PRÁCTICA

Sistemas Avanzados de Integración de Información

Hito 4

Creación de bases de datos relacionales

GRUPO 04

-

Tomás Carretero Alarcón
Alejandro Miñambres Mateos
Pablo Martín de Benito

-

Índice

El objetivo de la práctica es una búsqueda de información en el ámbito de la meteorología en España con el propósito de integrar más adelante dicha información en una base de datos.

Con el fin de lograr esto hemos obtenido las fuentes de datos que se explicarán a continuación y que se utilizarán para obtener una fuente global centralizada que nos ayudará a obtener la información que nos interese para un futuro.

1. Fuentes utilizadas

Fuente 1:

Almacena información sobre los datos climatológicos mensuales en Asturias en 2020.
Esta información está en aemet.
https://opendata.aemet.es/centrodedescargas/productosAEMET?

Esta fuente contiene:

> ***climatologiaAsturias****(fecha, indicativo, nombre, provincia, tm_mes, tm_max, tm_min, np_010 , n_nie, evap, p_mes)*

Descripción del esquema:

- **fecha**: Fecha del día, con el formato AAAA-MM-DD
- **indicativo**: Código que identifica una estación meteorológica.
- **nombre**: Nombre (ubicación) de la estación.
- **provincia**: Provincia de la estación.
- ***tm_mes***: Temperatura media mensual
- **tm_min**: Temperatura media mensual de las mínimas.
- ***tm_max***: Temperatura media mensual de las máximas .
- ***np_010***: Número de días de precipitación mayor o igual que 1*mm.*
- ***n_nie****:* Número de días de nieve en el mes.
- ***evap***: Evaporación total mensual (*mm*).
- **p_mes:** Precipitación total mensual (mm)

Fuente 2:

Almacena información sobre los datos semanales de los embalses peninsulares con capacidad superior a 5 hm^3, desde 1988 hasta 2022.
Esta información está en miteco.gob.es:
https://www.miteco.gob.es/es/agua/temas/evaluacion-de-los-recursos-hidricos/boletin-hidrologico.html

datosEmbalses*(embalse_nombre, fecha, agua_total, agua_actual)*

Descripción del esquema:

- **embalse_nombre**: Nombre del embalse.
- **fecha**: Fecha del dato.(DD-MM-AAAA)
- **agua_total**: Capacidad total a nivel máximo normal.
- **agua_actual**: Reserva de agua en la fecha actual.

Fuente 3:

Datos generales obtenidos de la wikipedia sobre los embalses y la provincia donde se encuentran.
La siguiente información ha sido obtenida de la wikipedia

ambitoEmbalse*(provincia, nombre_embalse)*

- **provincia**: identificada la provincia a la que pertenece el embalse.
- **nombre_embalse**, completo del embalse.

La clave primaria está formada por los dos atributos, ya que hay embalses en varias provincias.

Fuente 4:

Contiene datos mensuales generales de asturias desde el 2020 hasta la actualidad
Climatología (sadei.es)

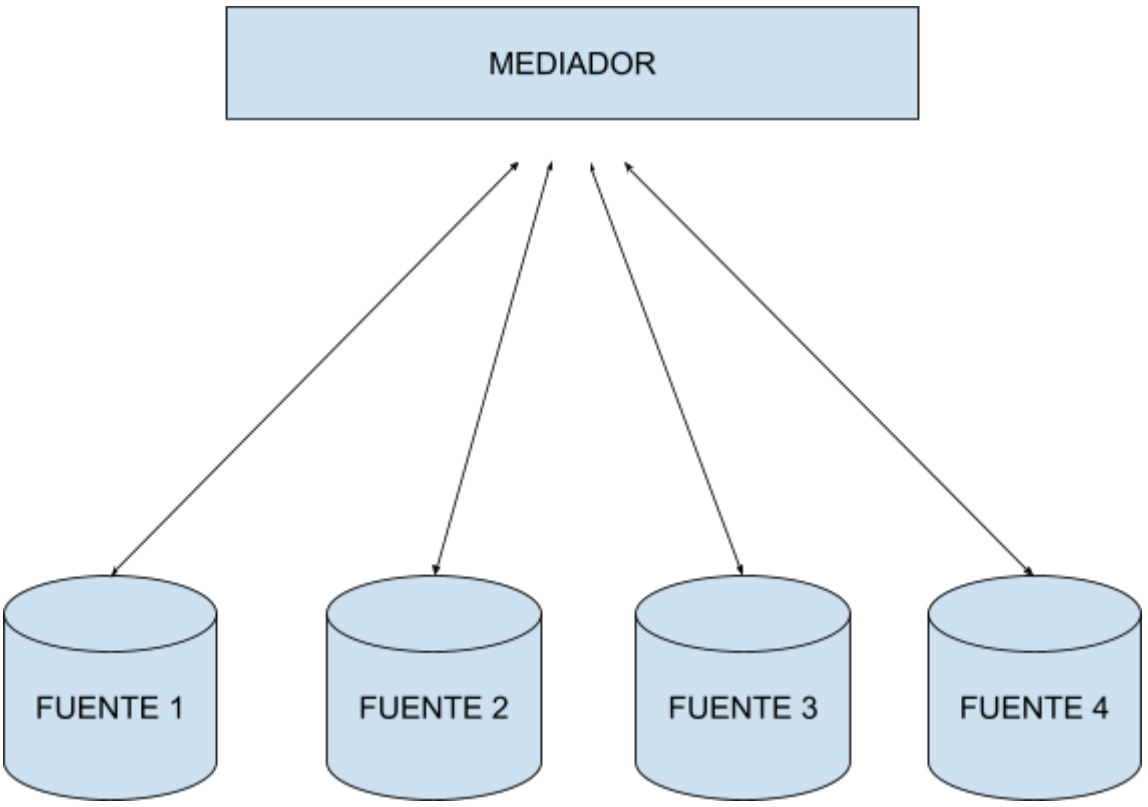
temperaturaAsturias*(fecha, estacionClimatologica, tmed, tmed_max, tmed_min, precip)*

- ***fecha***: Fecha del día, con el formato AAAA-MM-DD.
- ***estacionClimatologica*****:** nombre de la estación.
- ***tmed*****:** Temperatura media mensual.
- ***tmed_max*****:** Temperatura media mensual de las máximas.
- ***tmed_min*****:** Temperatura media mensual de las mínimas.
- ***precip*****:** Acumulado de la precipitación mensual (*mm*).

2. Mediador

Un mediador es un componente que actúa como intermediario entre los usuarios y las múltiples fuentes de datos subyacentes. Su función principal es unificar y simplificar el acceso a la información almacenada en diferentes bases de datos o sistemas heterogéneos.

Se seguirá el siguiente esquema:



Construimos el mediador con la siguiente información:

Resumen mensual de las temperaturas en las diferentes localizaciones de Asturias.

temperaturaAst(*fecha, nombre, tmed, tmed_max, tmed_min****)***

Precipitaciones mensuales en Asturias.

precipitacionesAst(*fecha, nombre, precip****)***

Número de días de precipitación y de nieve junto con el nivel del embalse en la comunidad de Asturias y los niveles de evaporación mensuales.

embalsesAst*(fecha, nombre, nombre_embalse, provincia, agua_actual, agua_total, np_010, n_nie, evap)*

*nombre = nombre estación donde se mide.

*nombre embalse = nombre del embalse.

3. Consultas

- *Fecha en las que la temperatura media supera los 15 ºC en las diferentes zonas de Asturias.*
- *Fechas y zonas en las que las precipitaciones son abundantes. (Mayores de 7 mm).*
- *Fecha, nombre y agua de los embalses en los meses que llovió más de 10 días y un acumulado mayor de 10mm.*

Dichas consultas quedarían representadas mediante el lenguaje SQL de la siguiente manera:

➔ ***Consulta 1:***

```
SELECT T.fecha, T.nombre, T.tmed
FROM temperaturaAst T
WHERE T.tmed > 15;
```

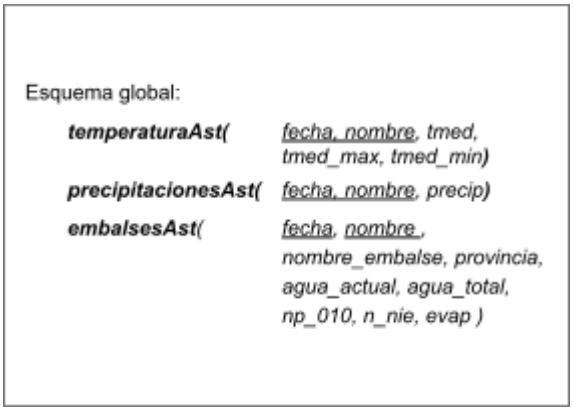
➔ ***Consulta 2:***

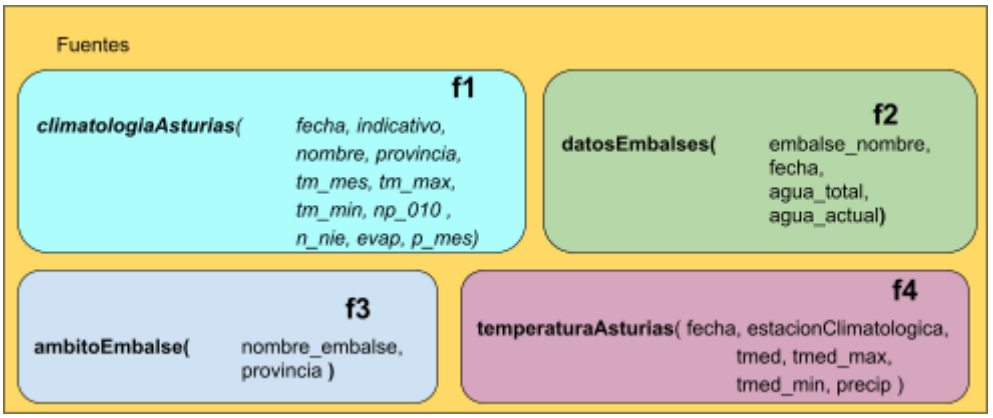
```
SELECT P.fecha, P.nombre, P.precip
FROM precipitacionesAst P
WHERE P.precip > 7;
```

➔ ***Consulta 3:***

```
SELECT P.fecha, E.nombreEmbalse, E.agua_actual
FROM precipitacionesAst P, embalsesAst E
WHERE E.np_010 > 9 and P.precip > 9
        E.fecha = P.fecha and E.nombre = P.nombre;
```

4. Descripciones GAV y LAV de las consultas.





4.1 Mappings GAV:

Tenemos cuatro fuentes en donde cada una tiene una sola tabla, también tenemos un mediador con tres tablas.

Vamos a utilizar el mediador y cada una de las seis tablas ver cómo la podemos obtener a partir de las fuentes de datos que tenemos.

Esquema 1:
Descripción de ***temperaturaAst*** como un conjunto de vistas sobre las fuentes de datos:

$$temperaturaAst(fecha,\ nombre, tmed, tmedMax,\ tmedMin)$$
$$\supseteq\ f4.temperaturaAsturias(fecha,\ nombre,\ tmed, tmedMax, tmedMin, precip)$$
$$\supseteq\ f1.climatologiaAsturias(fecha,\ indicativo, nombre,\ provincia\ tmed, tmedMax, tmedMin, np010, nnie, evap, pmes)$$

Esquema 2:

Descripción de ***precipitacionesAst*** como un conjunto de vistas sobre las fuentes de datos:

$$precipitacionesAst(fecha,\ nombre,\ precip)$$
$$\supseteq\ f4.temperaturaAsturias(fecha,\ nombre, tmed, tmedMax, tmedMin, precip)$$
$$\supseteq\ f1.climatologiaAsturias(fecha,\ indicativo, nombre,\ provincia, tmed, tmedMax, tmedMin, np010, nnie, evap, precip)$$

Esquema 3:

Descripción de ***embalsesAst*** como un conjunto de vistas sobre las fuentes de datos:

$$embalsesAst(fecha,\ nombre,\ nombreEmbalse,\ provincia,\ aguaActual,\ aguaTotal,\ np010, nNie, evap)$$
$$\supseteq$$
$$f1.climatologiaAsturias(\ fecha, indicativo, nombre, provincia, tmMes, tmMax, tmMin, np010, nNie, evap,\ prec)$$
$$,$$
$$f2.datosEmbalses(nombreEmbalse, fecha, aguaTotal, aguaActual)$$
$$,$$
$$f3.ambitoEmbalse(nombreEmbalse, provincia)$$

Consultas:

Desdoblamiento de las consultas usando los mapping GAV.

➔ *Consulta 1:*

```
SELECT T.fecha, T.nombre, T.tmed
FROM temperaturaAst T
WHERE T.tmed > 15;
```

$$q1(fecha,\ nombre, tmed) :-\ temperaturaAst(fecha,\ nombre,\ tmed,\ tmedMax,\ tmedMin),\ tmed\ >\ 15$$

Reescribimos la consulta.

$$q1'(fecha,\ nombre, tmed) :-\ f4.temperaturaAsturias(fecha,\ nombre, tmed, tmedMax, tmedMin, precip),\ tmed\ >\ 15$$

$$q1''(fecha,\ nombre, tmed) :-\ f1.climatologiaAsturias(fecha, indicativo,\ nombre, tmed, tmedMax, tmedMin, np010, nnie, evap\ , precip),\ tmed\ >\ 15$$

➔ ***Consulta 2:***

```
SELECT P.fecha, P.nombre, P.precip
FROM precipitacionesAst P
WHERE P.precip > 7;
```

$q2(fecha,\ nombre,\ precip) :-\ precipitacionesAst(fecha,\ nombre,\ precip)$

Reescribimos la consulta.

$$q2'(fecha,\ nombre,\ precip) :-\ f4.temperatura\ Asturias(fecha, nombre, tmed, tmedMax, tmedMin, precip)$$
$$precip\ >\ 7$$
$$q2''(fecha,\ nombre,\ precip) :- f1.climatologiaAsturias(fecha, indicativo,\ nombre, tmed,$$
$$tmedMax, tmedMin, np010, nnie, evap\ , precip),\ precip\ >\ 7$$

➔ ***Consulta 3:***

```
SELECT P.fecha, E.nombreEmbalse, E.agua_actual
FROM precipitacionesAst P, embalsesAst E
WHERE E.np_010 > 9 and P.precip > 10 ;
```

$$q3(fecha,\ nombreEmbalse,\ aguaActual) :-\ precipitacionesAst(fecha,\ nombre,\ precip),$$
$$embalsesAst(fecha,\ nombre,\ nombreEmbalse,\ provincia,$$
$$aguaActual,\ aguaTotal, np010, nNie, evap),$$
$$np010\ >\ 9,\ precip\ >\ 10$$

$$q3'(fecha,\ nombreEmbalse,\ aguaActual) :-$$
$$f4.temperaturaAsturias(fecha,\ nombre, tmed, tmedMax,$$
$$tmedMin, precip),$$
$$f1.climatologiaAsturias(\ fecha, indicativo, nombre,$$
$$provincia, tmMes, tmMax, tmMin, np010, nNie, evap,\ prec),$$
$$f3.ambitoEmbalse(nombreEmbalse, provincia),$$
$$f2.datosEmbalses(nombreEmbalse, fecha, aguaTotal,$$
$$aguaActual),\ np010\ >\ 9,\ precip\ >\ 10$$

$$q3''(fecha,\ nombreEmbalse,\ aguaActual) :-$$
$$f1.climatologiaAsturias(fecha,\ indicativo1, nombre,\ provincia1,$$
$$tmed1, tmedMax1, tmedMin1, np011, nnie1, evap1, precip1),$$
$$f1.climatologiaAsturias(fecha,\ indicativo, nombre,\ provincia,$$
$$tmed, tmedMax, tmedMin, np010, nnie, evap, precip),$$
$$f3.ambitoEmbalse(nombreEmbalse, provincia),$$
$$f2.datosEmbalses(nombreEmbalse, fecha, aguaTotal,$$
$$aguaActual)\ ,\ np010\ >\ 9,\ precip\ >\ 10$$

*Hay que poner diferentes atributos en el primer f1, porque las variables vienen de otra tabla del mediador, por lo que solo queremos que coincida la fecha y el nombre, con el de la tabla embalsesAst.

4.2 Mappings LAV:

Primero, obtenemos los mapping LAV del mediador que corresponden a las fuentes.

$$m1:\ f1.climatologiaAsturias(\ fecha, indicativo, nombre, provincia, tmMes, tmMax, tmMin, np010, nNie, evap,\ precip) \subseteq temperaturaAst(\ fecha,\ nombre,\ tmMes,\ tmMax,\ tmMin), precipitacionesAst(\ fecha,\ nombre,\ precip), embalsesAst(\ fecha,\ nombre,\ nombreEmbalse, provincia, aguaActual, np010, nNie, evap)$$

$$m2:\ f2.datosEmbalses(nombreEmbalse, fecha, aguaTotal, aguaActual\ ) \subseteq embalsesAst(fecha, nombre,\ nombreEmbalse,\ provincia,\ aguaActual, aguaTotal,\ np010, nNie, evap)$$

$$m3:\ f3.\ ambitoEmbalse(\ nombre\_embalse,\ provincia\ ) \subseteq embalsesAst(fecha, nombre,\ \ nombre\_embalse,\ provincia, aguaActual, aguaTotal,\ np010, nNie, evap)$$

$$m4:\ f4.temperaturaAsturias(fecha,\ nombre, tmed, tmMax, tmedMin, precip) \subseteq temperaturaAst(\ fecha,\ nombre,\ tmMes,\ tmMax,\ tmMin), precipitacionesAst(\ fecha,\ nombre,\ precip)$$

Una vez obtenidos los mapping LAV, vamos a reescribir las consultas haciendo uso del algoritmo basado en buckets, que consiste en que para cada uno de los átomos de la consulta, le construimos un contenedor con las vistas relevantes.

Reescritura de la consulta usando los mapping LAV.

➔ *Consulta 1:*

$$q1(fecha,\ nombre,\ tmed) :-\ temperaturaAst(fecha,\ nombre,\ tmed,\ tmedMax,\ tmedMin),\ tmed\ >\ 15$$

temperaturaAst(fecha, nombre, tmed, tmedMax, tmedMin)
f1.climatologiaAsturias(fecha,indicativo,nombre,provincia,tmed,tmMax,tmMin, np010,nNie,evap, precip)
f4.temperaturaAsturias(fecha, nombre,tmed,tmMax,tmedMin,precip)

Las variables cabecera son ***fecha, nombre y tmed***.
Obtenemos las reescrituras posibles:

$$q1_1(fecha,\ nombre,\ tmed) :-$$
f1.climatologiaAsturias(fecha,indicativo,nombre,provincia, tmMes,tmMax, tmMin, np010,nNie,evap, precip),
$$tmed\ >\ 15$$

$$q1_2(fecha,\ nombre,\ tmed) :-$$
f4.temperaturaAsturias(fecha, nombre,tmed,tmMax,tmedMin,precip),
$$tmed\ >\ 15$$

Puesto que sólo tienen una fuente y no existen reescrituras más simples, terminamos.

➔ *Consulta 2:*

$q2(fecha,\ nombre,\ precip) :-\ precipitacionesAst(fecha,\ nombre,\ precip),\ precip\ >\ 7$

precipitacionesAst(fecha, nombre, precip)
$f1.climatologiaAsturias(fecha,\ indicativo,\ nombre,\ provincia,\ tmMes,\ tmMax, tmMin,\ np010,\ nNie,\ evap,\ precip)$
$f4.temperaturaAsturias(fecha,\ nombre,\ tmed, tmedMax,\ tmedMin,\ precip)$

Las variables cabecera son ***fecha, nombre y precip***.
Las únicas reescrituras posibles son:

$q2(fecha,\ nombre,\ precip) :-\ f1.climatologiasAsturias(fecha,\ indicativo,\ nombre,\ provincia,\ tmMes,\ tmMax, tmMin, np010, nNie, evap, precip),\ precip\ >\ 7$

$q2'(fecha,\ nombre,\ precip) :-\ f4.temperaturaAsturias(fecha, nombre, tmed, tmedMax, tmedMin,\ precip),\ precip\ >\ 7$

➔ *Consulta 3:*

$$q3(fecha, nombreE, aguaActual) :- \; precipitacionesAst(fecha, nombre, precip),$$
$$embalsesAst(fecha, nombreE, provincia,$$
$$aguaActual, aguaTotal, np010, nNie, evap),$$
$$np010 > 9, precip > 10, provincia = "Asturias"$$

precipitacionesAst(fecha, nombre, precip)	***embalsesAst(fecha, nombre, nombreE, provincia, aguaActual,aguaTotal, np010, nNie, evap),***
f1.climatologiaAsturias(fecha,indicativo, nombre,provincia, tmMes,tmMax,. tmMin,np010, nNie,evap, precip)	$f2.datosEmbalses(nombreE, fecha, aguaTotal, aguaActual)$
f4.temperaturaAsturias(fecha, nombre, tmed,tmMax, tmedMin,precip)	

Las variables cabecera son ***fecha, nombreE, aguaActual***.
Las otras fuentes no entran porque no cumplen el algoritmo.
Obtenemos las reescrituras operando los diferentes átomos con el producto cartesiano.

$$q3_1(fecha, nombreE, aguaActual): -$$
$$f1.climatologiasAsturias(fecha, indicativo, nombre, provincia,$$
$$tmMes, tmMax, tmMin, np010, nNie, evap, precio),$$
$$f2.datosEmbalses(nombreE, fecha, aguaActual, aguaTotal)$$

$$q3_2(fecha, nombreE, aguaActual): -$$
$$f4.temperaturaAsturias(fecha, nombre, tmed, tmMax, tmedMin, precip),$$
$$f2.datosEmbalses(nombreE, fecha, aguaActual, aguaTotal)$$

Existirían reescrituras alternativas pero las variables cabecera no aparecen en dicha reescritura por lo que no son válidas.

A continuación, desdoblamos las consultas para comprobar si son válidas.

$$q3_1(fecha, nombreE, aguaActual): -$$
$$temperaturaAst(fecha, nombre, tmMes, tmMax, tmMin),$$
$$precipitacionesAst(fecha, nombre, precip),$$
$$embalsesAst(fecha, nombre, nombreE, provincia, aguaActual, np010, nNie, evap),$$
$$embalsesAst(fecha, nombre, nombreE, provincia, aguaActual, aguaTotal, np010, nNie, evap).$$

$$q3_2(fecha, nombreE, aguaActual): -$$
$$temperaturaAst(fecha, nombre, tmMes, tmMax, tmMin),$$
$$precipitacionesAst(fecha, nombre, precip),$$
$$embalsesAst(fecha, nombre, nombreE, provincia, aguaActual, np010, nNie, evap).$$

5. Acceso a las fuentes

Accederemos a las diferentes fuentes usando diferentes técnicas aprendidas.

***Acceso a la fuente 1*:**

En este caso la página de aemet nos proporciona una API, la cual usaremos para obtener los datos que necesitamos..

Vemos que hay varios problemas como que el campo *evap* no aparece siempre y que la *fecha* no está en el formato que habíamos previsto, por lo que lo añadimos como string y posteriormente cambiaremos su formato.

Además cuando buscamos información sobre estos datos mostraba los campos *provincia* y *nombre*, pero estos no aparecen aquí.
Como en este caso los datos obtenidos son de la misma estación de Asturias lo añadimos manualmente en el programa.

```
#Estos datos deberían de aparecer pero no aparecen por lo que los metemos a mano
provincia = 'Asturias'
nombre = 'Aeropuerto'

for dato in datos_api:
    fecha = dato['fecha']
    indicativo = dato['indicativo']
    tm_mes = dato['tm_mes']
    tm_max = dato['tm_max']
    tm_min = dato['tm_min']
    p_mes = dato['p_mes']
    p_max = dato['p_max']
    np_010 = dato['np_010']
    n_nie = dato['n_nie']
    if 'evap' in dato:
        evap = dato['evap']
    else:
        evap = None
```

Al ejecutar el programa diseñado, los resultados que insertamos y nos muestra, son correctos.

fecha	indicativo	nombre	provincia	tm_mes	tm_max	tm_min	p_mes	p_max	np_010	n_nie	evap
2020-1	1212E	Aeropuerto	Asturias	10.1	14	6.2	68.7	12.2(18)	13	0	642
2020-10	1212E	Aeropuerto	Asturias	14.8	18.5	11.1	237.4	61.4(01)	15	0	830
2020-11	1212E	Aeropuerto	Asturias	13.5	17.6	9.4	32.9	7.0(02)	6	0	548
2020-12	1212E	Aeropuerto	Asturias	10.3	13.6	6.9	396.9	50.2(27)	23	0	625
2020-13	1212E	Aeropuerto	Asturias	14.5	18.1	10.8	1251.8	61.4(01/oct)	132	0	NULL

Acceso a la fuente 2:

Esta fuente nos generaba los datos en un formato el cual no podíamos manejar, por esa razón transformamos los datos usando el programa R. Finalmente nos quedamos solo con los datos de 2020 hasta la actualidad, y además seleccionamos para tener solo un dato por mes, en vez de un dato a la semana.

Haciendo un programa adaptado a ese formato vamos obteniendo los datos

```
# Abrir el archivo en modo lectura
with open('DatosEmbalsesResum.txt', 'r',encoding='iso-8859-1') as archivo:
    # Leer todas las líneas del archivo
    lineas = archivo.readlines()

datos=[]

for linea in lineas:
    # Separar la línea por punto y coma
    valores = linea.strip().split(';')
    # Agregar los valores a la lista de datos
    datos.append(valores)
```

Había problemas en la notación decimal de los números por lo que había que cambiarlo para poder añadirlo a nuestra base de datos sql.

```
for fila in datos:
    embalse_nombre=fila[1]
    fecha=fila[2]
    agua_total = float(fila[3].replace(',', '.').strip('"'))
    agua_actual= float(fila[4].replace(',', '.').strip('"'))

    try:
        cursor.execute('''INSERT INTO datosEmbalse (embalse_nombre, fecha, agua_total, agua_actual)
                          VALUES (%s, %s, %s, %s)''', (embalse_nombre,fecha,agua_total,agua_actual))
        connection.commit()  # Commit the transaction
    except Exception as e:
        print("Error occurred:", e)
        connection.rollback()  # Rollback the transaction in case of error
```

Una vez ejecutado el código, los datos que tenemos son:

embalse_nombre	fecha	agua_total	agua_actual
"Alfilorios"	"19/03/2024"	8	6
"Alfilorios"	"22/02/2022"	8	7
"Alfilorios"	"23/02/2021"	8	8
"Alfilorios"	"24/11/2020"	8	6
"Alfilorios"	"25/01/2022"	8	7
"Alfilorios"	"25/02/2020"	8	8

Destacar que pese a todas las simplificaciones de los datos explicadas anteriormente, este archivo seguía teniendo más de trece mil líneas, por esa razón para ganar velocidad en el mediador y consultas decidimos meter directamente solo los embalses de Asturias a modo de resumen.

Acceso a la fuente 3:

En este caso obtenemos los datos mediante scrapping. En nuestro caso, necesitábamos un conjunto de datos que relaciona las provincias de España y los embalses que había en cada una, sin embargo no encontramos fuentes que nos los proporcionaran de manera sencilla.

Por esa razón obtenemos los datos directamente de Wikipedia usando scrapping.
Usamos para ello la herramienta de *ParseHub* y obtuvimos un archivo *.csv* el cual ya solo había que adaptar a nuestra base de datos.

Uno de los problemas que nos surgió fue que la provincia no se encontraba como tal escrita, por lo tanto hicimos uso de expresiones regulares para obtener solamente la parte del texto que nos interese, en este caso, el nombre de la provincia.

```python
# Función para extraer el nombre de la provincia
def extraer_provincia(texto):
    # Buscar el patrón "Embalses de [provincia]" y extraer solo el nombre de la provincia
    match = re.search(r'Embalses de (.*)$', texto)
    if match:
        match2 = re.search(r'la .* de (.*)$',match.group(1))
        if match2:
            return match2.group(1)
        else:
            return match.group(1)
    else:
        return None
```

También había que modificar el nombre del embalse. Para ello se usó otra expresión regular similar a esa.
Por esa razón también decidimos no usar la sentencia *LOAD DATA INFILE*, ya que nos complicaba el hecho de trabajar con las expresiones regulares.

Finalmente se introducen los datos en nuestra fuente de la siguiente manera:

```python
# Abrir el archivo CSV y leer los datos
with open('embalsesProvincia.csv', 'r') as csv_file:
    csv_reader = csv.reader(csv_file)
    next(csv_reader)  # Si hay encabezados, omítelos
    for row in csv_reader:
        # Expresion regular para obtener la provincia
        prov = extraer_provincia(row[0])
        cursor.execute('''INSERT INTO ambitoEmbalse (provincia, nombreEmbalse)
                          VALUES (%s, %s)''', (prov, row[2]))
```

Ejecutando dicho programa, los datos se encontrarán almacenados en nuestra base de datos de la manera que se muestra a continuación.

provincia	nombreEmbalse
La Coruña	Abegondo-Cecebre
Palencia	Aguilar
Lérida	Aiguamòg
Badajoz	Alange
Lérida	Albagés
Cáceres	Alcántara

Acceso a la fuente 4:

Para la cuarta fuente nosotros hemos utilizado un método de acceso a los datos mediante el punto SparQL.
Nuestra fuente sería datos.gob.es que cuenta con un punto sparql que utilizaremos.

Para ello, a través del código, tenemos que acceder al punto SparQL, esto se hace utilizando dicha URL que te proporciona la página.
Para ello nosotros hemos utilizado una librería de python ***SPARQLWrapper***.

```
endpoint_url = "http://datos.gob.es/virtuoso/sparql"

# Crear un objeto SPARQLWrapper para el punto SPARQL
sparql = SPARQLWrapper(endpoint_url)
```

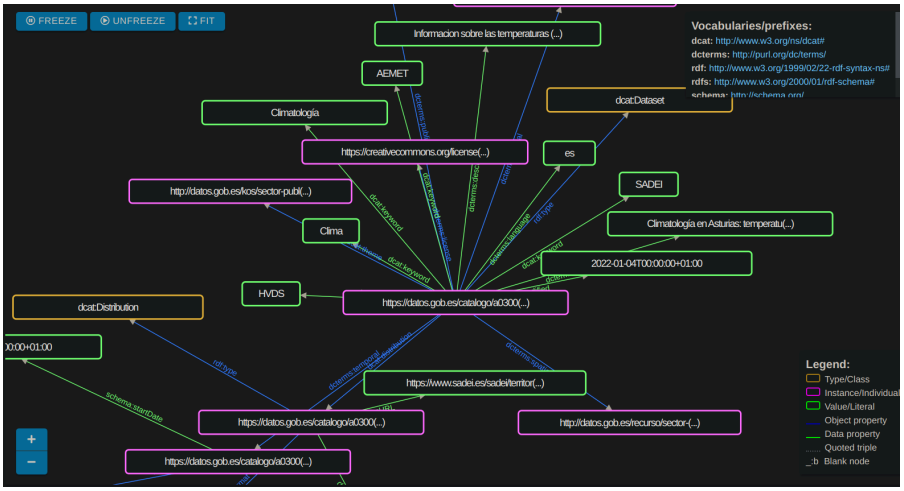
De esta manera ya tendríamos configurado el punto SparQL por lo que habría que establecerle la consulta que se quiere hacer.

```
# Consulta SPARQL
query = """
    select distinct ?dataset ?url where {
    ?dataset a <http://www.w3.org/ns/dcat#Dataset> .
    ?dataset <http://purl.org/dc/terms/title> "Climatología en Asturias: temperaturas, precipitaciones y horas de sol"@es .
    ?dataset <http://www.w3.org/ns/dcat#distribution> ?distribution .
    ?distribution <http://www.w3.org/ns/dcat#accessURL> ?url .
    }
"""
```

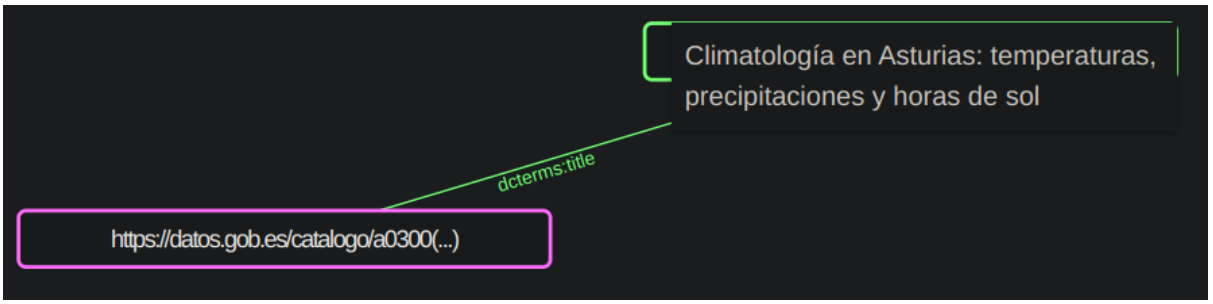
Esta consulta lo que nos permite es obtener el dataset que estamos buscando, para ello accedemos al catálogo de datos que tiene *datos.gob*.
Seguido de ello, una vez tenemos todos los datasets, del catálogo, obtendremos el que queremos a través del título, en nuestro caso era *Climatología en Asturias: temperaturas, precipitaciones y horas de sol*.

Este título lo podemos obtener a partir de la red semántica del archivo y que podemos descargar.

Así lo visualizamos con isSemantic:



Así más en concreto, obtenemos el título del dataset, en este caso…



Una vez obtenemos el dataset en la consulta, obtenemos el recurso que nos permitirá descargarlo.
Para ello hay que acceder al campo *Distribution* en donde se encuentra el *accessURL*, que como su nombre indica, nos devuelve el URL de acceso al recurso.

Ya con la consulta construida, se la establecemos al conector y obtenemos los datos.

```
# Establecer la consulta SPARQL en el objeto SPARQLWrapper
sparql.setQuery(query)

# Los resultados deben ser devueltos en formato JSON
sparql.setReturnFormat(JSON)

try:
    # Realizar la consulta SPARQL
    results = sparql.query().convert()
    return results

except Exception as e:
    print(f"Error occurred: {e}")
```

Para insertar los datos, como un dataset como los anteriores ya insertados, recorremos el csv de manera que cada fila la insertamos con un bucle.

```
# Abrir el archivo CSV y leer los datos
with open('f4.csv', 'r') as csv_file:
    csv_reader = csv.reader(csv_file)
    next(csv_reader)  # Si hay encabezados, omítelos
    for row in csv_reader:
        fecha = row[0]
        estacion = row[1]
        tmed = row[2]
        tmed_max = row[3]
        tmed_min = row[4]
        precip = row[5]
        cursor.execute('''INSERT INTO temperaturaAsturias (fecha, estacionClimatologica, tmed,tmed_max, tmed_min, precip)
                          VALUES (%s, %s, %s, %s, %s, %s)''', (fecha, estacion, tmed, tmed_max, tmed_min, precip))

# Guardar los cambios y cerrar la conexión
connection.commit()
connection.close()
```

Cabe destacar que la fuente que hemos elegido desde el principio, ha sido un poco más complicado poder obtener los datos, y debido a la naturaleza de nuestra fuente, que son datos climatológicos sobre la comunidad de Asturias, y el propio recurso que nos devuelve la consulta SparQL, nos lleva a otra página interfaz donde tenemos que seleccionar los campos y filas que queremos obtener y después realizar otros pasos hasta finalmente poder descargar el conjunto de datos. .

Al ser una página regional (sadei.es), no cuenta ni con un api, ni otro punto SparQL para poder obtener los datos en sí.
Por lo que, debido al límite de tiempo, hemos decidido no cambiar la fuente, que causaría cambiar gran parte de la práctica obteniendo así por medio del punto SparQL, el enlace a la página de *sadei.es* y a partir de ahí, seleccionar los campos manualmente al no tener otro método de acceso.

Una vez descargado el archivo *.csv*, se sigue el proceso normal de inserción de los datos.

Mediador:

Para construir el mediador usaremos los mappings vistos anteriormente, de tal forma que vamos haciendo consultas sobre los datos de las fuentes y con los mappings se introducen los datos en las tablas del mediador.

Hay que destacar varias partes del código:

```
# Convertir fecha2_str al formato "2020-01"
fecha2 = datetime.strptime(fecha1.strip('"'), "%d/%m/%Y").strftime("%Y-%m")

año, mes = fecha.split('-')
if(len(mes))==1:
    mes = '0'+mes

fecha= f"{año}-{mes}"

if (provincia==provincia2 and fecha2 == fecha):
```

El primero es el formato de las fechas, el cual queremos que sea %Y-%m, sin embargo solo viene así en la fuente 4. Por lo que tenemos que hacer las dos transformaciones de la figura previa para poder compararlas y ver si son iguales.

Por otro lado tenemos fuentes que contienen datos similares, por lo tanto hay que hacer la integración.
Por defecto el mediador usará las precipitaciones y temperaturas de la fuente 4, ya que estas contienen las de todas las estaciones meteorológicas. Sin embargo si esta fuente falla usaremos la fuente 1, que contiene los mismos datos pero solo del Aeropuerto.

```
try:
    cursor4.execute("SELECT fecha, estacionClimatologica, tmed, tmed_max, tmed_min FROM temperaturaAsturias")
    datos_f4 = cursor4.fetchall()

    for fila in datos_f4:
        fecha, nombre, tmed, tmed_max, tmed_min = fila

        sql = "INSERT INTO temperaturaAst (fecha, nombre, tmed, tmed_max, tmed_min) VALUES (%s, %s, %s, %s, %s)"
        val = (fecha, nombre, tmed, tmed_max, tmed_min)
        try:
                cursor.execute(sql, val)
                connection.commit()  # Commit the transaction
        except Exception as e:
                print("Error occurred:", e)
                connection.rollback()  # Rollback the transaction in case of error
```

```
except Exception :
    print("Fuente 1")
    cursor1.execute("SELECT fecha, nombre, tm_mes, tm_max, tm_min FROM climatologiaAsturias")
    datos_f1 = cursor1.fetchall()

    for fila in datos_f1:
        fecha, nombre, tmed, tmed_max, tmed_min = fila
        # Convertir fecha al formato "2020-01"
        año, mes = fecha.split('-')
        if(len(mes))==1:
            mes = '0'+mes

        fecha= f"{año}-{mes}"


        sql = "INSERT INTO temperaturaAst (fecha, nombre, tmed, tmed_max, tmed_min) VALUES (%s, %s, %s, %s, %s)"
        val = (fecha, nombre, tmed, tmed_max, tmed_min)
        try:
            cursor.execute(sql, val)
            connection.commit()  # Commit the transaction
        except Exception as e:
            print("Error occurred:", e)
            connection.rollback()  # Rollback the transaction in case of error
```

De esa forma conseguimos que siempre estén cargados al menos los datos de la estación del aeropuerto, la cual es la que nos interesa para las consultas propuestas.

6. Aclaraciones:

En primer lugar, comentar que decidimos meter las fuentes originales en nuestras bases de datos (en la máquina virtual) y que el mediador use los datos desde ahí.
Esto es debido a:

- Datos inmutables. Algunas fuentes como la f1 que solo tiene datos de 2020, u otras como la fuente 3, que contiene información que no cambia a corto plazo.
- Gran cantidad de datos. Como en la fuente f2, que al estar accediendo constantemente a ella, llevaría un gran tiempo de cómputo tanto para el mediador, como para la interfaz. Por esa razón se decidió usar las fuentes de datos locales reduciendo los datos previamente.

Además para una mayor facilidad en el manejo de los datos tenemos un archivo de python por cada inserción de datos en cada base de datos.
En la creación de esas bases de datos se declaran las claves primarias de las tablas, esto genera que si se intenta insertar un dato a la tabla cuyas claves ya están introducidas, se obtendrá un error.
Esto es una medida de seguridad que nos permite que no se inserten datos maliciosos o que se borren los datos. Además nos sirve como copia de seguridad en el caso en que la fuente original borre o modifique los datos o el formato de los datos.