:03 AM	05/30/2020 02:42:08 AM	05/30/2020 02:44:37 AM			Fire	05/30/2020 02:49:5
06/14/2020 07:39:36 PM	06/14/2020 07:41:19 PM	06/14/2020 07:42:49 PM	06/14/2020 07:44:00 PM	06/14/2020 07:47:42 PM			Fire	06/14/2020 07:52:4
06/01/2020 03:21:08 PM	06/01/2020 03:24:11 PM	06/01/2020 03:24:23 PM	06/01/2020 03:26:03 PM				Fire	06/01/2020 03:28:2
05/23/2020 11:22:40 PM	05/23/2020 11:24:00 PM	05/23/2020 11:24:16 PM	05/23/2020 11:26:24 PM				Fire	05/23/2020 11:28:3
05/18/2020 06:50:49 PM	05/18/2020 06:51:42 PM	05/18/2020 06:51:48 PM	05/18/2020 06:55:38 PM	05/18/2020 06:58:32 PM			Fire	05/18/2020 07:11:5
06/10/2020 06:23:57 PM	06/10/2020 06:24:48 PM	06/10/2020 06:25:44 PM	06/10/2020 06:26:45 PM				Fire	06/10/2020 06:48:4
06/15/2020 04:35:38 PM	06/15/2020 04:37:17 PM	06/15/2020 04:37:31 PM	06/15/2020 04:38:37 PM	06/15/2020 04:42:21 PM			Fire	06/15/2020 04:55:1
08/11/2018 11:38:04 AM	08/11/2018 11:39:47 AM	08/11/2018 11:40:09 AM	08/11/2018 11:40:52 AM				Fire	08/11/2018 11:57:4
05/12/2020 05:27:53 AM	05/12/2020 05:27:53 AM	05/12/2020 05:28:00 AM	05/12/2020 05:29:24 AM				Fire	05/12/2020 05:31:0
06/13/2020 02:09:34 PM	06/13/2020 02:11:51 PM	06/13/2020 02:12:58 PM	06/13/2020 02:15:34 PM	06/13/2020 02:19:35 PM			Fire	06/13/2020 03:18:4
05/20/2020 03:01:00 AM	05/20/2020 03:01:33 AM	05/20/2020 03:02:22 AM	05/20/2020 03:03:43 AM	05/20/2020 03:04:49 AM			Fire	05/20/2020 03:47:0
06/11/2020 10:01:33 PM	06/11/2020 10:03:45 PM	06/11/2020 10:04:06 PM	06/11/2020 10:05:22 PM	06/11/2020 10:09:23 PM			Fire	06/11/2020 10:12:0
05/31/2020 10:58:44 AM	05/31/2020 11:01:58 AM	05/31/2020 11:02:37 AM					Fire	05/31/2020 11:04:0
05/13/2020 02:01:18 AM	05/13/2020 02:02:36 AM	05/13/2020 02:03:16 AM	05/13/2020 02:04:50 AM	05/13/2020 02:06:28 AM			Fire	05/13/2020 02:34:5

He tratado averiguar cómo se ha realizado el cálculo del Retraso.

He probado a restar varios campos de fechas entre sí, a sumarlos y dividirlos… pero no consigo obtener el resultado que ellos han obtenido.

Por ejemplo, he restado el campo OnSCene, (que es cuando una unidad llegar al lugar del accidente) con el campo del Aviso Recibido, para ello transformamos los campos a datatime.

También vemos que hay filas erróneas, (*como puede ser que una unidad haya llegado 2 horas antes de que se dé el aviso… al lugar del accidente…*)

```
In [85]: #Pruebas
         dfHoras.select("*").where(col("NumeroIncidente") == 21065237).head(59)

Out[85]: [Row(NumeroLlamada=211523622, IDUnidad='RS2', NumeroIncidente=21065237, TipoLlamada='Traffic Collision', ReceivedDtTm='06/01/2021 11:23:42 PM', OnSceneDtTm='06/01/2021 11:33:34 PM', CallFinalDisposition='Fire', Battalion='B10', AreaEstacion=9, Prioridad='3', GrupoTipoLlamada='Potentially Life-Threatening', NumeroAlarmas=1, Retraso=1.0, DtTmRecibido=datetime.datetime(2021, 6, 1, 23, 23, 42), DtTmEnLugar=datetime.datetime(2021, 6, 1, 23, 33, 34), DiffInSeconds=592, DiffInMinutes=10.0),
          Row(NumeroLlamada=211523622, IDUnidad='T09', NumeroIncidente=21065237, TipoLlamada='Traffic Collision', ReceivedDtTm='06/01/2021 11:23:42 PM', OnSceneDtTm='06/01/2021 11:30:16 PM', CallFinalDisposition='Fire', Battalion='B10', AreaEstacion=9, Prioridad='3', GrupoTipoLlamada='Potentially Life-Threatening', NumeroAlarmas=1, Retraso=1.0, DtTmRecibido=datetime.datetime(2021, 6, 1, 23, 23, 42), DtTmEnLugar=datetime.datetime(2021, 6, 1, 23, 30, 16), DiffInSeconds=394, DiffInMinutes=7.0),
          Row(NumeroLlamada=211523622, IDUnidad='73', NumeroIncidente=21065237, TipoLlamada='Traffic Collision', ReceivedDtTm='06/01/2021 11:23:42 PM', OnSceneDtTm='06/01/2021 11:32:28 PM', CallFinalDisposition='Fire', Battalion='B10', AreaEstacion=9, Prioridad='3', GrupoTipoLlamada='Potentially Life-Threatening', NumeroAlarmas=1, Retraso=1.0, DtTmRecibido=datetime.datetime(2021, 6, 1, 23, 23, 42), DtTmEnLugar=datetime.datetime(2021, 6, 1, 23, 32, 28), DiffInSeconds=526, DiffInMinutes=9.0),
          Row(NumeroLlamada=211523622, IDUnidad='B10', NumeroIncidente=21065237, TipoLlamada='Traffic Collision', ReceivedDtTm='06/01/2021 11:23:42 PM', OnSceneDtTm='06/01/2021 09:31:00 PM', CallFinalDisposition='Fire', Battalion='B10', AreaEstacion=9, Prioridad='3', GrupoTipoLlamada='Potentially Life-Threatening', NumeroAlarmas=1, Retraso=1.0, DtTmRecibido=datetime.datetime(2021, 6, 1, 23, 23, 42), DtTmEnLugar=datetime.datetime(2021, 6, 1, 21, 31), DiffInSeconds=-6762, DiffInMinutes=-113.0),
          Row(NumeroLlamada=211523622, IDUnidad='RC3', NumeroIncidente=21065237, TipoLlamada='Traffic Collision', ReceivedDtTm='06/01/2021 11:23:42 PM', OnSceneDtTm='06/01/2021 11:36:38 PM', CallFinalDisposition='Fire', Battalion='B10', AreaEstacion=9, Prioridad='3', GrupoTipoLlamada='Potentially Life-Threatening', NumeroAlarmas=1, Retraso=1.0, DtTmRecibido=datetime.datetime(2021, 6, 1, 23, 23, 42), DtTmEnLugar=datetime.datetime(2021, 6, 1, 23, 36, 38), DiffInSeconds=776, DiffInMinutes=13.0),
          Row(NumeroLlamada=211523622, IDUnidad='67', NumeroIncidente=21065237, TipoLlamada='Traffic Collision', ReceivedDtTm='06/01/2021 11:23:42 PM', OnSceneDtTm='06/01/2021 11:33:56 PM', CallFinalDisposition='Fire', Battalion='B10', AreaEstacion=9, Prioridad='3', GrupoTipoLlamada='Potentially Life-Threatening', NumeroAlarmas=1, Retraso=1.0, DtTmRecibido=datetime.datetime(2021, 6, 1, 23, 23, 42), DtTmEnLugar=datetime.datetime(2021, 6, 1, 23, 33, 56), DiffInSeconds=614, DiffInMinutes=10.0),
          Row(NumeroLlamada=211523622, IDUnidad='E09', NumeroIncidente=21065237, TipoLlamada='Traffic Collision', ReceivedDtTm='06/01/2021 11:23:42 PM', OnSceneDtTm='06/01/2021 11:29:51 PM', CallFinalDisposition='Fire', Battalion='B10', Area
```

En conclusión, no he conseguido averiguar cómo se ha calculado esta columna, con el fin de mejorar los tiempos…

Tras realizar estas comprobaciones, vamos a volver a realizar una limpieza eliminando los campos que no necesitamos y transformamos el campo FechaLLamada en datatime renombrando a Fecha, para utilizarlo más tarde.

Nos quedamos con los siguientes datos, para realizar el cuadro o visualizaciones:

```
In [100]: dfFinal = dfHoras.select("NumeroIncidente","Fecha","TipoLlamada", "IDUnidad", "Battalion", "AreaEstacion", "Retraso")
          dfFinal.show()
```

NumeroIncidente	Fecha	TipoLlamada	IDUnidad	Battalion	AreaEstacion	Retraso
22017753	2022-02-07 00:00:00	Alarms	E31	B07	31	11.0
22040539	2022-03-29 00:00:00	Alarms	B03	B03	8	2.0
22040523	2022-03-29 00:00:00	Other	E22	B08	22	14.0
21095766	2021-08-11 00:00:00	Outside Fire	E09	B10	9	1.0
20053540	2020-05-07 00:00:00	Alarms	T19	B08	19	35.0
20052195	2020-05-03 00:00:00	Structure Fire	E38	B04	16	13.0
20067217	2020-06-09 00:00:00	Alarms	E22	B08	22	14.0
20068692	2020-06-13 00:00:00	Alarms	B07	B07	31	11.0
20062727	2020-05-30 00:00:00	Alarms	B04	B04	16	13.0
20069110	2020-06-14 00:00:00	Alarms	B09	B08	33	16.0
20058051	2020-05-18 00:00:00	Alarms	T11	B06	24	38.0
20069414	2020-06-15 00:00:00	Structure Fire	B06	B06	11	22.0
20068676	2020-06-13 00:00:00	Water Rescue	T10	B04	51	27.0
20058524	2020-05-20 00:00:00	Structure Fire	T03	B02	3	36.0
20068088	2020-06-11 00:00:00	Structure Fire	B08	B08	23	35.0
20055770	2020-05-13 00:00:00	Structure Fire	T06	B05	6	9.0
20077054	2020-07-04 00:00:00	Outside Fire	B03	B06	7	20.0
20059881	2020-05-23 00:00:00	Alarms	E03	B01	41	21.0
20079492	2020-07-10 00:00:00	Structure Fire	RS1	B02	1	34.0
20073872	2020-06-27 00:00:00	Alarms	T01	B03	1	34.0

only showing top 20 rows

Si realizamos la misma comprobación vemos que obtenemos un resultado similar

```
In [85]: dfHoras.select("Retraso").where(col("Retraso") > 0).count()
Out[85]: 580603
```

```
In [86]: dfHoras.select("Retraso").where(col("Retraso") > 10).count()
Out[86]: 415849
```

```
In [73]: (415849 * 100) / 580603
Out[73]: 71.62363956093922
```

Persistimos varios csv con distintas transformaciones a S3

```
#******************************** Ejecutar ********************************
from pyspark.sql.functions import min, max, avg, sum
dftotalAlarmas = dfHoras.select(sum("NumeroAlarmas").alias("TotalAlarmas"),avg("Retraso").alias("MediaRetraso") ,min(
dftotalAlarmas.show()
```

```
+------------+-----------------+----------+----------+
|TotalAlarmas|     MediaRetraso|MinRetraso|MaxRetraso|
+------------+-----------------+----------+----------+
|      584935|21.46079334760585|       1.0|      41.0|
+------------+-----------------+----------+----------+
```

```
#******************************** Ejecutar ********************************
#S3 -> Subimos los datos del maximo minimo y media:
dftotalAlarmas.coalesce(1).write.mode("overwrite").option("header",True).csv("s3a://proyectobomberos/Resultados/Total
```

```
In [37]: #******************************** Ejecutar ********************************
dftiposLlamada = dfHoras.select("TipoLlamada").groupby('TipoLlamada').count().orderBy("count",ascending=Falsе
dftiposLlamada.show(10)
```

```
+--------------------+------+
|         TipoLlamada| count|
+--------------------+------+
|              Alarms|248318|
|      Structure Fire|149350|
|Citizen Assist / ...| 36874|
|        Outside Fire| 33917|
|               Other| 27661|
|    Medical Incident| 22243|
|Gas Leak (Natural...| 13498|
|   Electrical Hazard| 11777|
|Elevator / Escala...|  8497|
|        Vehicle Fire|  6964|
+--------------------+------+
only showing top 10 rows
```

```
In [ ]: #******************************** Ejecutar ********************************
#S3 -> Subimos los tipos de llamada
dftiposLlamada.coalesce(1).write.mode("overwrite").option("header",True).csv("s3a://proyectobomberos/Resultac
```

```
#******************************** Ejecutar ********************************
dfFinal.show(10)
```

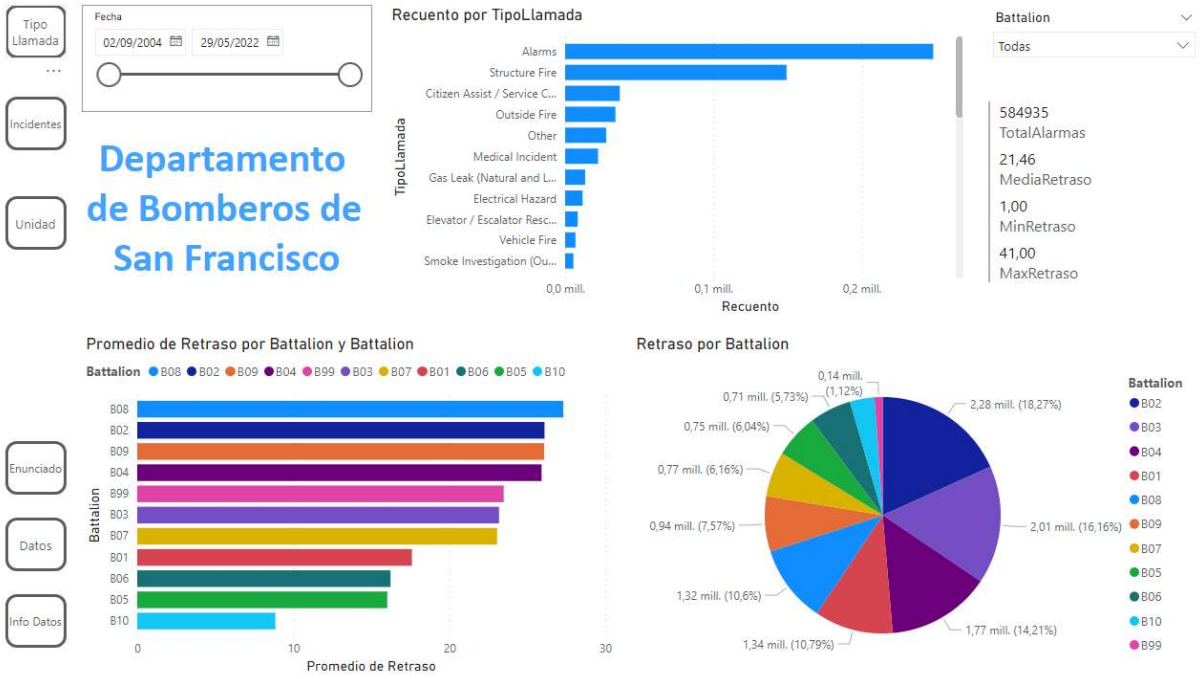
```
+---------------+-------------------+--------------+--------+---------+------------+-------+
|NumeroIncidente|              Fecha|   TipoLlamada|IDUnidad|Battalion|AreaEstacion|Retraso|
+---------------+-------------------+--------------+--------+---------+------------+-------+
|       22017753|2022-02-07 00:00:00|        Alarms|     E31|      B07|          31|   11.0|
|       22040539|2022-03-29 00:00:00|        Alarms|     B03|      B03|           8|    2.0|
|       22040523|2022-03-29 00:00:00|         Other|     E22|      B08|          22|   14.0|
|       21095766|2021-08-11 00:00:00|  Outside Fire|     E09|      B10|           9|    1.0|
|       20053540|2020-05-07 00:00:00|        Alarms|     T19|      B08|          19|   35.0|
|       20052195|2020-05-03 00:00:00|Structure Fire|     E38|      B04|          16|   13.0|
|       20067217|2020-06-09 00:00:00|        Alarms|     E22|      B08|          22|   14.0|
|       20068692|2020-06-13 00:00:00|        Alarms|     B07|      B07|          31|   11.0|
|       20062727|2020-05-30 00:00:00|        Alarms|     B04|      B04|          16|   13.0|
|       20069110|2020-06-14 00:00:00|        Alarms|     B09|      B08|          33|   16.0|
+---------------+-------------------+--------------+--------+---------+------------+-------+
only showing top 10 rows
```

```
#******************************** Ejecutar ********************************
#S3 -> El Resultado Final
dfFinal.coalesce(1).write.mode("overwrite").option("header",True).csv("s3a://proyectobomberos/Resultados/dfFinal")
```

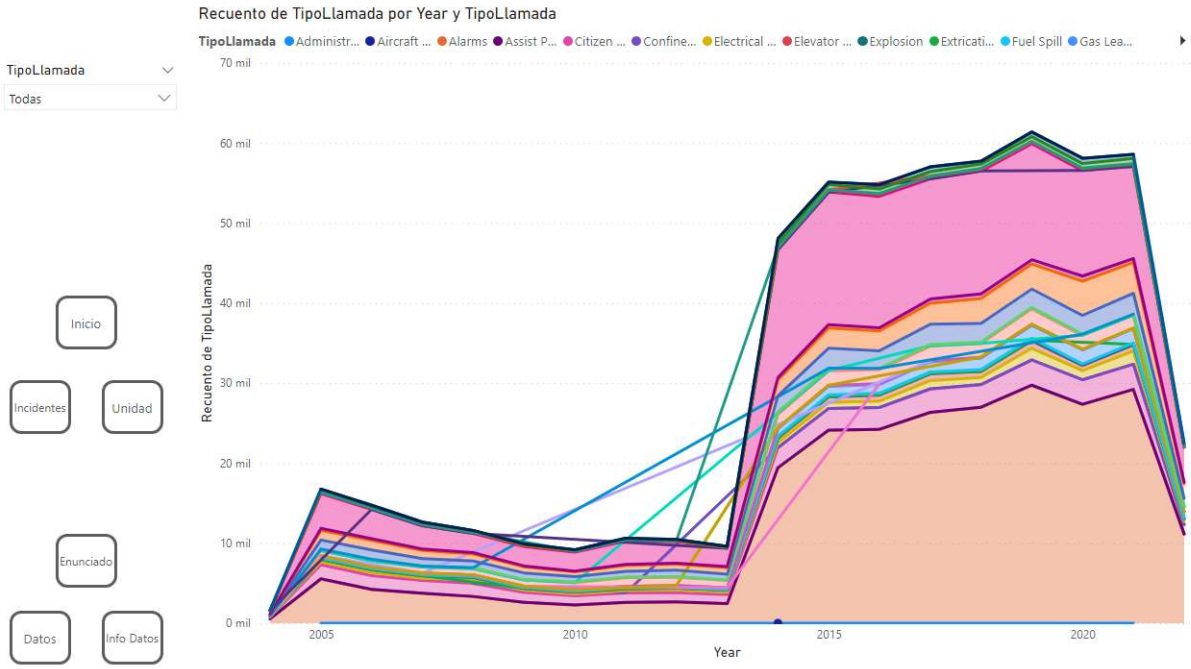
Visualizaciones Con PowerBI:

https://app.powerbi.com/reportEmbed?reportId=f424c37c-089c-41c1-bf85-efe7197a079e&autoAuth=true&ctid=759108f9-cdb3-4b04-b976-3e9b3d9ad0be&config=eyJjbHVzdGVyVXJsIjoiaHR0cHM6Ly93YWJpLW5vcnRoLWV1cm9wZS1oLXByaW1hcnktcmVkaXJlY3QuYW5hbHlzaXMud2luZG93cy5uZXQvIn0%3D

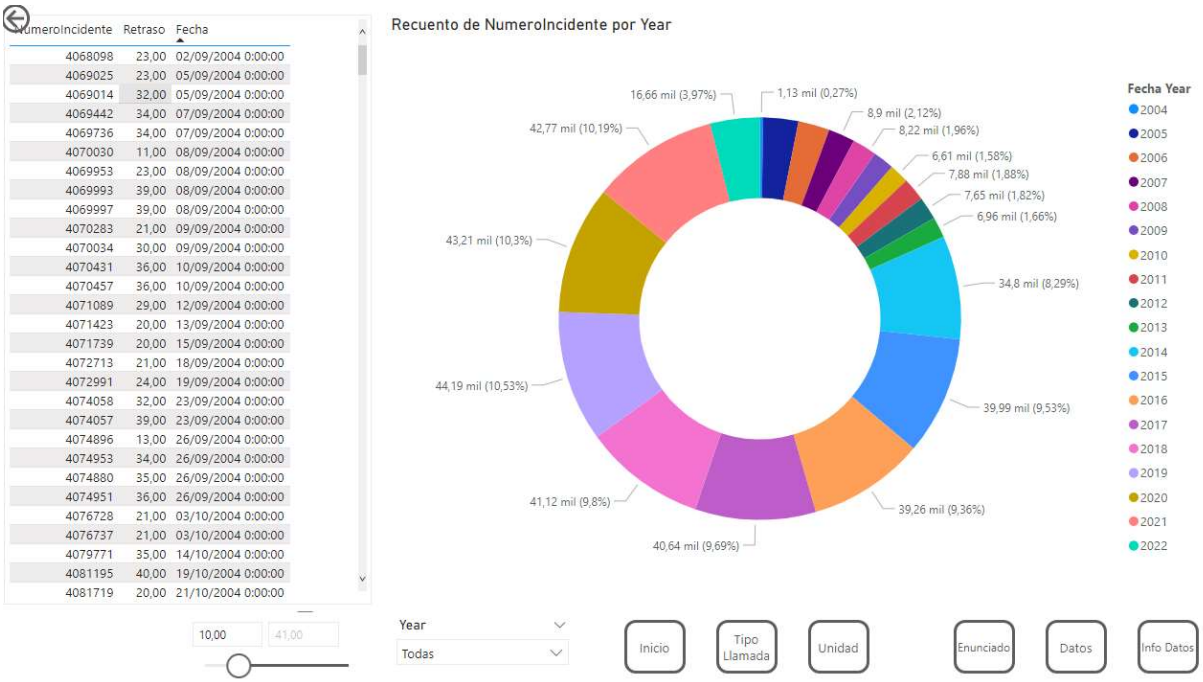
INICIO:



TIPO DE LLAMADA:



INCIDENTES:



TIPO DE UNIDAD:

Se adjunta Código *ProyectoBomberos.ipynb* y Cuadro PowerBI *Proyecto.pbix*