PROYECTO AEMET

Desarrollo de Aplicaciones
para Ciencia de Datos

2º Ciencia e Ingeniería de Datos

Escuela de Ingeniería Informática - Universidad de Las Palmas de Gran Canaria (ULPGC)

Resumen

El proyecto consiste en 3 programas. En el primero, que es un módulo del proyecto AEMET2 llamado Feeder, queremos un programa que descargue cada hora los datos de todas las estaciones meteorológicas desde el webservice de la AEMET. Con estos datos el programa debe generar un datalake con datos meteorológicos de Gran Canaria. Esto lo conseguimos haciendo un segundo get a la url que conseguimos con el get del snippet de código que se nos proporcionó. Luego, filtro por los que la latitud y longitud correspondan a la isla y esos son los que entrarán a los ficheros. Además de los ficheros.events, que son los que nos interesan, tengo otro archivo events.json, que se usa para evitar datos duplicados, comprobando que no existan en el events.json que es el que tiene todos, así cuando el programa se ejecute cada hora solo meterá los datos nuevos en los ficheros YYYYMMDD.events. Los eventos que se escriben en esos ficheros solo tienen los datos que se nos pidieron, que eran el fint (fecha y hora), ubi (lugar), idema (id del lugar), tamax y tamin (temperatura máxima y mínima). IMPORTANTE: RECORDAR PONER UNA API KEY PARA PODER EJECUTAR EL PROGRAMA.

En el segundo, un programa que, a partir de los datos del datamart construya un datamart con las temperaturas máximas y mínimas de cada día en la isla. Se trata de crear una base de datos SQLite (Datamart.db) en la que se crearán dos tablas: una de temperaturas máximas y otra de temperaturas mínimas. En estas tablas se añade un registro cada día con la siguiente estructura: fecha, hora, lugar, estación, valor.. Para esto, se crea la base de datos, se borran las tablas si existen para crear nuevas actualizadas e introducirle posteriormente los datos de cada fichero del datamart. En el método ProcessFiles se encarga de coger cada fichero del datalake que acabe en .events y ProcessFile de la clase Insert lo que hace es que de cada fichero pasado por ProcessFiles, coge los elementos de los JsonObject, y los mete en una lista de eventos, para posteriormente filtrarlos por el que mayor y menor temperatura tenga del día, para posteriormente hacer el insert con las nuevas listas de eventos filtradas. Este programa se ejecuta cada 12 horas, así esta base de datos está actualizada 2 veces al día.

Y por último en el tercero, Un programa que, usando el datamart creado por el programa anterior, ofrezca una API REST para realizar consultas de los sitios con mayor temperatura y menor temperatura de la isla en un rango de días:

GET /v1/places/with-max-temperature?from={date}&to={date}

GET /v1/places/with-min-temperature?from={date}&to={date}

IMPORTANTE: para hacer las consultas se ha de hacer de la siguiente manera:

http://localhost:4567/v1/places/with-max-temperature?from=20230111&to=20230113

http://localhost:4567/v1/places/with-min-temperature?from=20230111&to=20230113

Aclaro esto ya que imagino que podría causarle confusión con programas de compañeros que hayan puesto guiones entre el año y el mes y el mes y el día. Yo decidí hacerlo de esta manera ya que como los ficheros del datalake del programa 1 tienen la estructura yyyyMMdd.events hacer las peticiones en la apirest de la misma manera.

Recursos utilizados

-Intellij
-Git
-GitHub
-Google
-Spark
-Google Docs
-Talend API Tester
-Jsoup
-Maven
-Maven central
-Gson
-SQlite

Diseño

Model view controller (Sin view porque no se crea interfaz)

Líneas futuras

La idea de negocio en este caso es clara, se puede hacer predicciones del tiempo en días posteriores.

Conclusiones

Lecciones aprendidas: Gracias a este proyecto ahora me ha quedado aún más claro cómo usar Spark y cómo funcionan las Api Rest. También me manejo mejor con Jsoup y Json. Otra lección aprendida es que gracias a la corrección del trabajo de spotify ahora se que no debo usar rutas absolutas para referenciar a un fichero (en este caso la creación de la base de datos .db), ya que esto no es portable a otros entornos. En su lugar, apliqué una ruta relativa como me aconsejó.

Si empezara de nuevo: Quizás usaría interfaces, sin embargo, dudo que este programa se pueda reutilizar con otras cosas aparte de AEMET.

Bibliografía

https://sparkjava.com/
https://jsoup.org/
https://opendata.aemet.es/centrodedescargas/inicio

Joaquín Ibáñez Penalva
13/01/23