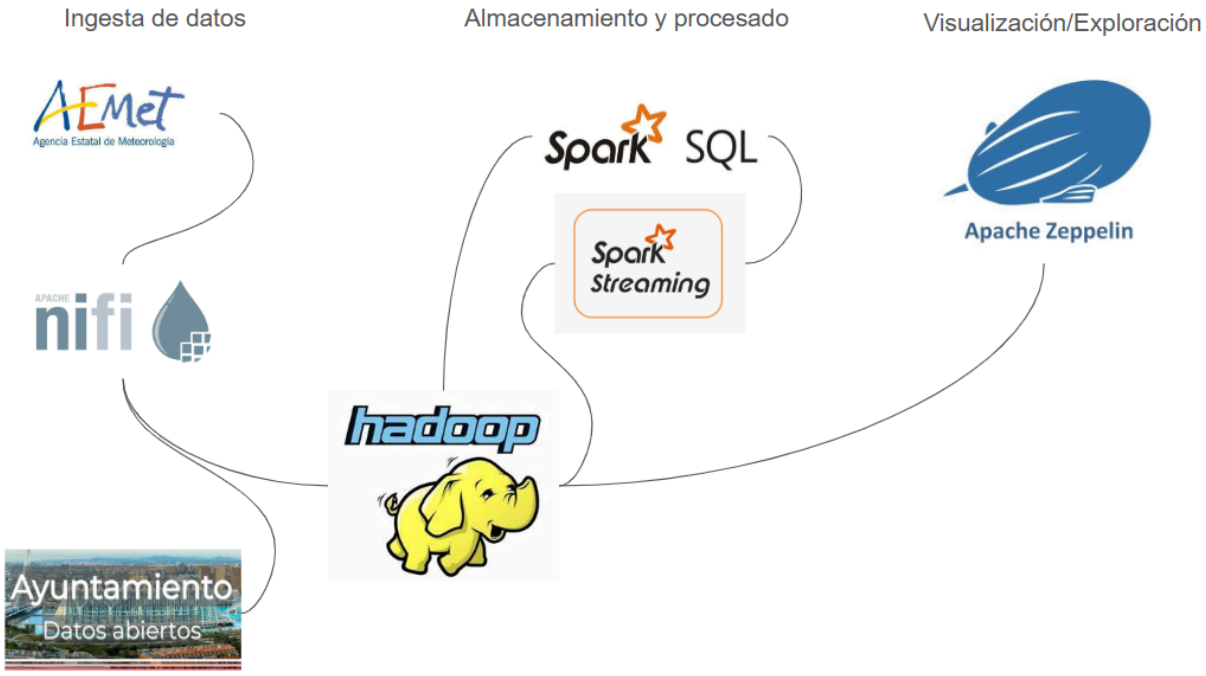
Practica Data lake

En este documento voy a explicar en detalle cómo he montado una infraestructura de ingesta, procesado y explotación de datos en tiempo real enfocados en estudiar el tráfico de la ciudad de Valencia. En mi caso, la arquitectura del Data lake está fuertemente enfocada en el tiempo “real” y he aprovechado al máximo la frecuencia con la que cada fuente de información envía datos para acercarme lo más posible a ese tiempo real.

La estructura tecnológica y la vinculación con las capas que presenta el Data lake son las siguientes:

Cabe destacar que debido al fuerte enfoque en tiempo real que presenta la infraestructura de datos, no sería estrictamente correcto definirlo como un lago con una arquitectura lambda, más bien como una fusión intermedia entre dicha arquitectura y una arquitectura kappa (como seguramente debe de pasar en muchas implementaciones prácticas en big data). Dicho esto, si que es verdad que encontramos capas bien diferenciadas que los las siguientes que podemos tomar como índice de este documento:

- Ingesta
- Almacenamiento
- Procesado en streaming
- Procesamiento en batch
- Visualización



En lo siguiente de la práctica vamos a detallar cómo cada una de las tecnologías interactúan entre sí en cada una de las secciones.

Ingesta de datos

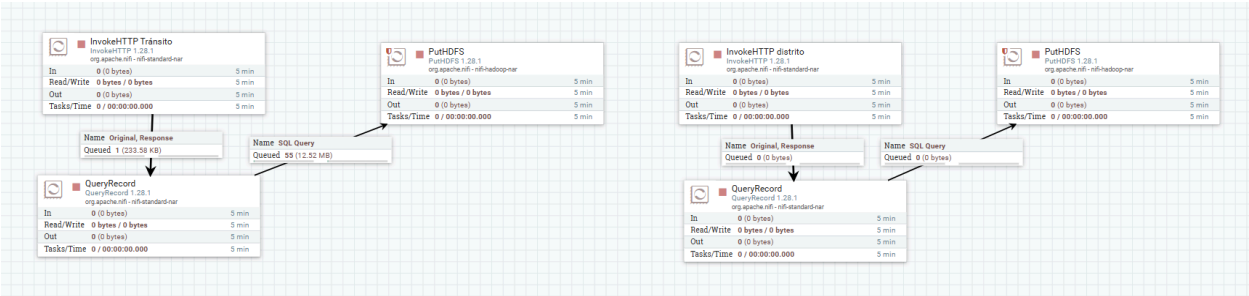
Existen 4 fuentes distintas que alimentan el Data Lake, tres forman parte de los datos abiertos de Valéncia y una es de la AEMET la Agéncia Estatal de Meteorología.

Dentro de los datos abiertos de València he decidido coger, aparte de los datos de tránsito actualizados cada 3 minutos por calle, los datos estáticos de las coordenadas de los 19 distritos de Valencia, y el tráfico del sistema público de bicicletas llamado Valenbisi, que se actualiza cada 10 minutos.

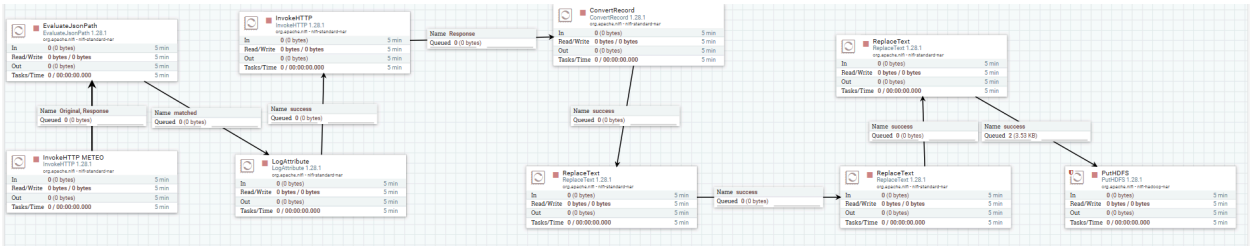
Por otra lado tenemos los datos de la AEMET que consisten en leer los datos de la estación meteorológica del aeropuerto de Valencia que nos dan mucha información meteorológica.

Como vamos a ir viendo en las diferentes secciones, las fuentes no son muy diferentes entre ellas en cuanto a formato se refiere pero, a cambio, tienen un gran potencial complementario y representan diferentes aspectos importantes relacionados o que afectan directamente al tráfico.

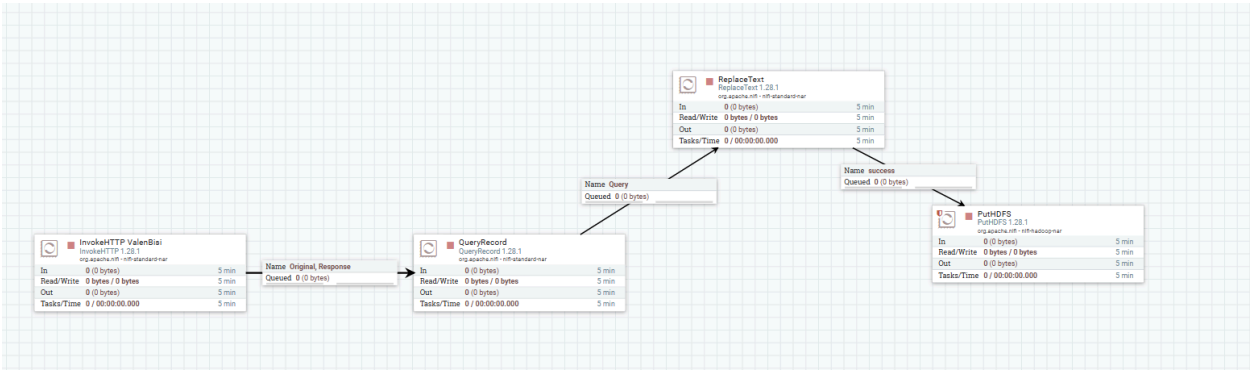
La tecnología que hemos utilizado aquí es Apache Nifi, a continuación se muestran todos los flujos de datos:



Cómo podemos ver, tanto la API del tráfico de Valencia como la de los municipios permite descargar los datos en CSV en tiempo real de los 380 radares. Aquí la única transformación que hago es utilizar los Controllers de Nifi CSVReader y CSVRecordSetWriter dentro de QueryRecord, para que el CSVRecordSetWriter vuelva a escribir en CSV quitando los encabezados de la primera fila en ambos casos y pueda meter los datos directamente el tabl Hive que le corresponde con PutHDFS.



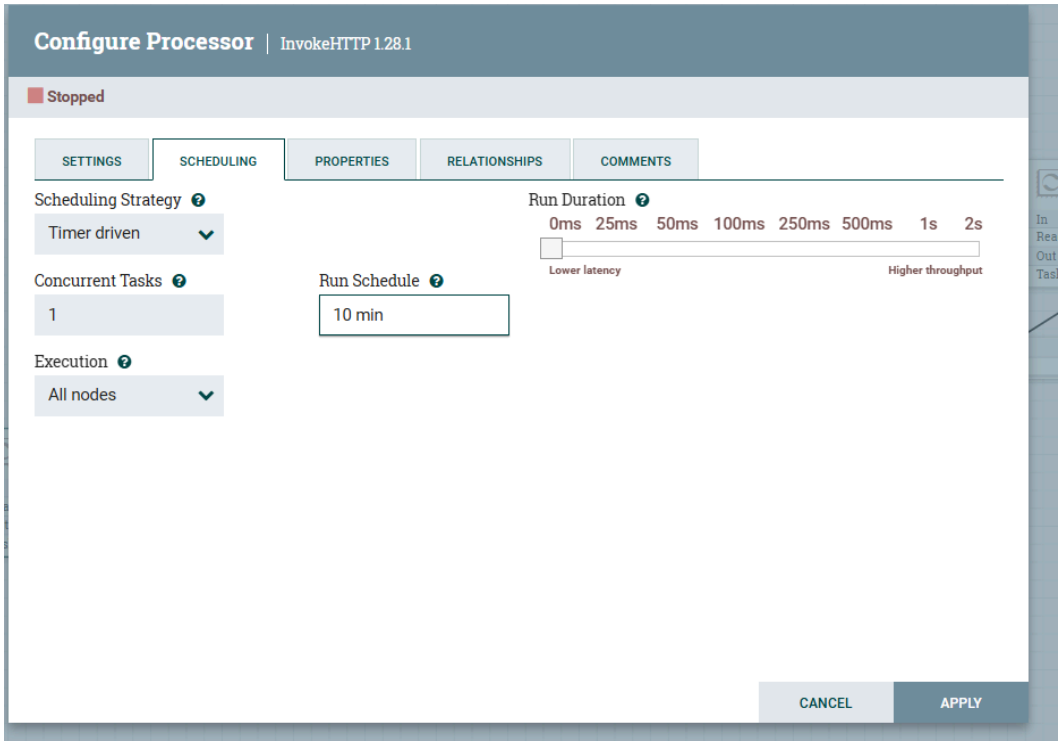
La meteorología requiere más transformaciones antes de poder despejar los datos de la API en sí. Lo que devuelve InvokeHTTP METEO es un JSON en el que el campo “datos” contiene una URL en la cual se genera otro JSON con la información, también en JSON que es el que lee el segundo InvokeHTTP usando como URL el attributo datos_url extraído mediante EvaluateJSONPath, ConvertRecord vuelve un CSV y con los 3 ReplaceText cambiamos la fecha (quitamos timezone, quitamos la T de timezone y cambiamos las barras “/” por “-”) para que tenga un valor válido cómo fecha dentro de la tabla hive en su columna correspondiente tipo TIMESTAMP. Finalmente lo metemos dentro de PutHDFS en la tabla correspondiente.



Para los datos de Valenbisi QueryRecord vuelve a hacer un SELECT * FROM FLOWFILE sin más pero volvemos a quitar la primera fila con el controller CSVSetRecordWriter y ReplaceText cambia otra vez la fecha antes de que PutHDFS los guarde en el directorio de la tabla Hive.

Por último, para acercar lo más posible el comportamiento de la ingesta a tiempo real he tenido en cuenta la frecuencia relativa con la que las APIs envían datos nuevos. Siendo así, y he configurado la frecuencia de los InvokeHTTP en consonancia, por ejemplo, este es el caso de

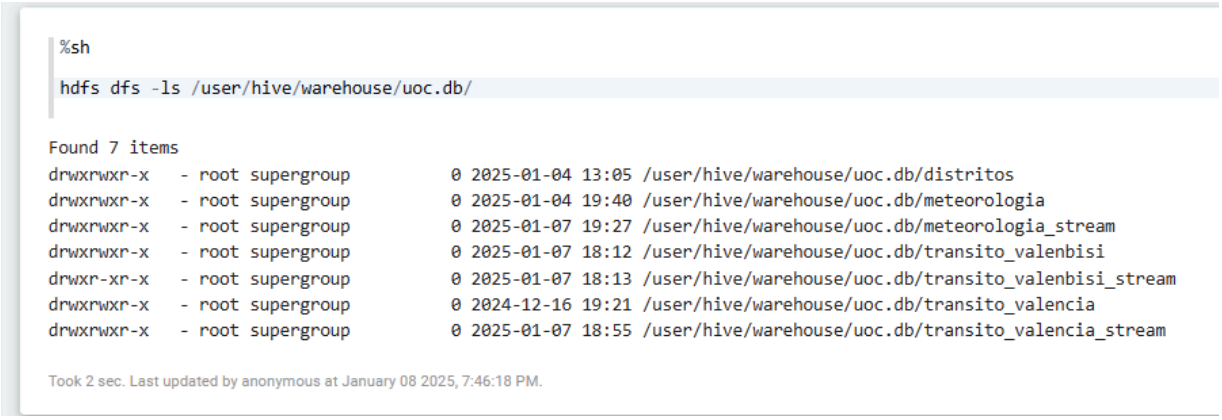
los flujos de Valenbisi que se actualizan cada 10 minutos:



Siguiendo la misma lógica, he puesto 3 min en el InvokeHTTP del tránsito de Valencia y una hora en el de la meteorología ya que se actualiza horariamente. Así evitamos sobrecargar la red de tráfico de datos innecesario y evitar almacenar datos redundantes.

Almacenamiento

Para el almacenamiento he utilizado la tecnología de HDFS de Hadoop, dentro de Hadoop utilizamos las tablas hive que se nutren de los siguientes directorios. Dentro de estos directorios podemos encontrar en cada uno de ellos varios archivos, cada uno de ellos contiene la algunas de las filas y todos comparten el mismo formato CSV para que cuando llamamos a la tabla, aparezcan los registros dentro de estos archivos, esta es una diferencia importante respecto de las tablas SQL tradicionales donde la base de datos es un único bloque transparente para nosotros y el contenido de las tablas no es rastreable como un conjunto de archivos.
A continuación detallo la los directorios:



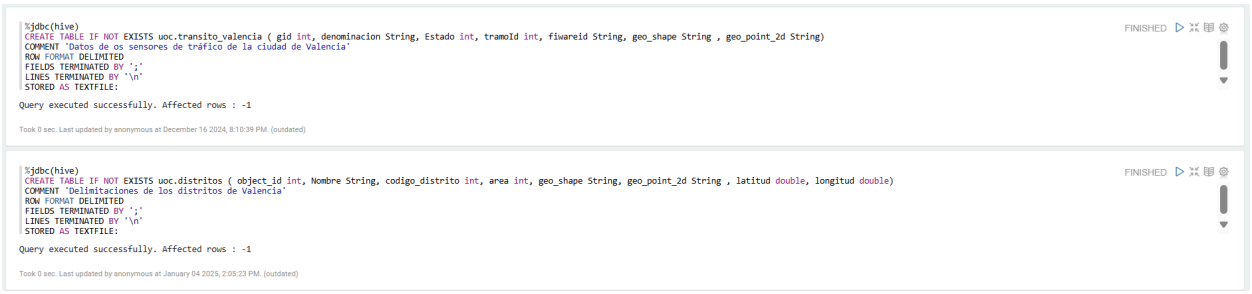
```
%sh

hdfs dfs -ls /user/hive/warehouse/uoc.db/

Found 7 items
drwxrwxr-x   - root supergroup          0 2025-01-04 13:05 /user/hive/warehouse/uoc.db/distritos
drwxrwxr-x   - root supergroup          0 2025-01-04 19:40 /user/hive/warehouse/uoc.db/meteorologia
drwxrwxr-x   - root supergroup          0 2025-01-07 19:27 /user/hive/warehouse/uoc.db/meteorologia_stream
drwxrwxr-x   - root supergroup          0 2025-01-07 18:12 /user/hive/warehouse/uoc.db/transito_valenbisi
drwxr-xr-x   - root supergroup          0 2025-01-07 18:13 /user/hive/warehouse/uoc.db/transito_valenbisi_stream
drwxrwxr-x   - root supergroup          0 2024-12-16 19:21 /user/hive/warehouse/uoc.db/transito_valencia
drwxrwxr-x   - root supergroup          0 2025-01-07 18:55 /user/hive/warehouse/uoc.db/transito_valencia_stream

Took 2 sec. Last updated by anonymous at January 08 2025, 7:46:18 PM.
```

Los datos van avanzando por estos directorios a medida que el ciclo de vida de los datos llega a su punto final donde son almacenados como una serie temporal. Este proceso empieza cuando las tablas que NO tienen la terminación “_stream” reciben los flujos de datos directos desde Nifi. En esas tablas se guardan los datos en crudo con un mínimo preprocesamiento descrito en la ingesta para poder ser interpretados por las tablas hive cuando son guardados como archivos que contienen registros de esas tablas.
Los distritos no cambian, sino no es en el muy largo plazo posiblemente, cosa de la cual si nos enteramos solo tendríamos que borrar la tabla distritos con “TRUNCATE TABLE uoc.distritos” para volver a extraer los nuevos distritos con el flujo de nifi para volverlos a utilizar.

Cuando los datos llegan a las tablas no streaming son leídos inmediatamente ya que estas están siendo monitoreadas por scripts de Spark Structured Streaming y realizan la lectura y procesado de los nuevos registros que aparecen en las tablas, una vez este proceso se ha terminado entonces ya podemos dar por finalizado el ciclo de vida de los datos y estos quedan guardados como una serie temporal dentro de las tablas streaming.
Ahora que he explicado cómo y dónde se van almacenando los datos en HDFS. Puedo comparar con un poco más de profundidad la arquitectura propuesta con las habituales Kappa y Lambda. Cómo podemos ver y respecto de la arquitectura Lambda, la capa de streaming conformada por spark streaming y las tablas “_stream” que llena es una capa que si bien ofrece los datos cerca del tiempo “real”, estos no son eliminados sino que son persistidos en Hadoop, mientras que en una capa streaming de una arquitectura Lambda el contenido de esta se elimina periódicamente cuando la capa batch recupera esos datos. Por otra parte, tampoco podemos hablar de una Kappa al uso ya que no solo tenemos la capa de streaming sino también una primera capa batch donde guardamos los datos en crudo. De modo que podemos decir que es una arquitectura híbrida. La razón por la que no he optado por eliminar las tablas que reciben los datos en crudo y pasar directamente de Nifi a Spark Streaming para luego pasar a Hadoop es precisamente porque guardar esos datos lo más cerca posible a su formato original es una buena práctica que nos ayuda a debuggear procesos scriptados sobre esos datos cuando su formato cambia, bajo esta perspectiva podemos entender el formato como información adicional sobre los datos, como un tipo de metadato que podemos necesitar consultar para, por ejemplo, próximos proyectos con esos datos. La contrapartida esto es necesitar más espacio de almacenamiento pero gracias a las tecnologías de escalado horizontal que nos permiten las tecnologías que estamos utilizando, esto no sería un problema

(ya que para esta práctica he puesto que la arquitectura no corre en mi ordenador local sino en un teórico clúster distribuido, que es donde los encontramos en los entornos de producción).

Por otra parte, para que esos directorios de las tablas hive se creen hay que crear las tablas Hive como objetos dentro del Hive, este es el código utilizado para crearlas:



```
%jdbc(hive)
CREATE TABLE IF NOT EXISTS uoc.transito_valencia ( gid int, denominacion String, Estado int, tramoId int, fiwareid String, geo_shape String , geo_point_2d String)
COMMENT 'Datos de os sensores de tráfico de la ciudad de Valencia'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

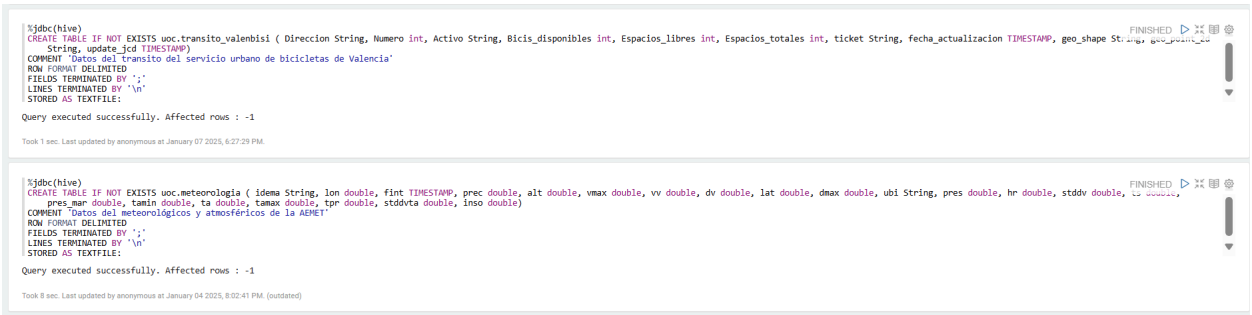
```
Query executed successfully. Affected rows : -1
```

Took 0 sec. Last updated by anonymous at December 16 2024, 8:10:39 PM. (outdated)


```
%jdbc(hive)
CREATE TABLE IF NOT EXISTS uoc.distritos ( object_id int, Nombre String, codigo_distrito int, area int, geo_shape String, geo_point_2d String , latitud double, longitud double)
COMMENT 'Delimitaciones de los distritos de Valencia'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

```
Query executed successfully. Affected rows : -1
```

Took 0 sec. Last updated by anonymous at January 04 2025, 2:05:23 PM. (outdated)



```
%jdbc(hive)
CREATE TABLE IF NOT EXISTS uoc.transito_valenbisi ( Direccion String, Numero int, Activo String, Bicis_disponibles int, Espacios_libres int, Espacios_totales int, ticket String, fecha_actualizacion TIMESTAMP, geo_shape String, geo_point_2d
  String, update_jcd TIMESTAMP)
COMMENT 'Datos del transito del servicio urbano de bicicletas de Valencia'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

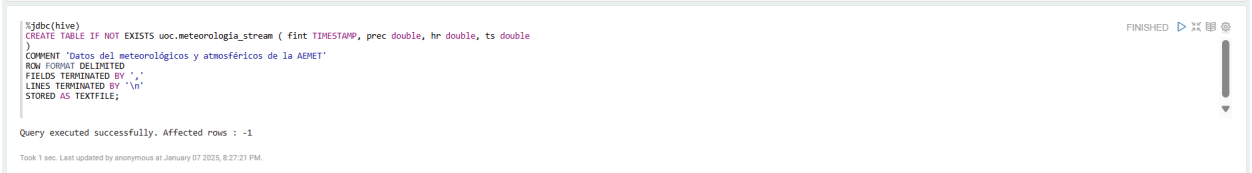
```
Query executed successfully. Affected rows : -1
```

Took 1 sec. Last updated by anonymous at January 07 2025, 6:27:29 PM.


```
%jdbc(hive)
CREATE TABLE IF NOT EXISTS uoc.meteorologia ( idema String, lon double, fint TIMESTAMP, prec double, alt double, vmax double, vv double, dv double, lat double, dmax double, ubi String, pres double, hr double, stddv double, ts double,
  pres_mar double, tamin double, ta double, tamax double, tpr double, stddvta double, inso double)
COMMENT 'Datos del meteorológicos y atmosféricos de la AEMET'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

```
Query executed successfully. Affected rows : -1
```

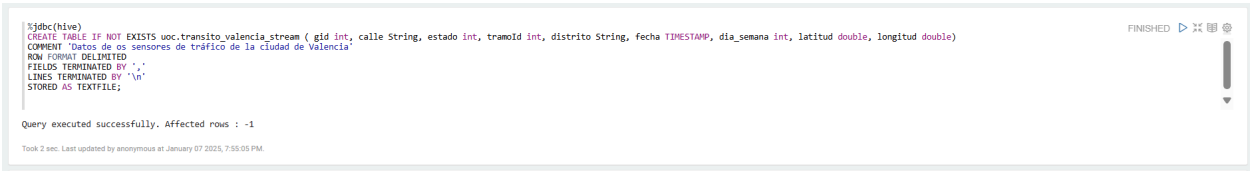
Took 8 sec. Last updated by anonymous at January 04 2025, 8:02:41 PM. (outdated)



```
%jdbc(hive)
CREATE TABLE IF NOT EXISTS uoc.meteorologia_stream ( fint TIMESTAMP, prec double, hr double, ts double
)
COMMENT 'Datos del meteorológicos y atmosféricos de la AEMET'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

```
Query executed successfully. Affected rows : -1
```

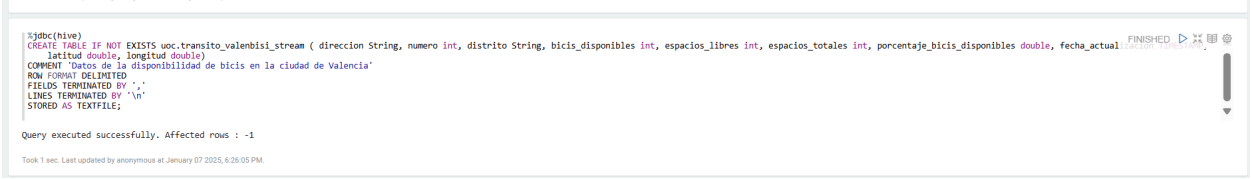
Took 1 sec. Last updated by anonymous at January 07 2025, 8:27:21 PM.



```
%jdbc(hive)
CREATE TABLE IF NOT EXISTS uoc.transito_valencia_stream ( gid int, calle String, estado int, tramoId int, distrito String, fecha TIMESTAMP, dia_semana int, latitud double, longitud double)
COMMENT 'Datos de os sensores de tráfico de la ciudad de Valencia'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

```
Query executed successfully. Affected rows : -1
```

Took 2 sec. Last updated by anonymous at January 07 2025, 7:55:05 PM.



```
%jdbc(hive)
CREATE TABLE IF NOT EXISTS uoc.transito_valenbisi_stream ( direccion String, numero int, distrito String, bicis_disponibles int, espacios_libres int, espacios_totales int, porcentaje_bicis_disponibles double, fecha_actual
    latitud double, longitud double)
COMMENT 'Datos de la disponibilidad de bicis en la ciudad de Valencia'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

```
Query executed successfully. Affected rows : -1
```

Took 1 sec. Last updated by anonymous at January 07 2025, 6:26:05 PM.

Procesado

En cuanto el procesado existe el procesado en batch y el procesado en streaming.
El procesado en batch actúa como un complemento del procesado en streaming que crea dataframes de Spark de referencia para poder hacer un join dentro del streaming que añada los barrios a los dataframes en tiempo real que vamos recibiendo.

Procesado en Batch

A continuación detallo el procesado en batch con el que obtenemos el dataframe que nos devuelve el distrito al cual pertenecen cada calle y cada punto ValenBisi respectivamente.
Procesado Batch de transito_valencia + distritos:

```
%spark2

// Importa la sesión de Spark
val spark = org.apache.spark.sql.SparkSession.builder().enableHiveSupport().getOrCreate()

// Lee la tabla Hive
val dfWithDupl = spark.sql("SELECT * FROM uoc.transito_valencia")

val df = dfWithDupl.dropDuplicates("denominacion")

// Muestra el contenido de la tabla
df.show()
```

gid	denominacion	estado	tramoid	fiwareid	geo_shape	geo_point_2d
1872	CLARIANO D'ENTRADA	0	453	EstadoTrafico_453	{"coordinates"...	39.47812446802766...
2072	PASO INFERIOR CAT...	5	277	EstadoTrafico_277	{"coordinates"...	39.48099622267202...
2191	PÍO BAROJA (DE GR...	0	178	EstadoTrafico_178	{"coordinates"...	39.47750695112612...
2207	LLANO DE ZAIDÍA (...	0	446	EstadoTrafico_446	{"coordinates"...	39.48257278091984...
1959	GENERAL PALANCA	0	127	EstadoTrafico_127	{"coordinates"...	39.47298264684471...
1928	PRIMADO REIG (DE ...	0	150	EstadoTrafico_150	{"coordinates"...	39.48122239683984...
1903	MESTRE RODRIGO (D...	0	46	EstadoTrafico_46	{"coordinates"...	39.47984415696816...
1978	PLAZA AMÉRICA HAC...	0	39	EstadoTrafico_39	{"coordinates"...	39.46972984994812...
2161	VÍA DE SERVICIO D...	0	415	EstadoTrafico_415	{"coordinates"...	39.49786997709089...
2037	CAMPANAR (DE TIRS...	0	239	EstadoTrafico_239	{"coordinates"...	39.47928834615428...
1864	CLARIANO D´EIXIDA	0	2	EstadoTrafico_2	{"coordinates"...	39.4782757347188,...
2224	BLASCO IBAÑEZ (DE...	0	422	EstadoTrafico_422	{"coordinates"...	39.47567158968992...
1897	GENERAL AVILES (D...	0	201	EstadoTrafico_201	{"coordinates"...	39.48460789214968...
2022	PASO INFERIOR PRO...	5	192	EstadoTrafico_192	{"coordinates"...	39.45450938502522...

Took 46 sec. Last updated by anonymous at January 10 2025, 4:20:52 PM. (outdated)

```
%spark2
val dfWithLatLong = df.withColumn("latitud", split(col("geo_point_2d"), ",")(0).cast("double"))
                      .withColumn("longitud", split(col("geo_point_2d"), ",")(1).cast("double"))
val dfRemCol = dfWithLatLong.drop("fiwareid", "geo_shape","geo_point_2d")

// Mostrar el resultado
dfRemCol.show()
```

gid	denominacion	estado	tramoid	latitud	longitud
1872	CLARIANO D'ENTRADA	0	453	39.478124468027666	-0.35172374690040326
2072	PASO INFERIOR CAT...	5	277	39.480996222672026	-0.3518736145349449
2191	PÍO BAROJA (DE GR...	0	178	39.477506951126124	-0.40678078450819233
2207	LLANO DE ZAIDÍA (...	0	446	39.48257278091984	-0.38059117746346927
1959	GENERAL PALANCA	0	127	39.47298264684471	-0.36905445362031064
1928	PRIMADO REIG (DE ...	0	150	39.48122239683984	-0.3591387128407713
1903	MESTRE RODRIGO (D...	0	46	39.47984415696816	-0.3991210251862677
1978	PLAZA AMÉRICA HAC...	0	39	39.46972984994812	-0.3645844412809193
2161	VÍA DE SERVICIO D...	0	415	39.49786997709089	-0.3702672254981869
2037	CAMPANAR (DE TIRS...	0	239	39.47928834615428	-0.393913840353462
1864	CLARIANO D´EIXIDA	0	2	39.4782757347188	-0.3514178017354239
2224	BLASCO IBAÑEZ (DE...	0	422	39.475671589689924	-0.3539951342629055
1897	GENERAL AVILES (D...	0	201	39.48460789214968	-0.3991087551987221 3
2022	PASO INFERIOR PRO...	5	192	39.45450938502522	-0.3537327913242484

Took 4 sec. Last updated by anonymous at January 10 2025, 4:21:04 PM.

```
%spark2
val dfWithDat = dfRemCol.withColumn("date", date_trunc("hour", current_timestamp()))
val dfWithDate = dfWithDat.withColumn("dia_semana", dayofweek(col("date")))
// 1 es domingo por conveción en el mundo de las bases de datos
dfWithDate.show()
```

gid	denominacion	estado	tramoid	latitud	longitud	date	dia_semana
1872	CLARIANO D'ENTRADA	0	453	39.478124468027666	-0.35172374690040326	2025-01-10 15:00:00	6
2072	PASO INFERIOR CAT...	5	277	39.480996222672026	-0.3518736145349449	2025-01-10 15:00:00	6
2191	PÍO BAROJA (DE GR...	0	178	39.477506951126124	-0.40678078450819233	2025-01-10 15:00:00	6
2207	LLANO DE ZAIDÍA (...	0	446	39.48257278091984	-0.38059117746346927	2025-01-10 15:00:00	6
1959	GENERAL PALANCA	0	127	39.47298264684471	-0.36905445362031064	2025-01-10 15:00:00	6
1928	PRIMADO REIG (DE ...	0	150	39.48122239683984	-0.3591387128407713	2025-01-10 15:00:00	6
1903	MESTRE RODRIGO (D...	0	46	39.47984415696816	-0.3991210251862677	2025-01-10 15:00:00	6
1978	PLAZA AMÉRICA HAC...	0	39	39.46972984994812	-0.3645844412809193	2025-01-10 15:00:00	6
2161	VÍA DE SERVICIO D...	0	415	39.49786997709089	-0.3702672254981869	2025-01-10 15:00:00	6
2037	CAMPANAR (DE TIRS...	0	239	39.47928834615428	-0.393913840353462	2025-01-10 15:00:00	6
1864	CLARIANO D´EIXIDA	0	2	39.4782757347188	-0.3514178017354239	2025-01-10 15:00:00	6
2224	BLASCO IBAÑEZ (DE...	0	422	39.475671589689924	-0.3539951342629055	2025-01-10 15:00:00	6
1897	GENERAL AVILES (D...	0	201	39.48460789214968	-0.39910875519872213	2025-01-10 15:00:00	6
2022	PASO INFERIOR PRO...	5	192	39.45450938502522	-0.3537327913242484	2025-01-10 15:00:00	6

Took 2 sec. Last updated by anonymous at January 10 2025, 4:21:31 PM.

```
%spark2

val distritos = spark.sql("SELECT * FROM uoc.distritos")
distritos.show()
```

object_id	nombre	codigo_distrito	area	geo_shape	geo_point_2d	latitud	longitud
6	LA SAIDIA	5	19439315	{\"coordinates\"...	39.48509467962349...	39.48509467962349	-0.3750723425324177
8	EL PLA DEL REAL	6	16927128	{\"coordinates\"...	39.4745936913893,...	39.4745936913893	-0.3603780274174357
10	EXTRAMURS	3	19716162	{\"coordinates\"...	39.46902418764345...	39.469024187643456	-0.385780005864596
12	L'EIXAMPLE	2	17331405	{\"coordinates\"...	39.46411452334249...	39.46411452334249	-0.3704292703233094
13	CAMINS AL GRAU	12	23674732	{\"coordinates\"...	39.46279474527259...	39.46279474527259	-0.3478024542887355
14	JESUS	9	2984760	{\"coordinates\"...	39.44820147753643...	39.44820147753643	-0.3917921797268368
1	QUATRE CARRERES	10	null	{\"coordinates\"...	39.44509289592729...	39.44509289592729	-0.35868167840872534
167	POBLATS DEL SUD	19	null	{\"coordinates\"...	39.35589278378015...	39.35589278378015	-0.34738500202809064
23	PATRAIX	8	null	{\"coordinates\"...	39.45839774793266...	39.458397747932665	-0.4004125691493868
216	POBLATS DEL NORD	17	null	{\"coordinates\"...	39.52468048556513...	39.52468048556513	-0.35945821125735933
199	POBLATS DEL NORD	17	null	{\"coordinates\"...	39.55880504953868...	39.558805049538684	-0.30316995651115557
55	BENIMACLET	14	null	{\"coordinates\"...	39.48605709546641...	39.48605709546641	-0.35570925948113463
103	CAMPANAR	4	null	{\"coordinates\"...	39.48544365860748...	39.48544365860748	-0.4059322619837058
183	POBLATS MARITIMS	11	null	{\"coordinates\"...	39.45195253044364...	39.45195253044364	-0.32633568267921914

Took 1 sec. Last updated by anonymous at January 10 2025, 4:21:41 PM.

```
%spark2
import org.apache.spark.sql.expressions.Window

val dfWithDateRenamed = dfWithDate.withColumnRenamed("latitud", "latitud_dfWithDate")
                                  .withColumnRenamed("longitud", "longitud_dfWithDate")
                                  .withColumnRenamed("denominacion", "calle")
val distritosRenamed = distritos.withColumnRenamed("latitud", "latitud_distritos")
                                .withColumnRenamed("longitud","longitud_distritos")

val dfCrossjoin = dfWithDateRenamed.crossJoin(distritosRenamed)
  .withColumn("distancia_lat", abs(col("latitud_dfWithDate") - col("latitud_distritos")))
  .withColumn("distancia_lon", abs(col("longitud_dfWithDate") - col("longitud_distritos")))
  .withColumn("dist_total", col("distancia_lat") + col("distancia_lon"))

val ventana = Window.partitionBy("calle").orderBy("dist_total")

val mapeo = dfCrossjoin.withColumn("rank", rank().over(ventana))
            .filter(col("rank") === 1)
            .withColumnRenamed("nombre", "distrito")
            .select("calle", "distrito", "dist_total")

mapeo.show(false)
```

```
+-----------------------------------------------------------+---------------+---------------------+
|calle                                                      |distrito       |dist_total           |
+-----------------------------------------------------------+---------------+---------------------+
|CLARIANO D'ENTRADA                                         |ALGIROS        |0.009481855851544208 |
|PASO INFERIOR CATALUÑA DE SALIDA                           |BENIMACLET     |0.008896517740570764 |
|PÍO BAROJA (DE GRAL. AVILÉS A MANUEL DE FALLA)             |CAMPANAR       |0.008785230005842637 |
|LLANO DE ZAIDÍA (DE PUENTE DE LAS ARTES A POETA MONMENEU)|LA SAIDIA      |0.008040733634699704 |
|GENERAL PALANCA                                            |CIUTAT VELLA   |0.009138024004816336 |
|PRIMADO REIG (DE CATALUÑA A EMILIO BARÓ)                   |EL PLA DEL REAL|0.007868020027202438 |
|MESTRE RODRIGO (DES DE TIRSO DE MOLINA A GRAL. AVILES)     |CAMPANAR       |0.012410738436754998 |
|PLAZA AMÉRICA HACIA NAVARRO REVERTER                       |EL PLA DEL REAL|0.00907025530466199  |
|VÍA DE SERVICIO DE HNOS. MACHADO HACIA ALFAHUIR            |RASCANYA       |0.004746058832040212 |
|CAMPANAR (DE TIRSO DE MOLINA A PÍO XII)                    |BENICALAP      |0.01720735773188925  |
|CLARIANO D´EIXIDA                                          |ALGIROS        |0.009327177377698725 |
|BLASCO IBAÑEZ (DE ARAGÓN A CARDENAL BENLLOCH)              |EL PLA DEL REAL|0.007460791455152049 |
|GENERAL AVILES (DE PIO XII A MESTRE RODRIGO)               |CAMPANAR       |0.0076592732427808174|
|PASO INFERIOR PROFESOR LÓPEZ PIÑERO DE SALIDA              |CAMINS AL GRAU |0.014215697282883355 |
```

Took 14 sec. Last updated by anonymous at January 10 2025, 4:22:01 PM.

Procesado Batch de transito_valenbisi + distritos:

```
%spark2
val dfBici = spark.sql("SELECT * FROM uoc.transito_valenbisi")

val dfBiciDupl = dfBici.dropDuplicates("direccion")

dfBiciDupl.show()
```

```
+--------------------+------+------+----------------+--------------+---------------+------+------------------+--------------------+--------------------+----------+
|           direccion|numero|activo|bicis_disponibles|espacios_libres|espacios_totales|ticket|fecha_actualizacion|           geo_shape|        geo_point_2d|update_jcd|
+--------------------+------+------+----------------+--------------+---------------+------+------------------+--------------------+--------------------+----------+
|          Colón, 44|    70|     T|               2|            15|             17|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.46921322131946...|      null|
|Jerónimo Monsoriu...|    73|     T|               2|            18|             20|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.4662592905475,...|      null|
|Peris y Valero - ...|    37|     T|              19|             1|             20|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.46093780002837...|      null|
|Juan XXIII - Domi...|   175|     T|               6|             9|             15|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.49258939642365...|      null|
|Alfonso el Magnán...|    13|     T|              11|            13|             24|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.4720623227334,...|      null|
|       Zapadores, 23|   181|     T|               4|            11|             15|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.45799927509706...|      null|
|Corts Valencianes...|   204|     T|               0|            20|             20|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.48584338059417...|      null|
|Ribera - Plaza Ay...|    15|     T|               1|            16|             34|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.46908831533384...|      null|
|Giorgeta - Roig d...|   195|     T|               3|            13|             16|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.45938428720848...|      null|
|Hospital Nueva Fe...|   188|     T|               0|            40|             40|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.44462122517277...|      null|
|Ramiro de Maeztu ...|    72|     T|               4|            11|             15|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.46725720116240...|      null|
|Molinell - Calder...|    77|     T|              11|             9|             20|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.48497336307338...|      null|
|Perez Galdós - No...|   201|     T|               8|            12|             20|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.47171233289752...|      null|
|Jaime Roig - Bach...|   119|     T|              11|             8|             20|     F|2025-01-10 15:39:42|"{\"coordinates\"...|39.48252035422283...|      null|
```

Took 1 sec. Last updated by anonymous at January 10 2025, 3:57:34 PM.

```
%spark2
val dfLatLong = dfBiciDupl.withColumn("latitud", split(col("geo_point_2d"), ",")(0).cast("double"))
                          .withColumn("longitud", split(col("geo_point_2d"), ",")(1).cast("double"))
val dfRmCol = dfLatLong.drop("fiwareid", "geo_shape","geo_point_2d")

val dfTrunc = dfRmCol.withColumn("fecha_actualizacion", date_trunc("hour", col("fecha_actualizacion")))

dfTrunc.show()
```

```
+--------------------+------+------+----------------+--------------+---------------+------+-------------------+----------+-----------------+-------------------+
|           direccion|numero|activo|bicis_disponibles|espacios_libres|espacios_totales|ticket|fecha_actualizacion|update_jcd|          latitud|           longitud|
+--------------------+------+------+----------------+--------------+---------------+------+-------------------+----------+-----------------+-------------------+
|           Colón, 44|    70|     T|               2|            15|             17|     F|2025-01-10 15:00:00|      null|39.469213221319464| -0.3716462894107889|
|Jerónimo Monsoriu...|    73|     T|               2|            18|             20|     F|2025-01-10 15:00:00|      null| 39.4662592905475| -0.3428792786496773|
|Peris y Valero - ...|    37|     T|              19|             1|             20|     F|2025-01-10 15:00:00|      null|39.460937800028375| -0.3664406250017559|
|Juan XXIII - Domi...|   175|     T|               6|             9|             15|     F|2025-01-10 15:00:00|      null| 39.49258939642365|-0.38271239433339566|
|Alfonso el Magnán...|    13|     T|              11|            13|             24|     F|2025-01-10 15:00:00|      null| 39.4720623227334|-0.37087436964085335|
|       Zapadores, 23|   181|     T|               4|            11|             15|     F|2025-01-10 15:00:00|      null| 39.45799927509706| -0.3681913713233589|
|Corts Valencianes...|   204|     T|               0|            20|             20|     F|2025-01-10 15:00:00|      null|39.485843380594176| -0.3965874461307865|
|Ribera - Plaza Ay...|    15|     T|               1|            16|             34|     F|2025-01-10 15:00:00|      null| 39.46908831533384| -0.3756373880434071|
|Giorgeta - Roig d...|   195|     T|               3|            13|             16|     F|2025-01-10 15:00:00|      null|39.459384287208486| -0.3842814248129742|
|Hospital Nueva Fe...|   188|     T|               0|            40|             40|     F|2025-01-10 15:00:00|      null| 39.44462122517277| -0.3751583293336074|
|Ramiro de Maeztu ...|    72|     T|               4|            11|             15|     F|2025-01-10 15:00:00|      null|39.467257201162404|-0.34644628196214244|
|Molinell - Calder...|    77|     T|              11|             9|             20|     F|2025-01-10 15:00:00|      null| 39.48497336307338|-0.36566734195835493|
|Perez Galdós - No...|   201|     T|               8|            12|             20|     F|2025-01-10 15:00:00|      null| 39.47171233289752|-0.39366844724894096|
|Jaime Roig - Bach...|   119|     T|              11|             8|             20|     F|2025-01-10 15:00:00|      null| 39.48252035422283|-0.36322633473676447|
```

Took 1 sec. Last updated by anonymous at January 10 2025, 3:58:05 PM.

```
%spark2
val distritosBici = spark.sql("SELECT * FROM uoc.distritos")
```

```
distritosBici: org.apache.spark.sql.DataFrame = [object_id: int, nombre: string ... 6 more fields]
```

Took 0 sec. Last updated by anonymous at January 10 2025, 3:58:13 PM.

```
%spark2
import org.apache.spark.sql.expressions.Window

val dfTruncRenamed = dfTrunc.withColumnRenamed("latitud", "latitud_dfTrunc")
                            .withColumnRenamed("longitud", "longitud_dfTrunc")
                            .withColumnRenamed("direccion", "direccion_sobrante")

val distritosRenamedBici = distritosBici.withColumnRenamed("latitud", "latitud_distritosBici")
                                        .withColumnRenamed("longitud","longitud_distritosBici")

val dfCrossjoinBici = dfTruncRenamed.crossJoin(distritosRenamedBici)
  .withColumn("distancia_lat", abs(col("latitud_dfTrunc") - col("latitud_distritosBici")))
  .withColumn("distancia_lon", abs(col("longitud_dfTrunc") - col("longitud_distritosBici")))
  .withColumn("dist_total", col("distancia_lat") + col("distancia_lon"))

val ventanaBici = Window.partitionBy("direccion_sobrante").orderBy("dist_total")

val mapeoBici = dfCrossjoinBici.withColumn("rank", rank().over(ventanaBici))
            .filter(col("rank") === 1)
            .select("direccion_sobrante", "nombre", "dist_total")

mapeoBici.show(false)
```

```
+-------------------------------------+---------------+---------------------+
|direccion_sobrante                   |nombre         |dist_total           |
+-------------------------------------+---------------+---------------------+
|Colón, 44                            |L'EIXAMPLE     |0.0063157170644555505|
|Jerónimo Monsoriu - Industria        |CAMINS AL GRAU |0.008387720913967112 |
|Peris y Valero - Luis Santángel      |L'EIXAMPLE     |0.00716536863566597  |
|Juan XXIII - Domingo Gómez           |BENICALAP      |0.011102276025761137 |
|Alfonso el Magnánimo - Nave          |CIUTAT VELLA   |0.008238432095586334 |
|Zapadores, 23                        |L'EIXAMPLE     |0.0083531472453795   |
|Corts Valencianes - General Avilés   |CAMPANAR       |0.00974453783961532  |
|Ribera - Plaza Ayuntamiento          |CIUTAT VELLA   |0.00644942109259089  |
|Giorgeta - Roig de Corella           |EXTRAMURS      |0.011138481486590968 |
|Hospital Nueva Fe (consultas externas)|QUATRE CARRERES|0.016948316279403408 |
|Ramiro de Maeztu - Peris Brell       |CAMINS AL GRAU |0.0058186282164080505|
|Molinell - Calderón de la Barca      |LA SAIDIA      |0.009526317124169847 |
|Perez Galdós - Nou Moles             |EXTRAMURS      |0.010576586638407104 |
|Jaime Roig - Bachiller               |EL PLA DEL REAL|0.010774970152854946 |
```

Took 9 sec. Last updated by anonymous at January 10 2025, 3:58:27 PM.

Aunque el código se explica por sí mismo está bien explicar el propósito general del código. Para incluir cada punto Valenbisi o calle dentro de un distrito concreto se ha usado su longitud y latitud y se ha comparado con la longitud y latitud de cada uno de los puntos centrales de los distritos expresados también en longitud y latitud calculando la distancia de cada calle o punto Valenbisi a todos los barrios y seleccionando el barrio que más cerca está. Podríamos haber utilizado la distancia de Harvensire pero la curvatura de la tierra es irrelevante a la escala a la que trabajamos así que sencillamente se ha usado una aproximación que suma las diferencias entre longitud y latitud se utiliza como medidor de la distancia. El resultado final es una tabla con los cada uno de los barrios o puntos Valenbisi y el barrio con distancia total mínima asignada. Es decir, tablas de lookup llamadas “mapeo” y “mapeoBici”.

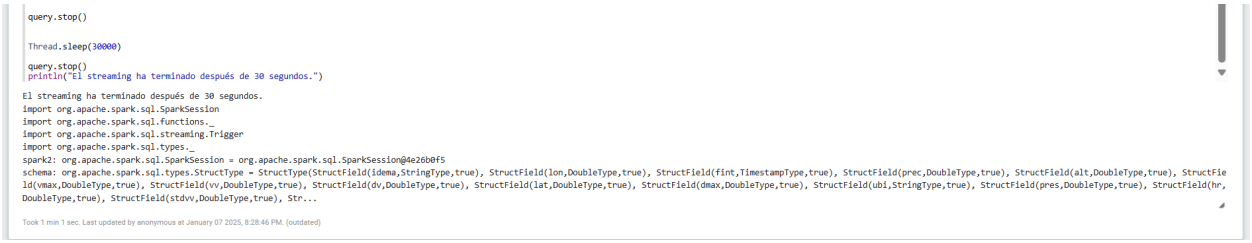
(Acabo de reparar en que la distancia debería haberse calculado mediante la distancia euclídea df.TruncRenamed.withColumn("dist_total", sqrt(pow(col("distancia_lat"), 2) + pow(col("distancia_lon"), 2))), si bien es un detalle que importante para los resultados arrojados es algo que se puede solventar cambiando el código por el propuesto ahora y que no va a arrojar grandes diferencias respecto del código utilizado ya que los números son pequeños.)

Procesado en streaming

El procesado en streaming utilizará estas tablas para devolver una transformación de los datos raw que llegan de nifi, hay tres procesos streaming, uno para la meteorología, otro para los datos de ValenBisi y otro para los datos del tránsito. A continuación los detallamos:

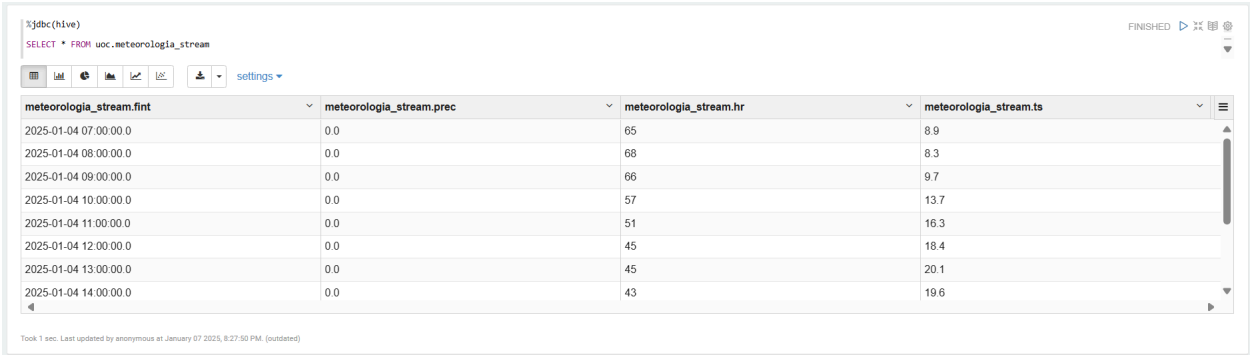
Streaming meteorología:

```
%spark2
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._
import org.apache.spark.sql.streaming.Trigger
import org.apache.spark.sql.types._

val spark2 = org.apache.spark.sql.SparkSession.builder().enableHiveSupport().getOrCreate()

val schema = StructType(Array(
  StructField("idema", StringType, true),
  StructField("lon", DoubleType, true),
  StructField("fint", TimestampType, true),
  StructField("prec", DoubleType, true),
  StructField("alt", DoubleType, true),
  StructField("vmax", DoubleType, true),
  StructField("vv", DoubleType, true),
  StructField("dv", DoubleType, true),
  StructField("lat", DoubleType, true),
  StructField("dmax", DoubleType, true),
  StructField("ubi", StringType, true),
  StructField("pres", DoubleType, true),
  StructField("hr", DoubleType, true),
  StructField("stdvv", DoubleType, true),
  StructField("ts", DoubleType, true),
  StructField("pres_mar", DoubleType, true),
  StructField("tamin", DoubleType, true),
  StructField("ta", DoubleType, true),
  StructField("tamax", DoubleType, true),
  StructField("tpr", DoubleType, true),
  StructField("stddv", DoubleType, true),
  StructField("inso", DoubleType, true)
 ))

val hdfsPath = "hdfs://namenode:8020/user/hive/warehouse/uoc.db/meteorologia"

val meteostr = spark2.readStream
  .format("csv")
  .option("delimiter",";")
  .schema(schema)
  .option("path", hdfsPath)
  .load()

val meteo_trans = meteostr.drop("idema", "lon", "alt", "vmax", "vv", "dv", "lat", "dmax", "stddv", "pres_mar", "tamin", "ta", "tamax", "tpr", "stddvta", "inso", "pres", "ubi", "stdvv")

val finish_meteo = meteo_trans.withColumn("fint", col("fint").cast("string"))

val outputPath = "hdfs://namenode:8020/user/hive/warehouse/uoc.db/meteorologia_stream/"

val outputPathTemp = "hdfs://namenode:8020/user/hive/warehouse/uoc.db/meteorologia_stream/temp"

val query = finish_meteo.writeStream
  .outputMode("append")
  .format("csv")
  .option("path", outputPath)
  .option("checkpointLocation", outputPath + "_checkpoint")
  .trigger(Trigger.ProcessingTime("10 seconds"))
  .start()


Thread.sleep(30000)
```



Aquí simplemente hemos hecho dos simples transformaciones en las que he pasado a “String” la hora para asegurarnos de que Hive pueda interpretar la hora ya que Hive no interpreta el TimestampType de Spark bien. Además, he quitado datos que no afectan directamente al tráfico como la dirección del viento o la velocidad de este. Nos hemos quedado con la precipitación acumulada de la última hora, la humedad relativa ya que si llega a niveles cercanos a 100% la probabilidad de lluvia aumenta mucho y además está relacionada, junto con la temperatura, con la sensación térmica, que también hemos cogido. Por último, también tenemos la hora a la que fue tomada la medición. El punto desde el cuál es tomada la medición es el aeropuerto de Valencia (lo podemos ver en los flujos de datos de Nifi y además pedí a la API de la AEMET que solo me envíe la información de esta estación, las otras estaban fuera de la ciudad de Valencia).

Aquí podemos ver cómo la información se pasa correctamente a la tabla meteorologia_stream:



Streaming de las bicicletas:

```
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._
import org.apache.spark.sql.streaming.Trigger
import org.apache.spark.sql.types._
import org.apache.spark.sql.expressions.Window


val schema = StructType(Array(
  StructField("direccion", StringType, true),
  StructField("numero", IntegerType, true),
  StructField("activo", StringType, true),
  StructField("bicis_disponibles", IntegerType, true),
  StructField("espacios_libres", IntegerType, true),
  StructField("espacios_totales", IntegerType, true),
  StructField("ticket", StringType, true),
  StructField("fecha_actualizacion", TimestampType, true),
  StructField("geo_shape", StringType, true),
  StructField("geo_point_2d", StringType, true),
  StructField("update_jcd", TimestampType, true)
))

val hdfsPath = "hdfs://namenode:8020/user/hive/warehouse/uoc.db/transito_valenbisi"

val spark2 = org.apache.spark.sql.SparkSession.builder().enableHiveSupport().getOrCreate()

val dfBici = spark2.readStream
  .format("csv")
  .option("delimiter",";")
  .schema(schema)
  .option("path", hdfsPath)
  .load()

val dfLatLong = dfBici.withColumn("latitud", split(col("geo_point_2d"), ",")(0).cast("double"))
                      .withColumn("longitud", split(col("geo_point_2d"), ",")(1).cast("double"))
val dfRmCol = dfLatLong.drop("fiwareid", "geo_shape","geo_point_2d")

val dfTrunc = dfRmCol.withColumn("fecha_actualizacion", date_trunc("hour", col("fecha_actualizacion")))

val dfTruncRenamed = dfTrunc.withColumnRenamed("latitud", "latitud_dfTrunc")
                            .withColumnRenamed("longitud", "longitud_dfTrunc")
                            .withColumnRenamed("direccion", "direccion_sobrante")

val dfWithDistrictBici = dfTrunc.join(mapeoBici, dfTrunc("direccion") === mapeoBici("direccion_sobrante"), "left")

val remDfWithDistrictBici = dfWithDistrictBici.drop("direccion_sobrante","dist_total")

val dfBiciRenamed = remDfWithDistrictBici.withColumnRenamed("nombre", "distrito")

val df_Bici = dfBiciRenamed.withColumn("porcentaje_bicis_disponibles", col("bicis_disponibles")/col("espacios_totales"))

val df_withDateBici = df_Bici.withColumn("fecha_actualizacion", col("fecha_actualizacion").cast("string"))
```

```
val finishBici = df_withDateBici.select("direccion", "numero", "distrito", "bicis_disponibles", "espacios_libres", "espacios_totales", "porcentaje_bicis_disponibles", "fecha_actualizacion", "latitud", "longitud")

val outputPathBici = "hdfs://namenode:8020/user/hive/warehouse/uoc.db/transito_valenbisi_stream"

val outputPathTempBici = "hdfs://namenode:8020/user/hive/warehouse/uoc.db/transito_valenbisi_stream/temp"

val query = finishBici.writeStream
  .outputMode("append")
  .format("csv")
  .option("path", outputPathBici)
  .option("checkpointLocation", outputPathBici + "_checkpoint")
  .trigger(Trigger.ProcessingTime("10 seconds"))
  .start()

Thread.sleep(30000)

query.stop()
println("El streaming ha terminado después de 30 segundos.")
```

```
-------------------------------------------
Batch: 0
-------------------------------------------
+--------------------------------------+------+--------------+----------------+--------------+---------------+-----------------------------+-------------------+-----------------+-------------------+
|direccion                             |numero|distrito      |bicis_disponibles|espacios_libres|espacios_totales|porcentaje_bicis_disponibles|fecha_actualizacion|latitud          |longitud           |
+--------------------------------------+------+--------------+----------------+--------------+---------------+-----------------------------+-------------------+-----------------+-------------------+
|Colón, 44                             |70    |L'EIXAMPLE    |2               |15            |17             |0.11764705882352941          |2025-01-10 15:00:00|39.469213221319464|-0.3716462894107889 |
|Jerónimo Monsoriu - Industria         |73    |CAMINS AL GRAU|2               |18            |20             |0.1                          |2025-01-10 15:00:00|39.4662592905475  |-0.3428792786496773 |
|Peris y Valero - Luis Santángel       |37    |L'EIXAMPLE    |19              |1             |20             |0.95                         |2025-01-10 15:00:00|39.460937800028375|-0.3664406250017559 |
|Juan XXIII - Domingo Gómez            |175   |BENICALAP     |6               |9             |15             |0.4                          |2025-01-10 15:00:00|39.49258939642365 |-0.38271239433339566|
|Alfonso el Magnánimo - Nave           |13    |CIUTAT VELLA  |11              |13            |24             |0.4583333333333333           |2025-01-10 15:00:00|39.4720623227334  |-0.37087436964085335|
|Zapadores, 23                         |181   |L'EIXAMPLE    |4               |11            |15             |0.26666666666666666          |2025-01-10 15:00:00|39.45799927509706 |-0.3681913713233589 |
|Corts Valencianes - General Avilés    |204   |CAMPANAR      |0               |20            |20             |0.0                          |2025-01-10 15:00:00|39.485843380594176|-0.3965874461307865 |
|Ribera - Plaza Ayuntamiento           |15    |CIUTAT VELLA  |1               |16            |34             |0.029411764705882353         |2025-01-10 15:00:00|39.46908831533384 |-0.3756373880434071 |
|Giorgeta - Roig de Corella            |195   |EXTRAMURS     |3               |13            |16             |0.1875                       |2025-01-10 15:00:00|39.459384287208486|-0.3842814248129742 |
|Hospital Nueva Fe (consultas externas)|188   |QUATRE CARRERES|0               |40            |40             |0.0                          |2025-01-10 15:00:00|39.44462122517277 |-0.3751583239336074 |
|Ramiro de Maeztu - Peris Brell        |72    |CAMINS AL GRAU|4               |11            |15             |0.26666666666666666          |2025-01-10 15:00:00|39.467257201162404|-0.34644628196214244|
```

Took 32 sec. Last updated by anonymous at January 10 2025, 3:59:45 PM. (outdated)

Streaming del transito:

```
%spark2

import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions._
import org.apache.spark.sql.streaming.Trigger
import org.apache.spark.sql.types._

val spark2 = org.apache.spark.sql.SparkSession.builder().enableHiveSupport().getOrCreate()

val schema = StructType(Array(
  StructField("gid", IntegerType, true),
  StructField("denominacion", StringType, true),
  StructField("estado", IntegerType, true),
  StructField("tramo_id", IntegerType, true),
  StructField("fiwareid", StringType, true),
  StructField("geo_shape", StringType, true),
  StructField("geo_point_2d", StringType, true)
))

val hdfsPath = "hdfs://namenode:8020/user/hive/warehouse/uoc.db/transito_valencia"


val df = spark2.readStream
  .format("csv")
  .option("delimiter",";")
  .schema(schema)
  .option("path", hdfsPath)
  .load()

val dfWithLatLong = df.withColumn("latitud", split(col("geo_point_2d"), ",")(0).cast("double"))
                      .withColumn("longitud", split(col("geo_point_2d"), ",")(1).cast("double"))
val dfRemCol = dfWithLatLong.drop("fiwareid", "geo_shape","geo_point_2d")

val dfWithDat = dfRemCol.withColumn("fecha", date_trunc("hour", current_timestamp()))

val dfWithDate = dfWithDat.withColumn("dia_semana", dayofweek(col("fecha")))

val dfWithDistrict = dfWithDate.join(mapeo, dfWithDate("denominacion") === mapeo("calle"), "left")

val remDfWithDistrict = dfWithDistrict.drop("denominacion","dist_total")

val dfRenamed = remDfWithDistrict.withColumnRenamed("nombre","distrito")

val dfReordered = dfRenamed.select("gid", "calle", "estado", "tramo_id", "distrito", "fecha", "dia_semana", "latitud", "longitud")

val dfWithStringDate = dfReordered.withColumn("fecha", col("fecha").cast("string"))

val outputPath = "hdfs://namenode:8020/user/hive/warehouse/uoc.db/transito_valencia_stream/"

val outputPathTemp = "hdfs://namenode:8020/user/hive/warehouse/uoc.db/transito_valencia_stream/temp"

val query = dfWithStringDate.writeStream
  .outputMode("append") // O puede ser "update" si quieres ver solo las filas actualizadas.
  .format("console")   // Especifica que quieres la salida en consola.
  .option("truncate", "false") // Evita que se trunquen las filas largas en la consola.
  .trigger(Trigger.ProcessingTime("10 seconds"))
  .start()
```

```
Thread.sleep(30000)

query.stop()

println("El streaming ha terminado después de 30 segundos.")
```

```
-------------------------------------------
Batch: 0
-------------------------------------------
+----+-----------------------------------------------------------+------+-------+--------------+-------------------+---------+-----------------+-------------------+
|gid |calle                                                      |estado|tramo_id|distrito      |fecha              |dia_semana|latitud          |longitud           |
+----+-----------------------------------------------------------+------+-------+--------------+-------------------+---------+-----------------+-------------------+
|1872|CLARIANO D'ENTRADA                                         |0     |453    |ALGIROS       |2025-01-10 15:00:00|6        |39.478124468027666|-0.35172374690040326|
|2072|PASO INFERIOR CATALUÑA DE SALIDA                           |5     |277    |BENIMACLET    |2025-01-10 15:00:00|6        |39.480996222672026|-0.3518736145349449 |
|2191|PÍO BAROJA (DE GRAL. AVILÉS A MANUEL DE FALLA)             |0     |178    |CAMPANAR      |2025-01-10 15:00:00|6        |39.477506951126124|-0.40678078450819233|
|2207|LLANO DE ZAIDÍA (DE PUENTE DE LAS ARTES A POETA MONMENEU)  |0     |446    |LA SAIDIA     |2025-01-10 15:00:00|6        |39.4825727809198 4|-0.38059117746346927|
|1959|GENERAL PALANCA                                            |0     |127    |CIUTAT VELLA  |2025-01-10 15:00:00|6        |39.47298264684471 |-0.36905445362031064|
|1928|PRIMADO REIG (DE CATALUÑA A EMILIO BARÓ)                   |0     |150    |EL PLA DEL REAL|2025-01-10 15:00:00|6        |39.48122239683984 |-0.3591387128407713 |
|1903|MESTRE RODRIGO (DES DE TIRSO DE MOLINA A GRAL. AVILES)     |0     |46     |CAMPANAR      |2025-01-10 15:00:00|6        |39.47984415696816 |-0.3991210251862677 |
|1978|PLAZA AMÉRICA HACIA NAVARRO REVERTER                       |0     |39     |EL PLA DEL REAL|2025-01-10 15:00:00|6        |39.46972904994812 |-0.3645844412809193 |
|2161|VÍA DE SERVICIO DE HNOS. MACHADO HACIA ALFAHUIR            |0     |415    |RASCANYA      |2025-01-10 15:00:00|6        |39.49786997780089 |-0.3702672254981869 |
|2037|CAMPANAR (DE TIRSO DE MOLINA A PÍO XII)                    |0     |239    |BENICALAP     |2025-01-10 15:00:00|6        |39.47928834615428 |-0.39391384035462   |
|1864|CLARIANO D´EIXIDA                                          |0     |2      |ALGIROS       |2025-01-10 15:00:00|6        |39.4782757347188  |-0.3514178017354239 |
|    |                                                           |      |       |              |                   |         |                 |                   |
+----+-----------------------------------------------------------+------+-------+--------------+-------------------+---------+-----------------+-------------------+

Took 1 min 0 sec. Last updated by anonymous at January 10 2025, 4:26:35 PM. (outdated)
```

Hay tres puntualizaciones que se deben hacer respecto de los scripts tal y como están ahora. Primeramente, si nos fijamos en el script de la meteorología y las bicicletas, la variable query en Scala se deberá instanciar en todos los scripts tal y como se hace en esos dos casos, ya que será así como guardaremos los resultados en el directorio de la tabla correspondiente (en cada caso, deberemos ajustar el directorio del output y el directorio de los checkpoints para que apunte a ese directorio). En el cas del tránsito instanciamos la query solo para que el dataframe resultante se envíe a la consola y se vea en pantalla pero una vez acabadas las transformaciones queremos guardar los datos a HDFS en todos los casos.
También hay que tener en cuenta que al final de los scripts, una vez se empiece a utilizar la infraestructura de datos, habrá que sustituir el siguiente fragmento final:

“Thread.sleep(30000)

query.stop()

println("El streaming ha terminado después de 30 segundos.")”

Por lo siguiente:

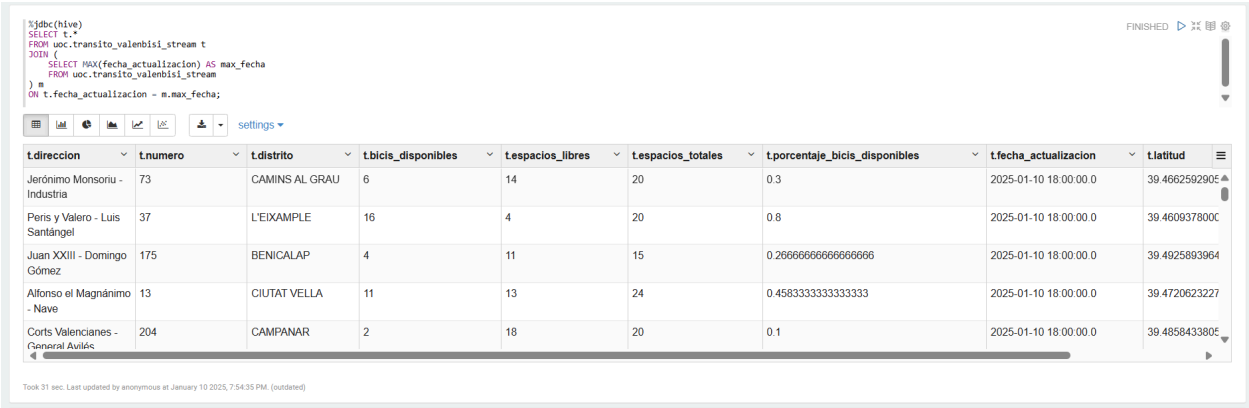
“query.awaitTermination()”

De este modo, Spark Streaming estará siempre mirando los nuevos datos que aparezcan desde Nifi a la carpeta de los datos en crudo, los leerá, los procesará, y guardará los datos en las carpetas “_stream” continuamente.

Por último, los timestamps están truncados a la hora de modo que sale la misma hora todo el rato y sin minutos hasta que pasa una hora, momento en el sale la siguiente. Esto lo puse así porque inicialmente escogí el umbral temporal de actualización de la información de la fuente de información que más tardaba en actualizarse ya que pretendía consolidar toda la información en una sola tabla que actuaría como serving layer sobre la cuál proyectar datos. Finalmente desestimé este acercamiento y hay 3 tablas de streaming distintas que forman la serving layer, cada una actualizada con tanta frecuencia como lo hace la fuente de datos. Esto significa que dentro de los scripts de streaming de transito_valencia_stream y transito_valencia_valenbisi truncaremos los timestamps al minuto y así podremos distinguir cada nuevo flujo de datos que llegue desde las fuentes de datos. Gracias a esto podremos recuperar de cada una de las tablas la información más reciente filtrando por la fecha más actual en cada caso con el mismo comando HiveQL. De esta manera hemos creado un sistema resistente a fallos, demoras, o asimetrías en la frecuencia de suministro de la información ya que siempre recuperaremos los datos más recientes posible, y podremos ver lo recientes y darnos cuenta de si hay alguna fuente de información que no se está actualizando con la frecuencia con la que debería ya que la fecha de actualización es de hace, por ejemplo, 2 horas.
Pero no acaba ahí, hay otra importante ventaja a tener en cuenta de la cuál, por desgracia, no voy a poder mostrar ejemplos gráficos ya que para ello habría tenido que haber estado recogiendo y almacenando datos de manera periódica pero lo importante es que es esta infraestructura nos permite hacerlo. Hablo del hecho de que tener en las tres tablas ”_stream” un acumulado temporal de los cambios de los sensores y las bicis, es decir, tenemos series temporales y eso nos permite utilizar el tiempo como una variable más, y cuanto más tiempo estemos recogiendo datos, más amplios puede ser el espectro temporal que estudiemos, desde horas, hasta años. Tanto solo tenemos que incluír en la consulta HiveQL a las tablas “stream” un *WHERE {nombre de la columna fecha en cada caso} > {fecha límite que queremos estudiar}.*

A continuación podemos ver los datos en sus correspondientes tablas Hive si las consultamos:

Resultados de las bicis:



```
%jdbc(hive)
SELECT t.*
FROM uoc.transito_valenbisi_stream t
JOIN (
    SELECT MAX(fecha_actualizacion) AS max_fecha
    FROM uoc.transito_valenbisi_stream
) m
ON t.fecha_actualizacion = m.max_fecha;
```

FINISHED

settings

t.direccion	t.numero	t.distrito	t.bicis_disponibles	t.espacios_libres	t.espacios_totales	t.porcentaje_bicis_disponibles	t.fecha_actualizacion	t.latitud
Jerónimo Monsoriu - Industria	73	CAMINS AL GRAU	6	14	20	0.3	2025-01-10 18:00:00.0	39.4662592905
Peris y Valero - Luis Santángel	37	L'EIXAMPLE	16	4	20	0.8	2025-01-10 18:00:00.0	39.4609378000
Juan XXIII - Domingo Gómez	175	BENICALAP	4	11	15	0.2666666666666666	2025-01-10 18:00:00.0	39.4925893964
Alfonso el Magnánimo - Nave	13	CIUTAT VELLA	11	13	24	0.4583333333333333	2025-01-10 18:00:00.0	39.4720623227
Corts Valencianes - General Avilés	204	CAMPANAR	2	18	20	0.1	2025-01-10 18:00:00.0	39.4858433805

Took 31 sec. Last updated by anonymous at January 10 2025, 7:54:35 PM. (outdated)

Resultado del streaming sobre la meteorología:

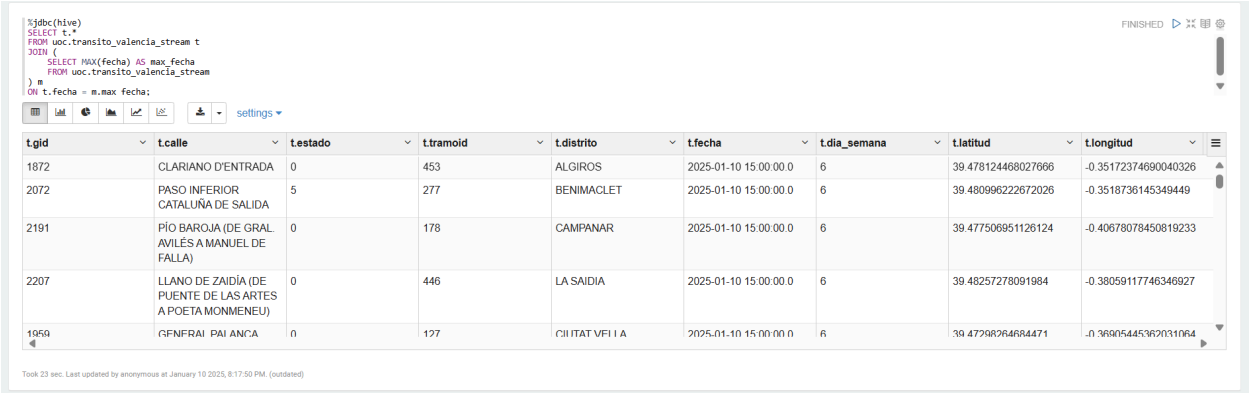
```
%jdbc(hive)
SELECT t.*
FROM uoc.meteorologia_stream t
JOIN (
    SELECT MAX(fint) AS max_fecha
    FROM uoc.meteorologia_stream
) m
ON t.fint = m.max_fecha;
```

FINISHED

settings

t.fint	t.prec	t.hr	t.ts
2025-01-04 19:00:00.0	0.0	49	10.3

Resultado del streaming sobre el tránsito:



```
%jdbc(hive)
SELECT t.*
FROM uoc.transito_valencia_stream t
JOIN (
    SELECT MAX(fecha) AS max_fecha
    FROM uoc.transito_valencia_stream
) m
ON t.fecha = m.max_fecha;
```

FINISHED

settings

t.gid	t.calle	t.estado	t.tramoid	t.distrito	t.fecha	t.dia_semana	t.latitud	t.longitud
1872	CLARIANO D'ENTRADA	0	453	ALGIROS	2025-01-10 15:00:00.0	6	39.478124468027666	-0.35172374690040326
2072	PASO INFERIOR CATALUÑA DE SALIDA	5	277	BENIMACLET	2025-01-10 15:00:00.0	6	39.480996222672026	-0.3518736145349449
2191	PÍO BAROJA (DE GRAL. AVILÉS A MANUEL DE FALLA)	0	178	CAMPANAR	2025-01-10 15:00:00.0	6	39.477506951126124	-0.40678078450819233
2207	LLANO DE ZAIDÍA (DE PUENTE DE LAS ARTES A POETA MONMENEU)	0	446	LA SAIDIA	2025-01-10 15:00:00.0	6	39.48257278091984	-0.38059117746346927
1959	GENERAL PALANCA	0	127	CIUTAT VELLA	2025-01-10 15:00:00.0	6	39.4729826468447	-0.3690544536203106

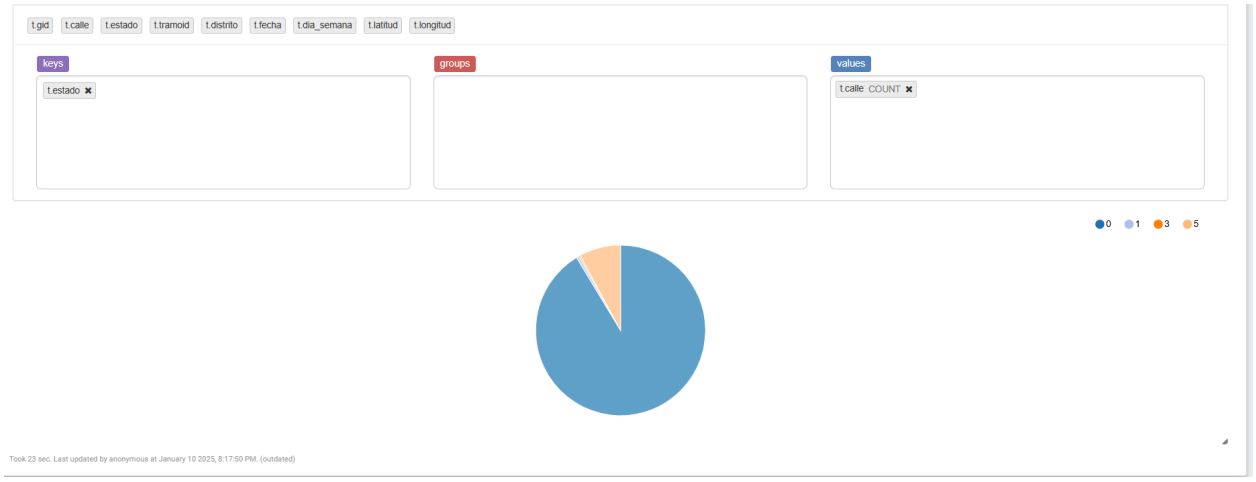
Took 23 sec. Last updated by anonymous at January 10 2025, 8:17:50 PM. (outdated)

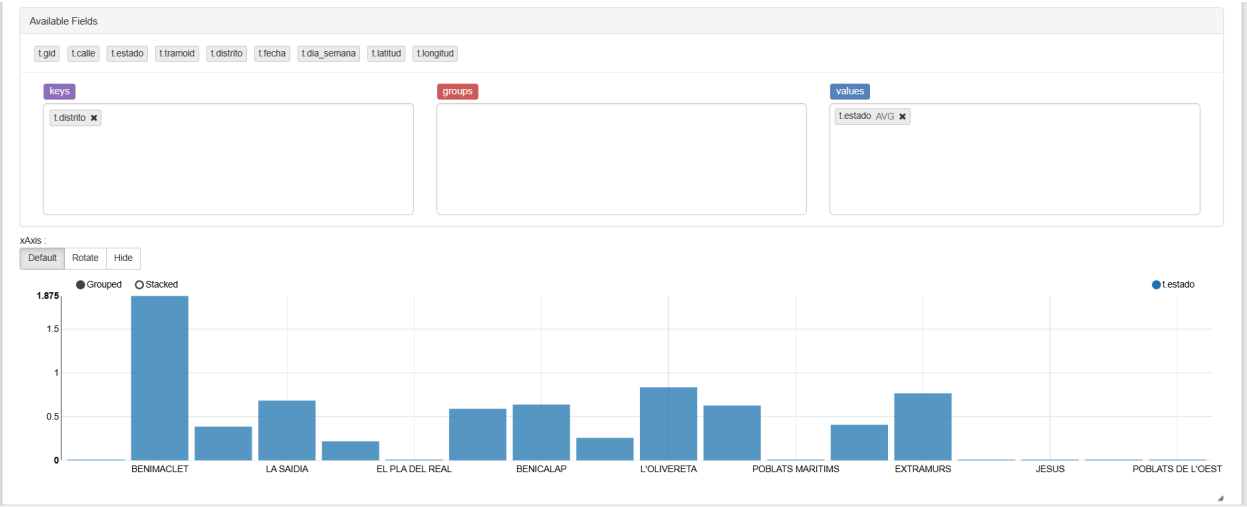
Visualización

Para la visualización yo he optado por utilizar las capacidades gráficas de zeppelin ya que son sorprendentemente variadas y personalizables pero además, al extraer los datos mediante consultas HiveQL de las tablas Hive tenemos acceso a crear gráficos sobre los datos que queramos y les podemos aplicar toda clase de filtros, agregaciones, u operaciones como columnas calculadas que queramos, el límite a la exploración y visualización de datos tan solo es limitado por nuestra imaginación. Además de esta manera no tenemos motor de indexación pero es que tenemos algo mejor ya que Hive está pensado para optimizar el rendimiento en sus consultas y podemos realizar particiones con columnas que tengan distribuciones de sus valores de manera más o menos equitativas a lo largo de las filas para realizar particiones igual de extensas pudiendo paralelizar el cálculo de las consultas aprovechando el hecho de que hemos salido del clúster y seguimos teniendo varios nodos.

transito_valencia_stream

A continuación muestro unas cuantas propuestas gráficas junto con el análisis correspondiente del tráfico de la ciudad extraído de la consulta HiveQL del apartado anterior sobre transito_valencia_stream:

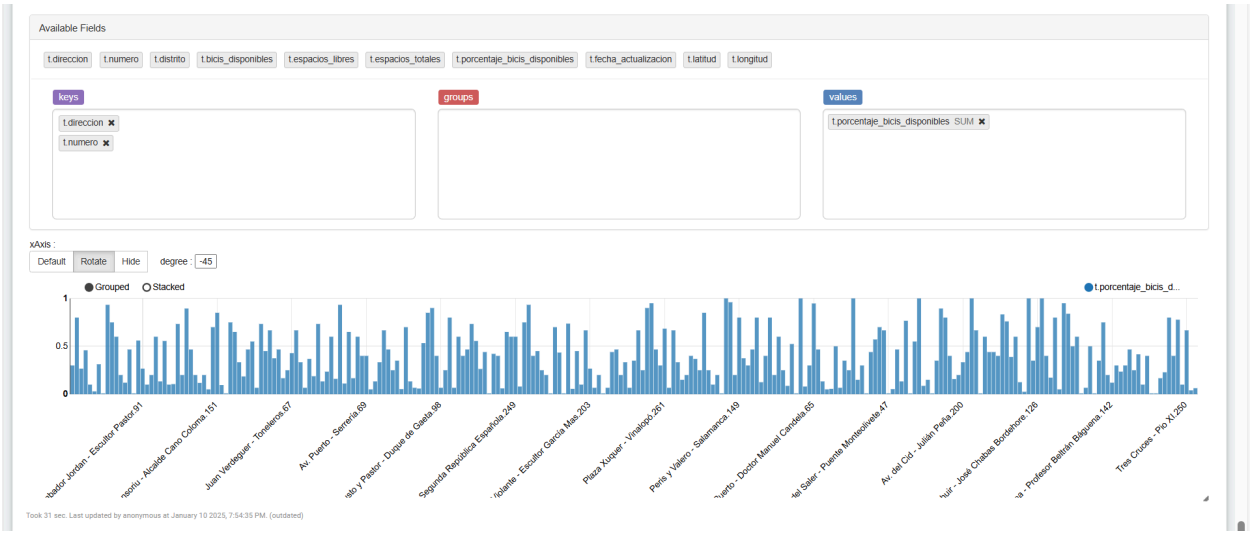


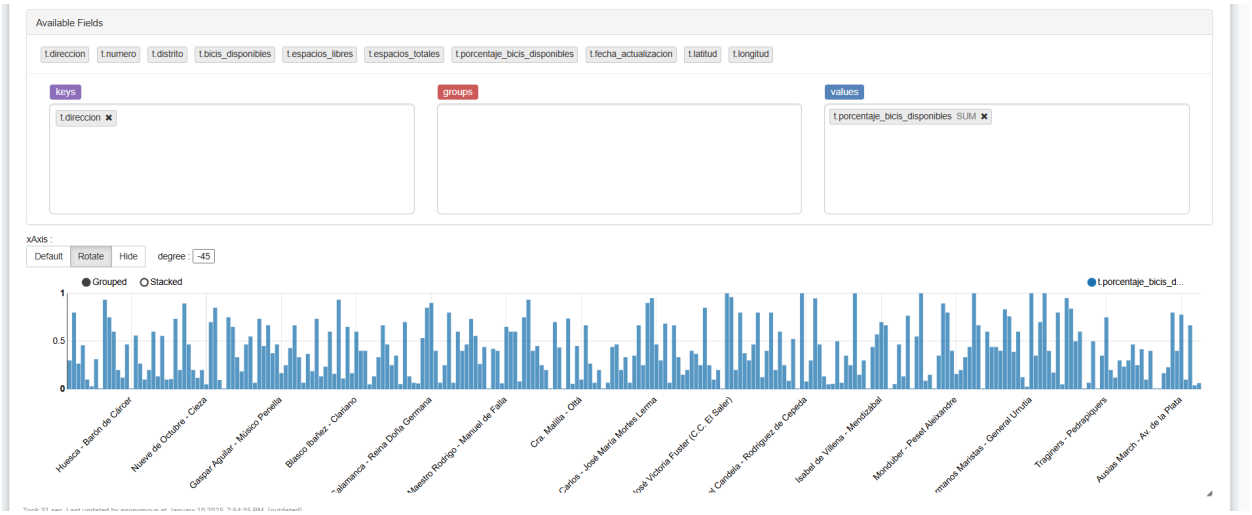


Podemos extraer varias características de los datos de tráfico de la Ciudad de Valencia:

- La mayor parte de las calles están despejadas
- Curiosamente, las calles no despejadas están casi todas directamente colapsadas, las calles pasan del estado 0 a 5, seguramente las que están en estados intermedios van están fluctuando a hacia los estados límite. Esto puede puede deberse a varios factores. En primer lugar, existe una tendencia bien conocida en todo el país y en Europa a la concentración de cada vez más población en ciudades. Lo cual genera, entre otras cosas, el colapso de los sistemas de transporte, el tráfico de vehículos es especialmente vulnerable a la congestión mientras que el transporte público suele ser más óptimo. Cabe destacar que la hora a la que estamos sacando datos son las 15:00, es muy probable que sobre las 8:00 de un día laborable gran parte de la ciudad esté colapsada. Quizás, y en contraposición a la suposición anterior, esto también podría ser indicativo de que los sensores son demasiado sensibles al tráfico y asignan el máximo nivel de concentración con un umbral demasiado bajo ya que los niveles intermedios parecen no ser útiles.
- Por último, como podemos ver en el segundo gráfico, la agrupación de la media del estado por distrito no parece ser muy informadora ya que hay ciertos distritos céntricos que tienen medias muy bajas como el Pla del Real y otros que no como Benimaclet. Es probable que el tráfico se comporte de tal manera que las calles principales que actúan como arterias de la ciudad y las que desembocan en ella sean las que se colapsan pero el resto de calles secundarias no lleven muchos coches.

transito_valenbisi:
Pasamos a analizar los datos del sistema público de bicicletas ValenBisi en la ciudad.

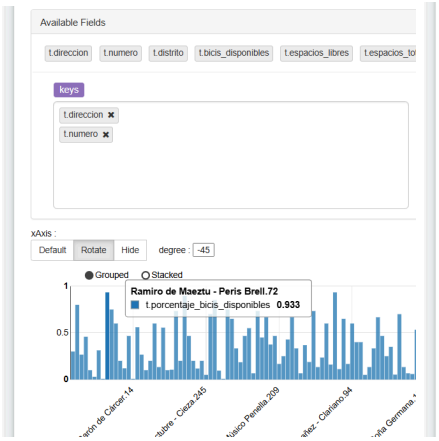


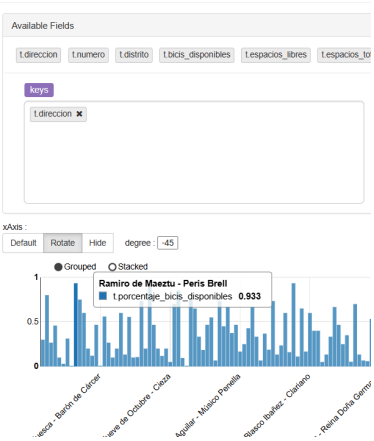


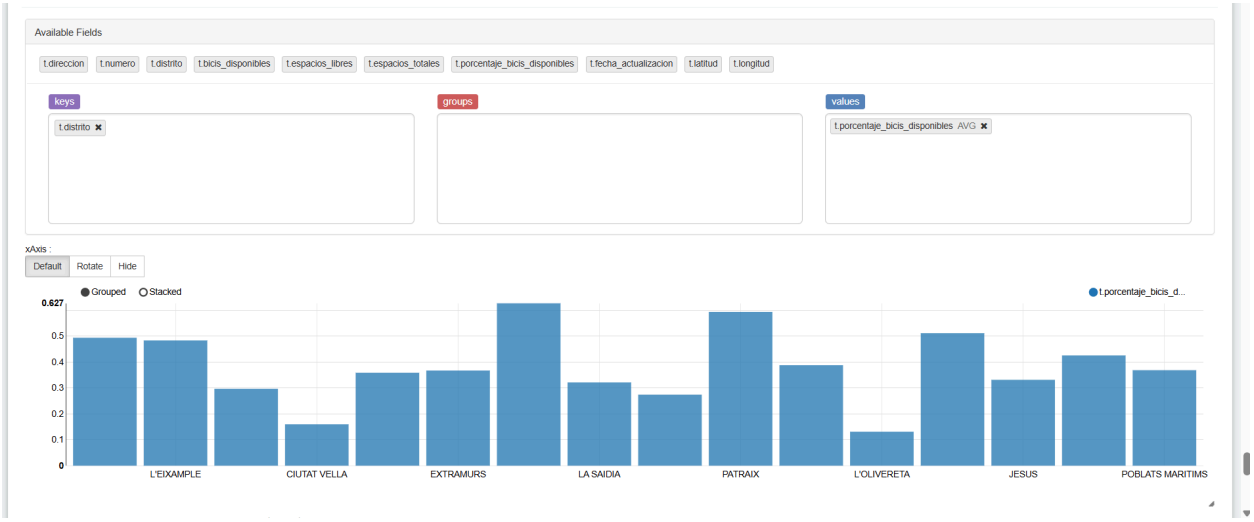
Para empezar, este sistema parece ser mucho más flexible en cuanto a soportar gran demanda de uso al contrario que las carreteras. Lo podemos inferir del hecho de que el porcentaje de bicis disponibles se sitúa en muchos valores distintos dentro de la la escala entre 0 y 1 posible.

También podemos descubrir relaciones entre las variables, en este caso, fijémonos que si volvemos a hacer el mismo gráfico eliminando del eje x los números de calle, el gráfico es exactamente igual. Esto significa que ningún puesto Valenbisi se pone en diferentes números de una sola calle, sino que cada punto ValenBisi está en una calle distinta. Los nombres son distintos entre el primer caso y el segundo porque los nombres del eje x se asignan de manera aleatoria en un subconjunto de los datos pero la forma del gráfico en la misma y podemos ver comprobar ya que podemos poner el cursor encima de las barras del gráfico para ver su

nombre:







Cuando miramos este segundo gráfico podemos ver cómo, en el caso de las bicis, el uso de estas sí que parecen estar correlacionadas con el hecho de que el distrito sea más céntrico o periférico.
Además en el caso de las bicis podemos ver cómo ninguno de los distritos sufre una escasez muy acusada de bicis disponibles descartando una inversión insuficiente en la infraestructura de ValenBisi en algunos distritos en caso de pasar.

meteorologia_stream

En el caso de la meteorología, debido a que tan solo tenemos una fila a la vez con los datos de la última hora registrada no es posible hacer gráficos en stream.

He presentado algunos ejemplos de visualización basados en el último flujo de datos recibidos, es decir, visualización del streaming de datos en tiempo real. Sin embargo, gracias a la acumulación de series temporales junto con una columna que nos dice cuándo han sido

tomados los datos en las tablas acabadas en “_stream” la infraestructura nos permite estudiar la evolución temporal de un distrito de interés o calle concreta, o de todos los distritos a la vez. Podemos ver si existe una diferencia de congestión y el nivel de esta en fines de semana respecto de días laborales, si afecta a zonas distintas, etc…

Ventajas del Data Lake respecto de un Data Warehouse

Por último, me gustaría enfatizar varias de las ventajas que este sistema tiene respecto de si utilizamos una arquitectura tipo data warehouse tradicional.
En primer lugar, la posibilidad de realizar un streaming contínuo de datos y recibirlo para nutrir nuestras tablas que podemos consultar en cualquier momento mediante una consulta ya preparada y que solo debemos “refreshear” cuando queramos saber la situación actual es una función que perderemos en un data warehouse tradicional ya que este no tiene procesos continuos de ingesta y almacenamiento de datos.

En segundo lugar, las capacidades de almacenamiento de un data warehouse no cuentan con escalabilidad horizontal y no podríamos guardar información acumulada de años recibida con una frecuencia de tres minutos ya que excedería los límites de almacenamiento tradicional.

En tercer lugar, esta infraestructura admite como fuente de información cualquier tipo de datos estructurados o semiestructurados, los datos de la meteorología, por ejemplo, son un JSON que se extrae de la URL de otro JSON. Estos formatos nos se pueden utilizar para nutrir un datawarehouse.

Por último, todo el proceso está automatizado dentro de Nifi y Spark y, si bien es posible automatizar procesos de ETL fuera de big data, estos pueden no ser robustos debido a que no están preparados para soportar cargas de ingestión variables con picos de demanda que requieren mayor poder computacional para ingerir y transformar todos los datos sin colapsar el sistema ni provocar retrasos ya que tanto Spark como Nifi pueden funcionar en varios nodos permiten la computación distribuida que puede agregar más nodos a funcionar si hacen falta en un momento a dado que soporten la carga extra.

Bibliografia:

Overview - Spark 3.5.4 Documentation

Apache NiFi

AEMET OpenData

Portal de Datos Abiertos del Ayuntamiento de València — Portal de l'Ajuntament de la ciutat de València

Apache Hive