Universidad Nacional Autónoma de México.
IIMAS

BASE DE DATOS ESTRUCTURADAS

Semestre 2025-1.

Profesor: Arreola Franco Fernando

Proyecto Final

Integrantes:

- Alanís González Sebastián
- Fabián Sandoval Erick José
- Sáenz de Buruaga Imanol Mendoza
- Villalón Pineda Luis Enrique .

Índice

1. Introducción

En un mundo cada vez más digitalizado y dependiente de los datos, la capacidad de analizar y transformar información en conocimiento se ha convertido en una habilidad indispensable. Este proyecto tiene como propósito aplicar las herramientas y técnicas aprendidas en el curso de bases de datos estructuradas para procesar, organizar y analizar datos meteorológicos provenientes de un servicio web. La importancia de este ejercicio radica no solo en la comprensión de los conceptos técnicos, sino también en la posibilidad de aplicar estos conocimientos a situaciones reales, generando soluciones eficientes y prácticas.

El análisis de datos inicia con la exploración de un conjunto de información proporcionado por el Servicio Meteorológico Nacional, disponible en formato JSON a través de una API. A partir de este punto, se plantea una serie de preguntas relevantes que guían el diseño del modelo de datos y las consultas analíticas. El primer paso es normalizar y estructurar la información, generando modelos entidad-relación y relacional que permitan organizar los datos de manera lógica y funcional. Estos modelos aseguran que los datos sean fáciles de manejar y consultar, estableciendo una base sólida para las siguientes etapas del proyecto.

Una parte esencial del proyecto es el desarrollo de un proceso ETL (Extracción, Transformación y Carga) utilizando la herramienta Pentaho. Este proceso se encarga de convertir los datos originales en un formato estructurado, realizar transformaciones específicas como la sustitución de valores numéricos por descripciones textuales, y asignar tipos de datos que se ajusten al modelo diseñado. Además, se eliminan atributos irrelevantes y se optimiza la representación de la información, garantizando que los datos cargados en la base de datos cumplan con los estándares establecidos.

Una vez procesados los datos, se procede a su análisis mediante consultas SQL avanzadas. Estas consultas permiten responder preguntas clave, como la identificación de municipios con descensos de temperatura significativos o el municipio con la cuarta temperatura más alta en cada estado. Este análisis no solo demuestra la capacidad técnica para manipular bases de datos, sino también la habilidad para interpretar la información y extraer conclusiones útiles.

Finalmente, el proyecto culmina con la creación de un dashboard interactivo que permite visualizar las temperaturas máximas de varios municipios por estado, ordenados de mayor a menor. Este dashboard, además de ser una herramienta útil para la toma de decisiones, refleja la integración de los diferentes elementos del proyecto y la importancia de la visualización en el análisis de datos.

En conclusión, este proyecto no solo pone en práctica los conceptos vistos en clase, sino que también resalta la relevancia de las bases de datos estructuradas en la resolución de problemas del mundo real. A través de un enfoque sistemático, se demuestra cómo el análisis y procesamiento de datos puede contribuir a una mejor comprensión de fenómenos complejos, ofreciendo herramientas prácticas y efectivas para la gestión de información.

2. Plan de Trabajo

Para lograr una organización eficiente y aprovechar al máximo las habilidades de cada miembro, decidimos dividir el proyecto en secciones específicas, asignando a cada integrante un área en función de sus fortalezas y preferencias. Sin embargo, esta división no implicó que cada uno se limitara únicamente a su sección asignada. Hasta la etapa de visualización de datos, cada uno de nosotros realizó todos los pasos del proyecto de manera individual. Esto

nos permitió tener una comprensión integral del trabajo y garantizar que todos estuvieran alineados con los objetivos generales.

La intención detrás de dividir el documento fue permitir que cada miembro profundizara más en su tema asignado, convirtiéndose en un referente para los demás en esa área. De esta forma, si algún integrante enfrentaba dificultades o tenía dudas sobre una sección específica, podía consultar con el compañero especializado en esa parte del trabajo. Esta estrategia fomentó la colaboración y el intercambio de conocimientos dentro del equipo.

La distribución del trabajo se realizó de la siguiente manera:

- Diseño: Enrique
- Procesamiento: Erick
- Analítica de datos: Sebastián
- Visualización: Imanol

Creemos que esta asignación permitió que cada integrante explotara al máximo sus habilidades, profundizando en su área y aportando un valor fundamental al proyecto.

En cuanto a la fase de visualización, acordamos trabajar juntos para obtener un resultado más coherente y de alta calidad. Realizamos sesiones colaborativas donde todos aportamos ideas y sugerencias. Para facilitar el proceso, utilizamos la computadora de nuestro compañero Imanol, quien tiene mayor experiencia en Power BI. Su conocimiento nos permitió aprovechar al máximo las funcionalidades de la herramienta, y al centralizar el trabajo en un solo equipo aseguramos una integración fluida de las contribuciones de todos. Esta forma de trabajar no solo enriqueció el proyecto, sino que también fortaleció nuestras habilidades de trabajo en equipo.

3. Análisis

Durante el análisis de la base de datos, descubrimos que ninguno de los miembros del equipo tenía experiencia previa con el formato de archivos JSON. Este hallazgo resultó particularmente interesante, ya que nos impulsó a investigar y comprender la estructura de datos asociada a este tipo de archivos. Al profundizar en el tema, aprendimos que los archivos JSON son ampliamente utilizados para almacenar datos no estructurados. Esto nos pareció curioso y relevante, especialmente porque es una materia que abordaremos en el próximo semestre. Encontrar este formato en una base de datos que esperábamos fuera estructurada fue inesperado, pero a la vez un desafío enriquecedor.

Con el objetivo de facilitar el manejo de los datos y asegurar que todos los miembros del equipo pudieran trabajar de manera eficiente, decidimos investigar cómo convertir los archivos JSON a un formato CSV. Esta conversión nos permitió manipular los datos en un modelo más familiar y accesible para todos, optimizando así nuestra colaboración y comprensión del conjunto de datos.

Al examinar la base de datos por primera vez, nuestra intención inicial fue normalizarla, principalmente porque identificamos que no cumplía ni siquiera con la primera forma normal. Esta falta de normalización nos generó cierta confusión y nos llevó a reflexionar sobre el grado de atención que las instituciones gubernamentales prestan a la estructura de las bases de datos

que publican. Pareciera que, en este caso, el énfasis se pone en cumplir con la entrega de los datos más que en adherirse a las normas establecidas para el diseño de bases de datos. Esta observación nos hizo cuestionar si se prioriza la calidad y usabilidad de la información o simplemente el cumplimiento de una obligación.

Esta situación nos condujo a identificar diversas limitaciones inherentes a la base de datos. La ausencia de normalización provoca redundancias e inconsistencias que pueden complicar su uso y análisis. Por ejemplo, al no contar con una tabla separada que asocie identificadores únicos con los nombres de los estados, nos enfrentamos a duplicidades al realizar consultas específicas. Si deseamos obtener el nombre de un estado en particular, este aparece repetido tantas veces como registros asociados existan en la tabla principal. Esto no solo dificulta el procesamiento, sino que también puede conducir a errores y a una interpretación errónea de los datos.

Además, la carencia de normalización en la primera forma normal impide que la base de datos cumpla con formas normales más avanzadas, como la segunda y tercera. Esto limita su eficiencia y afecta la integridad referencial de los datos. La normalización es fundamental para eliminar redundancias, evitar anomalías en las operaciones de actualización y garantizar la consistencia de la información almacenada.

Otro problema identificado se relaciona con el campo denominado 'dloc', el cual no se presenta en un formato amigable para el usuario. Observamos que este campo combina información de fecha y hora en un solo dato, lo que complica su manipulación y análisis. Consideramos que, al momento de graficar o analizar estos datos temporalmente, esta falta de formato puede afectar la precisión y claridad de los resultados, especialmente si deseamos visualizar información a nivel de días o identificar patrones temporales específicos. Por ello, sería una buena práctica descomponer este campo, separando la fecha y la hora en columnas distintas. Esta modificación facilitaría el manejo de los datos temporales y mejoraría la calidad de los análisis y visualizaciones que podamos realizar.

Durante este proceso, también reflexionamos sobre las mejores prácticas en la gestión y publicación de datos por parte de las instituciones. La experiencia nos mostró la importancia de contar con bases de datos bien estructuradas y normalizadas para facilitar su uso por terceros. Esto es especialmente relevante en el contexto actual, donde la transparencia y accesibilidad de la información son fundamentales para la toma de decisiones informadas y el desarrollo de soluciones basadas en datos.

4. Diseño

Modelo Entidad Relación (MER):

En la parte de diseño lo primero que se realizó fue saber que significaban cada columna , de esta manera nos hicimos una idea de como hacer el MER y, por lo tanto, ver si el MER cumplía las formas normales. Al ver que significaban cada columna identificamos las entidades, en este caso nos quedaron 3 'Clima', 'Estado' y 'Municipio'; al igual que los atributos que le corresponden a cada entidad. A continuación el Modelo Entidad Relación (MER)

Creemos que la creación de llaves primarias compuestas nos ayudara mejor a la hora de la normalización, pues si en el caso de Municipio dejáramos a 'idmun' como llave primaria sola va a existir duplicados, pues por cada estado la cuenta de municipios empieza en 1 hasta la n cantidad que tenga, lo cual no cumple con la normalización. Pero esta parte está mejor explicado más abajo en la justificación de la normalización.

Figura 1: Modelo Entidad Relacion.

La justificación de la cardinalidad de las relaciones es que un 'estado' puede tener muchos 'municipios' pero esos 'municipios' solo pueden pertenecer a un 'estado'; la misma lógica aplica para la relación 'municipio-clima' pues un clima solo puede estar relacionado con un municipio, pero un municipio puede tener muchos climas.

Una vez obtenido el MER hacemos nuestro Modelo Relacional, pero tengas mucho cuidado, pues al definir nuestros tipos de datos, el tipo de dato debe de coincidir cupón el que vamos a utilizar en nuestras tablas al final de nuestro Flujo ETL. Por lo que en este Modelo Relacional nuestro tipo de dato lo vamos a tomar como viene en la base de datos y ya en el modelo físico usaremos el tipo de datos que terminan en el flujo para que a la hora de cargar nuestros datos a PostgreSQL lo pueda procesar.

Modelo Relacional:

```
ESTADO { ides int PK,
    nes string }
MUNICIPIO{ idmun int,
    nmun string,
    lat int,
    lon int,
    ides int FK
    PRIMARY KEY(idmun, ides) }
CLIMA{ dloc string,
    ndia int,
    tmax int,
    tmin int,
    desciel string,
    probprec int,
```

```
prec int,
velvien int,
dirvienc int,
dirvieng int,
cc int,
raf int,
idmun int,
ides int,
PRIMARY KEY(idmun, ides, dloc),
FORGEIN KEY (idmun, ides) }
```

Veamos del modelo físico y la salida de cada uno de los datos ya cargados a través de PostgreSQL, adelantándonos un poco carguemos los datos que obtuvimos de nuestro flujo ETL,y como mencionamos anteriormente, algunos tipos de datos se modificaron para que quede de acuerdo a nuestras bases generadas.

Modelo Fisico:

```sql
CREATE TABLE ESTADO (
    ides INT PRIMARY KEY,
    nes VARCHAR(100)
);

COPY ESTADO (ides, nes)
FROM 'D:/Ciencia de Datos/5to/Base_Datos_Es/Estado.csv'
DELIMITER ','
CSV HEADER;
select * from ESTADO
```

	ides [PK] integer	nes character varying (100)
1	1	Aguascalientes
2	2	Baja California
3	3	Baja California Sur
4	4	Campeche
5	5	Coahuila
6	6	Colima
7	7	Chiapas
8	8	Chihuahua
9	9	Ciudad de México
10	10	Durango
11	11	Guanajuato
12	12	Guerrero
13	13	Hidalgo
14	14	Jalisco
15	15	Estado de México
16	16	Michoacán
17	17	Morelos
18	18	Nayarit
19	19	Nuevo León
20	20	Oaxaca
21	21	Puebla

```sql
CREATE TABLE MUNICIPIO (
    idmun INT,
    nmun VARCHAR(100),
    lat NUMERIC,
    lon NUMERIC,
    dh INT,
    ides INT NOT NULL,
    PRIMARY KEY (idmun, ides),
    FOREIGN KEY (ides) REFERENCES ESTADO(ides)
);

COPY MUNICIPIO(idmun, nmun, lat, lon, dh, ides)
FROM 'D:/Ciencia de Datos/5to/Base_Datos_Es/Municipios.csv'
DELIMITER ','
CSV HEADER;
select * from MUNICIPIO
```

	idmun [PK] integer	nmun character varying (100)	lat numeric	lon numeric	dh integer	ides [PK] integer
1	54	Magdalena Zahuatlán	17.3884	-97.229	6	20
2	61	Monjas	16.3672	-96.6368	6	20
3	62	Natividad	17.2969	-96.4333	6	20
4	63	Nazareno Etla	17.178	-96.824	6	20
5	77	Reyes Etla	17.2056	-96.8101	6	20
6	87	San Agustín Yatareni	17.0812	-96.6681	6	20
7	91	San Andrés Huayápam	17.1025	-96.6662	6	20
8	99	San Andrés Tepetlapa	17.6658	-98.3917	6	20
9	103	San Antonino Castillo V...	16.8032	-96.6846	6	20
10	107	San Antonio de la Cal	17.0313	-96.6993	6	20
11	127	San Cristóbal Amoltepe...	17.2845	-97.5699	6	20
12	128	San Cristóbal Lachirioa...	17.336	-96.165	6	20
13	145	San Francisco Lachigol...	17.0114	-96.5985	6	20
14	146	San Francisco Loguech...	16.3523	-96.37799999999999	6	20
15	157	San Jacinto Amilpas	17.1008	-96.7626	6	20
16	174	Ánimas Trujano	16.9889	-96.7132	6	20
17	192	San Juan Chilateca	16.8285	-96.67200000000001	6	20
18	216	San Juan Tabaá	17.305	-96.20700000000001	6	20
19	228	San Lorenzo Cuaunecuil...	18.2065	-96.9131	6	20
20	238	San Martín de los Canse...	16.6565	-96.729	6	20

En este caso agregamos el paso de ALTER TABLE CLIMA DROP COLUMN RAF, ya que en el ETL se nos pide eliminar esta columna y si no lo hacíamos a la hora de cargar los datos no iba a realizar la acción.

```sql
CREATE TABLE CLIMA (
    dloc DATE,
    cc NUMERIC,
    desciel VARCHAR(30),
    dirvienc VARCHAR(20),
    dirvieng NUMERIC,
    ndia VARCHAR(20),
    prec NUMERIC,
    probprec INT,
    tmax NUMERIC,
    tmin NUMERIC,
    velvien NUMERIC,
    raf INT,
    idmun INT NOT NULL,
    ides INT NOT NULL,
    PRIMARY KEY (idmun, ides, dloc),
    FOREIGN KEY (ides) REFERENCES ESTADO(ides),
    FOREIGN KEY (idmun, ides) REFERENCES MUNICIPIO(idmun, ides)
);


ALTER TABLE CLIMA DROP COLUMN raf;
copy CLIMA(cc, desciel, dirvienc,dirvieng,dloc, ndia, prec, probprec, tmax, tmin, velvien, idmun, ides)
FROM 'D:/Ciencia de Datos/5to/Base_Datos_Es/clima.csv'
DELIMITER ','
CSV HEADER;
```

```sql
select * from CLIMA
```

Data Output | Messages | Notifications

	dloc [PK] date	cc numeric	desciel character varying (30)	dirvienc character varying (20)	dirvieng numeric	ndia character varying (20)	prec numeric	probprec integer	tmax numeric	tmin numeric	velvien numeric
1	2024-11-09	78.96	Medio nublado	Norte	0	Lunes	0	0	22.8	10.1	7
2	2024-11-10	86.18	Poco nuboso	Norte	0	Martes	0	0	22.4	10	8
3	2024-11-11	78.79	Despejado	Norte	0	Miercoles	0	0	22	9.7	9
4	2024-11-12	77.47	Medio nublado	Noreste	45	Jueves	0	0	21.9	9	8
5	2024-11-09	76.97	Cielo nublado	Noreste	45	Lunes	0.1	0	25.6	13.5	4
6	2024-11-10	87.48	Medio nublado	Noreste	45	Martes	0.1	0	25.9	12.7	5
7	2024-11-11	95.19	Poco nuboso	Noreste	45	Miercoles	0.2	0	25.5	12.7	6
8	2024-11-12	74.65	Poco nuboso	Noreste	45	Jueves	0.1	0	25.3	12.9	5
9	2024-11-09	72.91	Cielo nublado	Noreste	45	Lunes	0.2	0	22.9	9.6	4
10	2024-11-10	89.11	Poco nuboso	Noreste	45	Martes	0	0	22.9	9.4	4
11	2024-11-11	92.97	Poco nuboso	Noreste	45	Miercoles	0	0	22.3	9.7	5
12	2024-11-12	82.24	Medio nublado	Noreste	45	Jueves	0	0	22.2	9	4
13	2024-11-09	81.98	Cielo nublado	Norte	0	Lunes	0	0	24.5	10.5	5
14	2024-11-10	88.17	Poco nuboso	Norte	0	Martes	0	0	24.6	10.2	5
15	2024-11-11	92.6	Poco nuboso	Norte	0	Miercoles	0	0	24.4	10	
16	2024-11-12	61.27	Poco nuboso	Norte	0	Jueves	0	0	23.7	9.6	5

1. **1FN:** Cumple con la primera forma normal, ya que todas las tablas tienen valores atómicos y solo existe una llave primaria en la tabla, aquí mencionamos lo que ya habíamos dicho que en municipio y clima generamos una llave compuesta para que cumpliera esta forma, pues al general la lave compuesta ahora si los valores son únicos para cada tabla, porque por ejemplo si en municipio dejábamos solo al 'idmun' como llave primaria no iba a cumplir, pues existen muchos 1 , ya que por estado se vuelve a reiniciar el contador de esta manera ya la llave es uncia en cada tabla, pues el mismo concepto se usa para clima y generar la llave compuesta con esos tres atributos hace que tengamos una predicción de acuerdo al día, estado y municipio como mi llave primaria lo cual hace que no allá valores repetidos en mi llave primaria

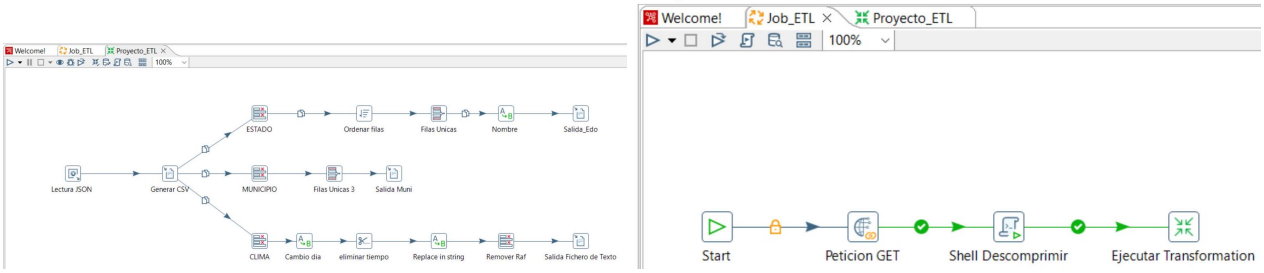
2. **2FN:** Cumple la 1FN y cumple el primer criterio...

3. **3FN:** Como cumple la 1FN y la 2FN, cumple el primer rubro...

5. Procesamiento

Nuestro flujo ETL está compuesto por dos archivos. El primero de ellos es la transformación que se encarga de realizar los puntos solicitados en el proyecto en relación con nuestra propuesta de modelo, mientras que el segundo es un job encargado de ejecutar todo el procesamiento en orden incluyendo la transformación mencionada.

Transformación:



Ahora, analizaremos cada step de ambos archivos paso por paso:
Comenzaremos con el Job, pues como tal es el que se encarga de ejecutar todo nuestro proceso.

1. Descargar los datos del sitio web por medio de una petición GET.

Petición GET: Esto lo conseguimos con el step HTTP que se encarga de realizar una petición de un servicio web, el cual por defecto es una operación de tipo GET, por lo cual automáticamente nos descarga el archivo consumido por el servicio en la dirección especificada. ${Internal.Entry.Current.Directory} es una variable que apunta a la dirección de memoria donde se tiene almacenado el job/transformación actual.

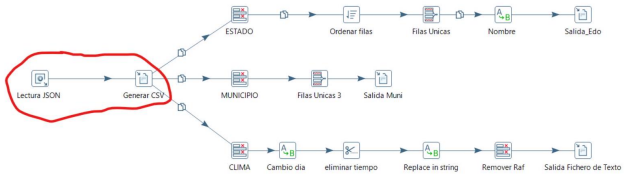
Shell Descomprimir:Como podemos ver, se descarga un archivo.gz del enlace proporcionado, el cual es una carpeta comprimida con un único documento dentro que es del tipo JSON, por lo que, para extraerlo utilizamos este step Shell en el cual insertamos un script que se ejecuta similar a la consola de nuestra máquina. En este comando descomprimimos el archivo.gz por medio de una herramiento que tuvimos que descargar llamada 7-zip. Ya después solo especificamos las rutas donde queremos guardar el archivo descomprimido y con

el parámetro -y forzamos la sobrescritura de archivos, esto por si volvemos a ejecutar el job no haya problema y se actualicen los valores actuales.

Ejecutar Transformación: El último paso de este job es mandar a llamar a ejecución la transformación que hace los siguientes pasos. Esto es sencillo, solo tenemos que especificar la ruta de nuestra transformación y en este caso tuvimos que desmarcar la columna wait for remote transformation to complete, ya que hacía que nunca terminará de ejecutarse la transformación.

2. Convertir de formato JSON a un formato estructurado

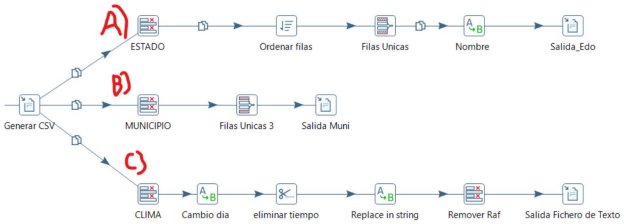
A partir de este punto comienza el análisis de nuestra transformación:



Lectura JSON: En primer lugar leímos nuestro archivo JSON con JSON input donde solamente especificamos la ruta donde esta guardado y especificamos los campos que queremos leer.

Generar CSV: Después continuamos el flujo con un text file output, para pasarlo a un formato estructurado, en este caso CSV, solo especificamos la ruta, la extensión del archivo y en cuanto a los campos solo le dimos GET fields.

Como podemos ver, a partir de aquí nuestra transformación se divide en tres salidas donde en cada una de ellas copiamos los resultados del csv generado en el paso anterior. A continuación explicaremos cada una de estas tres salidas:



A) Obtención de los 32 estados

ESTADO: Este es un step Select Values donde simplemente seleccionamos únicamente los campos de número de estado (ides) y número de estado (nes) de el csv del paso anterior.

Ordenar filas: Es un step Sort Rows, donde el resultado lo almacenamos en la variable temporal %%java.io.tmpdir%%. El orden sobre el cual deseamos ordenar
es en base al campo ides y podemos especificar varios otros valores, en este caso, que sea de forma ascendente.

Filas Unicas: Este paso llamado Unique rows filtra filas únicas, para eliminar todas las filas repetidas de un conjunto de datos. Se le necesita especificar un campo sobre el cual quiere eliminar las filas repetidas, que en este caso es ides.

Nombre: Este es un Replace in String, lo implementamos para tratar algunas entradas de nombres de estados que nos podrían dar conflictos o simplemente para homogeneizar los nombres. En este caso, hay tres nombres de estados del campo nes que aparecían con su nombre completo: Querétaro de Arteaga, Michoacán de Ocampo, y Veracruz de Ignacio

de Llave. Reemplazamos estos valores por su nombre más corto y conocido: Querétaro, Michoacán y Veracruz.

Salida_Edo: Finalmente guardamos la salida en un csv con Text File Output, especificamos la direccion interna de memoria con la variable ${Internal.Entry.Current.Directory} y al obtener los campos con el botón Get Fields pues nos apareceran los unicos dos campos con los que trabajamos en este flujo: ides y nes.

B) Obtención de municipios con sus latitudes y longitudes, entre otros

MUNICIPIO: Para comenzar con Select Values, seleccionamos solo los campos que nos interesan, en este caso idmun, nmun, lat, lon, dh y ides.

Filas Unicas 3: Es muy parecido al filas únicas del subflujo anterior, igual es un paso Unique rows, pero en esta ocasión ordenamos acorde a dos campos, los cuales son idmun ides, para que, en otras palabras, encontramos todos los municipios sin importar si aparece otro con el mismo nombre en otro estado, y al mismo tiempo eliminamos todos los renglones repetidos.

Salida Muni: Seguimos con la misma lógica, ahora guardamos esta salida en un archivo con extensión csv con ayuda de Text File Output en la dirección ${Internal.Entry.Current.Directory}/Municipios y únicamente resta seleccionar los campos a seleccionar con el botón Get Fields, que en este caso son idnum, nmun, lat, lon, dh, ides.

C) Obtención del clima

Cómo ultimo subflujo tenemos este donde obtenemos la salida de un archivo csv con los campos que nos interesan. NOTA IMPORTANTE: En este flujo es donde implementamos las transformaciones requeridas en el proyecto de: Sustituir el valor numérico del atributo ndia por el día de la semana correspondiente y Remover el atributo Raf. Lo de asignar tipos de datos adecuados de acuerdo a la propuesta de
nuestros modelos lo implementamos desde el principio e incluso en los otros dos subflujos.

CLIMA: Comenzamos con este step de Select Values donde seleccionamos los campos que nos interesan: cc, desciel, dirvienc, dirvieng, dloc, ndia, prec, probprec, raf, tmax, tmin, velvien, idmun, ides. En la parte de metadata cambiamos el tipo de datos del atributo ndia por string ya que nos interesa manipularlo más adelante.

Cambio día: En este step Replace in String hacemos un mapeo de los valores de la columna ndia, en este caso, reemplazamos el numero por el dia en una escala de 0=Lunes, 1=Martes, 2=Miercoles, 3=Jueves, 4=Viernes, 5=Sabado, 6=Domingo. Por eso que en el paso anterior en la metadata cambiamos el tipo de dato del ndia por string para poder manipularlo aquí.

Eliminar tiempo: Este es un step strings cut, donde recortamos los valores del campo dloc, a solo 8 carectares, ya que están en un formato de este tipo 20241121T00 donde no nos interesa la letra T ni ningun digito despues de este, por lo que recortamos la cadena.

Replace in String: Ya que tenemos el año mes y dia en ese orden en la columna de dloc, queremos cambiar a formato AAAA/MM/DD. Con este paso Replace in String es posible pero tenemos que ir asignando los digitos conforme estan en esta columna, para el año asignamos los primeros cuatro digitos, para el mes los siguientes dos y para el día los siguientes dos.

Remover Raf: Este es un Select Values donde seleccionamos todas las columnas que llevamos hasta ahora pero aprovechamos para en la parte de Remove escribir el nombre del

atributo raf y eliminarlo por ende de nuestro resultad. De igual forma en metadata cambiamos el tipo de dato de dloc de String a Date pues ya lo tenemos en un formato adecuado.

Salida Fichero de Texto: Finalmente solo queda guardar este ultimo resultado en nuestra memoria local, por lo que lo guardamos en esta dirección ${Internal.Entry.Current.Directory}/Clima con la extensión csv y seleccionamos todos los campos que queremos almacenar con el botón Get Fields.

Y con esto, llega al final nuestra transformación dando como resultado 6 archivos generados al final de la ejecución que quedarán guardados en la carpeta donde tengamos nuestra transformación y el job. Dichos resultados son los siguientes:

1. DailyForecast_MX.gz: Carpeta comprimida descargada del servicio web.
2. DailyForecast_MX: Archivo en formato JSON extraído de la carpeta.
3. Pronostico_TresDias.csv: Resultado de pasar el archivo JSON a csv.
4. Estado.csv: Conjunto con los 32 números de estado (ides) y sus nombres (nes).
5. Municipios.csv: Conjunto con todos los municipios del archivo, seleccionando todos estos campos: idmun, nmun, lat, lon, dh, ides
6. Clima.csv: Conjunto después de aplicar las transformaciones requeridas (cambios a ndia, fecha y remover raf). Los campos mostrados son los siguientes: cc, desciel, dirvienc, dirvieng, dloc, ndia, prec, probprec, tmax, tmin, velvien, idmun, ides.

6. Analítica de datos

Pasaremos a responder las preguntas que se nos piden:

A Mencione y explique detalladamente al menos dos ventajas y dos desventajas de la forma en que inicialmente se estaba representando la información.
La primera ventaja que podemos notar de la base de datos es que puedes ver todos los datos juntos y poder ver la base de datos completa y para qué nos podría servir la información que se nos proporciona. La segunda ventaja podría ser que al venir en un archivo JSON, estos archivos son utilizados para bases no estructuradas, por lo que podríamos meter tipos de datos estructurados. Más allá de eso, no encontramos otras ventajas por qué la forma en la que vienen no nos parece la más adecuada, por lo que justo mencionando una de las desventajas es que la base de datos no está normalizada y eso genera que existan datos repetidos y si queremos generar una consulta sobre los datos así, probablemente lo haga mal. Por último, también tenemos la desventaja es como estaban definidos los datos, por ejemplo en 'dloc' no había una estructura, solo estaba marcado como un tipo de dato 'string' cuando debería de tratarse como un tipo de dato 'date' y así paso con varios datos como con 'nes' cuando se quiera gráficas probablemente como nos pasó a nosotros tiene el nombre completo y tuvimos que cambiar el nombre por el que reconociera nuestro Visualizador (ejemplo 'Querétaro de Arteaga' por 'Querétaro'), al igual que tener los datos para alguien que solo vea la base de datos tal como se nos presenta si no tiene un contexto de los datos algunas

columnas no son claras de interpretar.

B Generar una consulta que muestre los 5 municipios donde se vayan a presentar descensos de temperatura(tmin)más marcados entre hoy y el siguiente día.
Vamos a explicar por qué esa consulta, trabajando con datos almacenados en la tabla CLIMA. A diferencia de otras variantes, aquí se utiliza una unión explícita con JOIN para comparar los registros de un día (c1) con los del día siguiente (c2), en lugar de hacer una comparación interna directa entre filas. La cláusula SELECT define las columnas que se desean en el resultado final. Se seleccionan el identificador del municipio (idmun), el identificador de la estación (ides), y las fechas correspondientes al día actual (c1.dloc) y al día siguiente (c2.dloc). También se incluyen las temperaturas mínimas del día actual (c1.tmin) y del día siguiente (c2.tmin), además de una columna calculada que representa la magnitud del descenso de temperatura entre ambos días, definida como (c1.tmin - c2.tmin). Esta diferencia se etiqueta como descenso. La consulta utiliza un JOIN explícito para combinar dos instancias de la tabla CLIMA, alias c1 y c2. La unión se realiza bajo las siguientes condiciones: los identificadores del municipio (idmun) y de la estación (ides) deben coincidir entre ambas tablas, y la fecha en c2 debe ser exactamente un día después de la fecha en c1. Esto se logra con la condición c2.dloc = c1.dloc + INTERVAL '1 day'. Este enfoque permite comparar los datos correspondientes a días consecutivos dentro de la misma estación y municipio. El filtro en la cláusula WHERE asegura que únicamente se incluyan en los resultados los casos donde la temperatura mínima del día actual (c1.tmin) sea mayor que la del día siguiente (c2.tmin). Esto garantiza que solo se consideren los descensos de temperatura, descartando aumentos o estabilidades. La cláusula ORDER BY descenso DESC organiza los resultados en orden descendente basado en la magnitud del descenso. Así, las mayores disminuciones de temperatura aparecen primero en el listado. Finalmente, la cláusula LIMIT 5 restringe la cantidad de filas devueltas a las cinco mayores disminuciones registradas.

```
SELECT c1.idmun, c1.ides, c1.dloc AS fecha_hoy, c2.dloc AS fecha_mañana, c1.tmin AS tmin_hoy,
       c2.tmin AS tmin_mañana, (c1.tmin - c2.tmin) AS descenso
FROM CLIMA c1 JOIN CLIMA c2 ON c1.idmun = c2.idmun  AND c1.ides = c2.ides  AND c2.dloc = c1.dloc + INTERVAL '1 day'
WHERE c1.tmin > c2.tmin ORDER BY descenso DESC LIMIT 5;
```

Data Output | Messages | Notifications

	idmun integer	ides integer	fecha_hoy date	fecha_mañana date	tmin_hoy numeric	tmin_mañana numeric	descenso numeric
1	30	28	2024-11-11	2024-11-12	23.8	19.8	4.0
2	13	28	2024-11-11	2024-11-12	23.6	19.7	3.9
3	8	28	2024-11-11	2024-11-12	23.6	20	3.6
4	1	28	2024-11-11	2024-11-12	22.9	19.8	3.1
5	18	28	2024-11-11	2024-11-12	22.6	19.7	2.9

C Generar una consulta que permita obtener, por estado, el municipio con la cuarta temperatura más alta.

La consulta la pensamos de la siguiente manera de la tabla CLIMA. Para lograrlo, utiliza una expresión común de tabla denominada ranking, que calcula un rango denso (DENSE_RANK) basado en las temperaturas máximas ordenadas de forma descendente dentro de cada estación. El proceso comienza con la definición de la CTE ranking mediante el bloque WITH. Dentro de esta CTE, se seleccionan tres columnas principales de la tabla CLIMA: el identificador del estado (ides), el identificador del municipio (idmun) y la temperatura máxima (tmax). Además, se genera una columna adicional, rank, utilizando la función de ventana DENSE_RANK(). Esta función asigna un rango a cada registro dentro de un grupo definido por la cláusula PARTITION BY ides, que

asegura que el cálculo del rango se haga de forma independiente para cada estado. Los registros en cada grupo se ordenan por tmax en orden descendente mediante la cláusula ORDER BY tmax DESC. Esto significa que el registro con la mayor temperatura máxima en un estado recibe el rango 1 , el siguiente mayor recibe $2,$ y así sucesivamente, asignando el mismo rango a registros con valores idénticos de tmax. La consulta principal utiliza la CTE ranking para filtrar los resultados. En el bloque SELECT, se extraen las columnas ides, idmun y tmax de la CTE. Luego, la cláusula WHERE rank $= 4$ asegura que solo se incluyan los registros con un rango de 4, es decir, aquellos que representan la cuarta temperatura máxima más alta dentro de cada estado. Este filtro elimina todos los demás rangos, dejando únicamente los registros relevantes. El uso de DENSE_RANK() garantiza que si varias temperaturas comparten la misma posición en el ranking (por ejemplo, si hay varios registros empatados en el tercer lugar), el rango siguiente será el cuarto lugar y no se omitirá ningún valor.

```sql
WITH ranking AS (
    SELECT
        ides,
        idmun,
        tmax,
        DENSE_RANK() OVER (PARTITION BY ides ORDER BY tmax DESC) AS rank
    FROM
        CLIMA
)
SELECT ides, idmun, tmax FROM ranking WHERE rank = 4;
```

Data Output | Messages | Notifications

	ides integer	idmun integer	tmax numeric
1	1	1	29.2
2	2	2	26.6
3	3	3	30.8
4	4	6	34
5	5	15	33.1
6	5	1	33.1
7	6	9	32.5
8	6	6	32.5
9	7	114	32.8
10	8	45	31.5
11	9	6	25.4

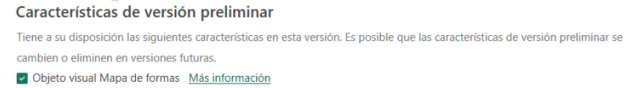
D Si con los datos con los que está trabajando tuviera que resolver un problema o implementar alguna mejora de una situación real, ¿cuál podría ser un caso de estudio?

Podríamos analizar la escasez de agua en relación con el clima, y así desarrollar estrategias para un uso más eficiente de este recurso vital. Por ejemplo, si la base de datos nos proporciona predicciones meteorológicas a tres días para la región, podríamos crear una base de datos que compile los datos climáticos diarios y clasifique las estaciones del año, como verano, invierno, etc. Esto nos permitiría generar series temporales para todos los municipios, facilitando la identificación de patrones y tendencias.Con esta información, podríamos proponer estrategias de almacenamiento de agua anticipándonos a las condiciones climáticas futuras. Si las predicciones a tres días indican un aumento de las temperaturas, se podría almacenar más agua para enfrentar una posible mayor demanda o menor disponibilidad. Si las series temporales muestran que nos encontramos en una época del año en la que es más difícil obtener agua, sería aún más necesario que el estado, como entidad reguladora, limite el suministro a los sectores que más la necesitan, como los municipios agrícolas o las zonas urbanas. Además, podríamos aplicar técnicas de clustering para identificar cuáles municipios

consumen más agua. Utilizando modelos de predicción, podríamos encontrar la manera óptima de distribuir el agua a todos los municipios, ajustando el suministro según las condiciones climáticas y las necesidades específicas de cada región. De esta forma, podríamos asegurar un acceso equitativo al agua, incluso en períodos de escasez, y promover un uso sostenible del recurso hídrico en función de las variaciones climáticas.

7. Visualización

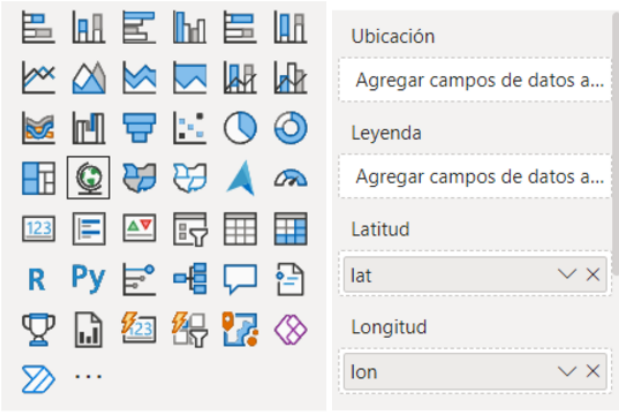
Para la parte de la visualización, debido a que previamente ya teníamos un conocimiento acerca de la plataforma Power bi y de igual manera nuestro conjunto de datos no tiene un tamaño excesivo (recordando que esta plataforma no es recomendable para el manejo de grandes cantidades de datos) usaremos esta plataforma para el desarrollo de las gráficas. Como parte inicial se propone un mapa de coloración que represente la temperatura por cada estado , esta opción se encuentra dentro de las opciones en desarrollo de power BI que se puede activar desde la configuración de la plataforma en el apartado de Características de versión preliminar.



Una de las principales limitaciones o problemas que nos encontramos al tomar esta representación es que por las características del país a representar, la parte de municipios todavía no es posible y justamente por eso la opción de representación se encuentra en fase preliminar, posteriormente se explica como se soluciono este problema ya que representar los municipios es parte esencial para el problema. Como paso previo, y como se aprendió en anteriores clases, la manipulación de los datos se hizo directamente desde pentaho (flujo etl) para que las tablas estén bien referenciadas entre sí al igual que valores consistentes que nuestro mapa pueda aceptar.

La métrica que tomará nuestro mapa en la parte de coloración será la temperatura máxima, ya que es el propósito de nuestra visualización. Puesto que nuestro mapa principal

no reconoce los municipios, insertamos una gráfica auxiliar a modo de mapa geográfico la cual tiene como ventaja principal que puede tomar atributos como longitud y latitud en lugar de ubicación por lo que estas dos columnas dentro de nuestra tabla municipio fueron de extrema utilidad.

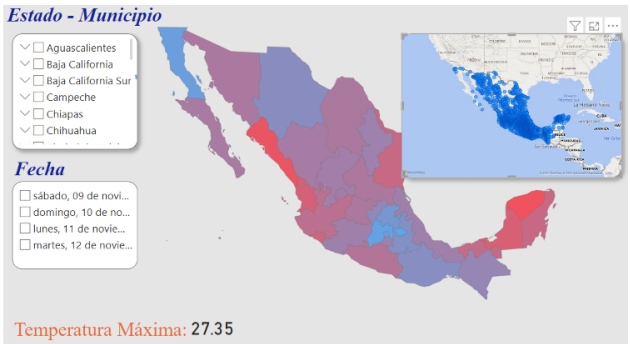


Para ambos mapas ajustamos diversos parámetros de estética y el más importante es ajustar para que al aplicar los filtros posteriores, estos indiquen y enfaticen haciendo zoom en la sección en específica dentro de ambos mapas. Tenemos los dos filtros relevantes:

1. Estado - Municipio

2. Dia para la consulta (recordando que todas son a las 12:00)

Por último, insertamos una ficha , dado que se nos solicita un dato duro a responder, el cual es la temperatura máxima de un estado/municipio en específico y el día reportado. De igual, un problema que se presenta es que al seleccionar alguno de los filtros, el enfoque en el mapa tarda un poco, pero esto es debido a las limitaciones técnicas de la computadora, una más potente podría desempeñarse con total fluidez La presentación queda de la siguiente manera:



Para terminar, debido a que nuestros datos sólo representan una periodo de tiempo en específico, a la hora de actualizar la visualización es necesario que los archivos .csv tengan el mismo nombre , la misma ubicación y por supuesto, el mismo número de atributos y estructura.

8. Concluiones

El desarrollo de este proyecto representó una experiencia integral para el equipo, donde se puso a prueba no solo el conocimiento técnico, sino también las habilidades de comunicación, organización y resolución de problemas. A lo largo de las distintas fases, enfrentamos desafíos que nos permitieron aprender cómo se lleva a cabo el desarrollo de un proyecto desde cero, logrando un resultado satisfactorio que cumple con los objetivos propuestos y que, además, trasciende los límites del aula para aplicarse en situaciones reales en cuanto a manejo de datos. Uno de los principales logros fue el diseño de un modelo de datos eficiente y bien estructurado, que permitió organizar y normalizar la información. Este proceso no solo facilitó la comprensión de los datos, sino que también garantizó la eficiencia en las consultas propuestas. Otro aspecto destacable fue la implementación del proceso ETL en Pentaho, una tarea compleja que implicó convertir los datos del formato JSON a un esquema estructurado, eliminar algunos atributos irrelevantes (como Raf), asignar tipos de datos adecuados y cargar la información en las tablas diseñadas. Esta parte fue crucial demostrando la importancia de las herramientas modernas en la gestión de datos. Además, la construcción del dashboard interactivo mediante Power BI fue un logro clave del proyecto. Este dashboard no solo cumple con los requisitos técnicos establecidos, sino que también destaca por su diseño intuitivo y centrado en la usabilidad. Desde el inicio, se planteó como un objetivo principal desarrollar una herramienta que permitiera a los usuarios explorar datos complejos de forma sencilla y visualmente atractiva.

El dashboard permite consultar fácilmente las temperaturas máximas por municipio y estado, ofreciendo a los usuarios una experiencia dinámica y accesible. A través de filtros interactivos y visualizaciones gráficas, los datos se presentan de manera clara, facilitando tanto la interpretación como la toma de decisiones informadas.

Una de las características más destacadas del dashboard es su capacidad para reflejar cambios si se necesitan actualizar a una fecha más cercana, integrándose directamente con la base de datos. Este aspecto fue en especial un reto técnico debido a que nunca se habia implementado algo parecido y se tuvo que investigar las características necesarias para lograr esto.

En resumen, el uso de Power BI no solo permitió cumplir con los objetivos técnicos del proyecto, sino que también fue esencial para crear una herramienta bastante intuitiva y gráfica para los usuarios. Este componente fue una prueba de la capacidad del equipo para transformar datos complejos en información valiosa y fácilmente interpretable.

Sin embargo, el camino no estuvo exento de desafíos. La integración de datos desde una API externa presentó problemas iniciales relacionados con la comprensión de la documentación y la limpieza de los datos obtenidos. Estos obstáculos fueron superados mediante un análisis detallado y una adecuada planificación. Otro desafío significativo fue el diseño del modelo relacional, donde se tuvo que encontrar un balance entre la normalización de datos y la optimización de las consultas analíticas. Adicionalmente, la coordinación entre los miembros del equipo fue un reto constante, especialmente en la asignación de responsabilidades y en la integración final del trabajo individual. El análisis y la creación de consultas avanzadas también supusieron un aprendizaje crucial. Responder preguntas específicas, como identificar los municipios con los descensos más pronunciados de temperatura, no solo requirió habilidades técnicas en SQL, sino también la capacidad de interpretar correctamente los datos y extraer conclusiones útiles. Este proceso nos permitió

comprender mejor la relación entre los datos meteorológicos y su aplicación en casos de estudio prácticos. Cada miembro del equipo enfrentó retos propios, desde la comprensión de herramientas nuevas como Pentaho hasta la resolución de errores inesperados en las distintas fases del proyecto. Estos momentos, aunque desafiantes, nos brindaron la oportunidad de mejorar nuestras habilidades técnicas y desarrollar una mayor resiliencia ante problemas complejos. Además, el trabajo en equipo fortaleció nuestras capacidades de comunicación y colaboración, permitiéndonos alcanzar objetivos comunes y aprender unos de otros. En conclusión, este proyecto no solo representó un ejercicio académico, sino una experiencia formativa que nos permitió aplicar conocimientos técnicos en un contexto práctico y realista. El resultado obtenido refleja no solo el esfuerzo conjunto, sino también el aprendizaje adquirido durante el proceso. Estamos convencidos de que las habilidades y conocimientos desarrollados serán fundamentales en futuros retos profesionales, permitiéndonos abordar problemas complejos de manera estructurada y eficiente.