Seminario sobre Pentaho Data Integration

Actividad 11

Enunciado del problema.

Se dispone de una URL de la que poder obtener la localización de las estaciones meteorológicas de la Dirección General de Tráfico (DGT).
(http://infocar.dgt.es/etraffic/BuscarElementos?latNS=44&longNS=10&latSW=35&longSW=-10.0&zoom=14&accion=getElementos&Camaras=false&SensoresTrafico=false&SensoresMeteorologico=true&Paneles=false&Radares=false&IncidenciasRETENCION=false&IncidenciasOBRAS=false&IncidenciasMETEOROLOGICA=false&IncidenciasPUERTOS=false&IncidenciasOTROS=false&IncidenciasEVENTOS=false&caracter=acontecimiento)

De esta URL se obtiene un documento JSON con una estructura fija, con el identificador de estación, su localización en forma de dirección y su posición geográfica.

{"carretera":"A-395","estado":1,"alias":"A-395 Pk 16.7 D","sentido":"DEC","PK":16.7,"tipo":"SensorMeteorologico","lng":-3.4831719398498535,"codEle":"GUID_MET_141","lat":37.1390495300293}

Con esta información se puede construir una tabla de una base de datos postgreSQL y extensión POSTGIS para almacenar los atributos y localización.

Además conocemos otra URL de la DGT que nos permite consultar los datos de una estación individual a partir de su identificador.

(http://infocar.dgt.es/etraffic/BuscarElementos?accion=getDetalles&tipo=SensorMeteorologico&indiceMapa=0&codEle=GUID_MET_70333)

El resultado de consultar dicha estación, de nuevo es un documento JSON con las últimas observaciones registradas por ella.

{"temp_rocio":"11", "fecha":"2015-10-26 17:29:00.0", "textoAdvertenciaPrecision": "La ubicación en el mapa es aproximada", "radiacion_global": "11", "vel_viento": "2","est_super": "--", "tipo_viento": "Normal", "alt_agua": "--", "temperatura": "12","visibilidad": "2000", "indiceMapa": "0", "t_super" :"--", "humedad" :"93", "i_Precipitaciones": "0", "n_Precipitacion": "Lluvia", "salinidad": "--", "t_congel": "--", "tip_viento": "49", "tipo": "SensorMeteorologico", "tiempo_Presente": "Lluvioso", "c_Precipitaciones": "2", "dir_viento": "--", "t_subsuelo": "--", "presion_A": "1018"}

Se desea:

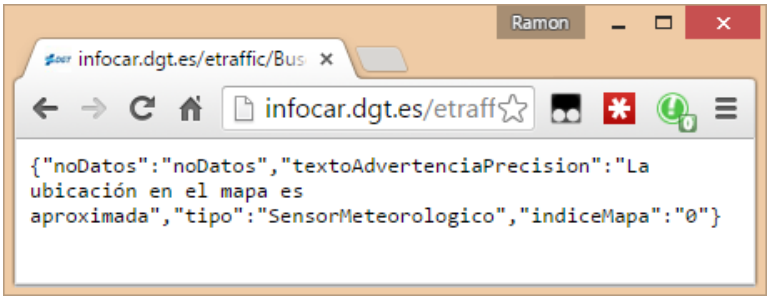
1.- Generar una transformación que acceda a la primera URL para recuperar los datos de las estaciones meteorológicas y los almacene en una tabla de PostGIS.

2.- Se desea generar una tarea que pueda ser invocada periódicamente, como tarea programada de Windows, para recuperar secuencialmente para cada estación las últimas observaciones, las procese desde el JSON y las almacene en otra tabla de postgreSQL. Todo

ello de tal forma que se mantenga la relación entre estación y observaciones por una clave común que es el identificador de estación. Hay que tener presente que no todas las estaciones tienen porqué estar activas en todo momento. Es decir pueden recuperarse consultas con resultados noDatos.

Información no válida:

http://infocar.dgt.es/etraffic/BuscarElementos?accion=getDetalles&tipo=SensorMeteorologico&indiceMapa=0&codEle=GUID_MET_70141



Información válida:

http://infocar.dgt.es/etraffic/BuscarElementos?accion=getDetalles&tipo=SensorMeteorologico&indiceMapa=0&codEle=GUID_MET_70143

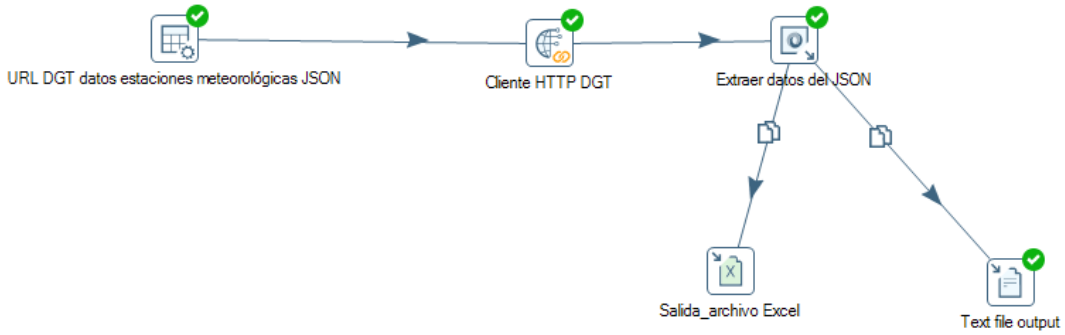
Solución basada en PDI:

Primera transformación:

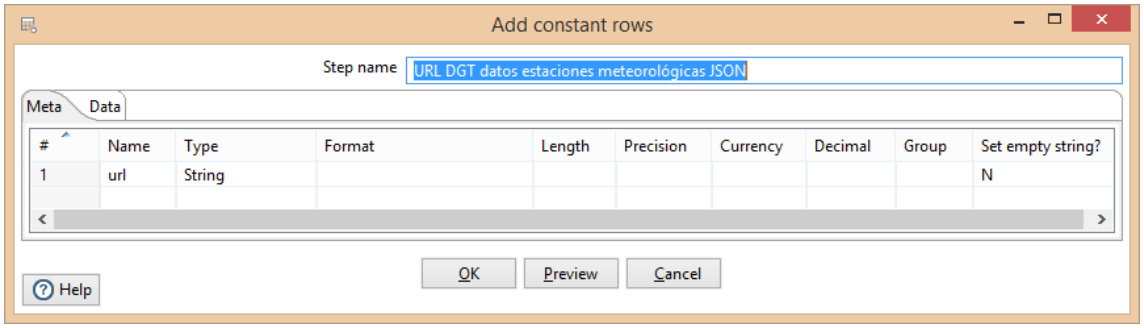
Se generará una primera transformación lee_dgt_meteo que acceda a la siguiente URL:

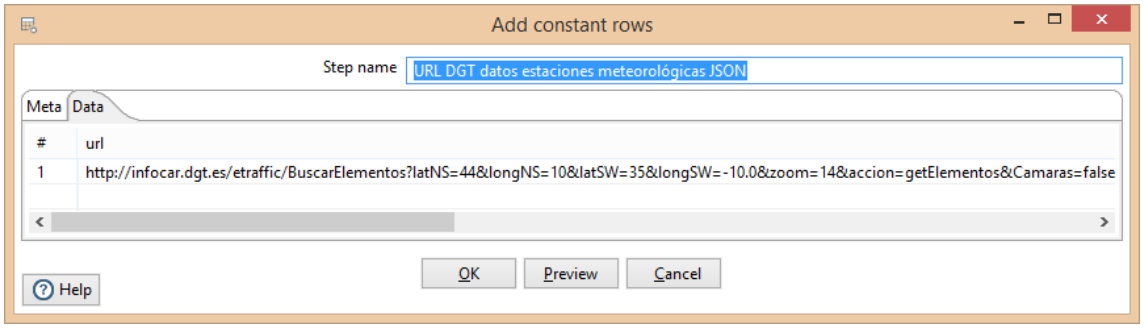
http://infocar.dgt.es/etraffic/BuscarElementos?latNS=44&longNS=10&latSW=35&longSW=-10.0&zoom=14&accion=getElementos&Camaras=false&SensoresTrafico=false&SensoresMeteorologico=true&Paneles=false&Radares=false&IncidenciasRETENCION=false&IncidenciasOBRAS=false&IncidenciasMETEOROLOGICA=false&IncidenciasPUERTOS=false&IncidenciasOTROS=false&IncidenciasEVENTOS=false&caracter=acontecimiento

para obtener las estaciones meteorológicas y que genere dos ficheros de salida. Un fichero en formato Excel con todos los datos y otro en formato csv solo con el nombre de las estaciones, en el campo: CodEle

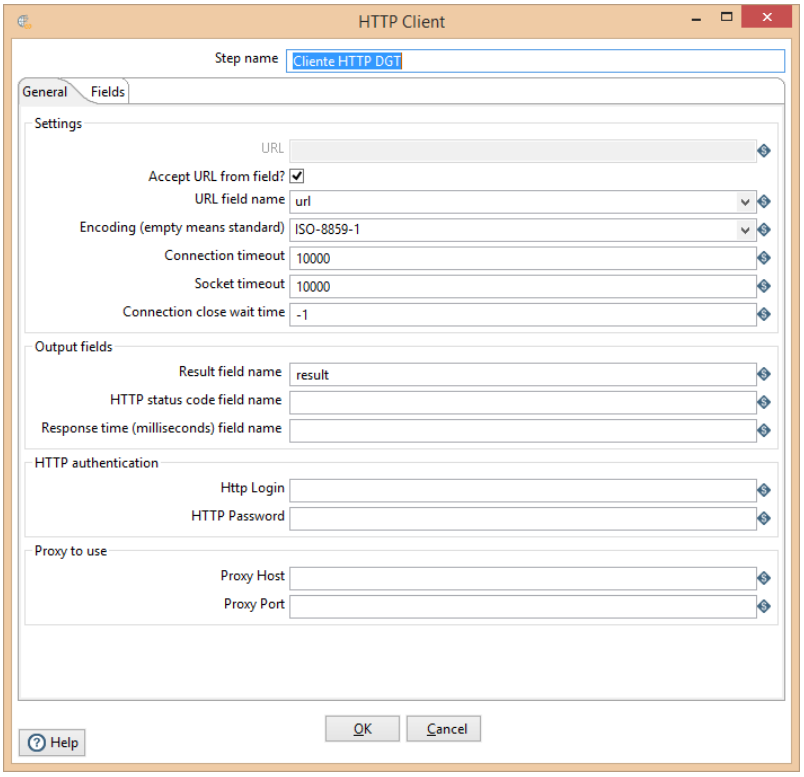


El primer paso es un “data grid”. Añadimos un atributo de tipo *string* y nombre *url*. En datos, asignamos al atributo *url* el valor de la *url* para obtener las estaciones meteorológicas:



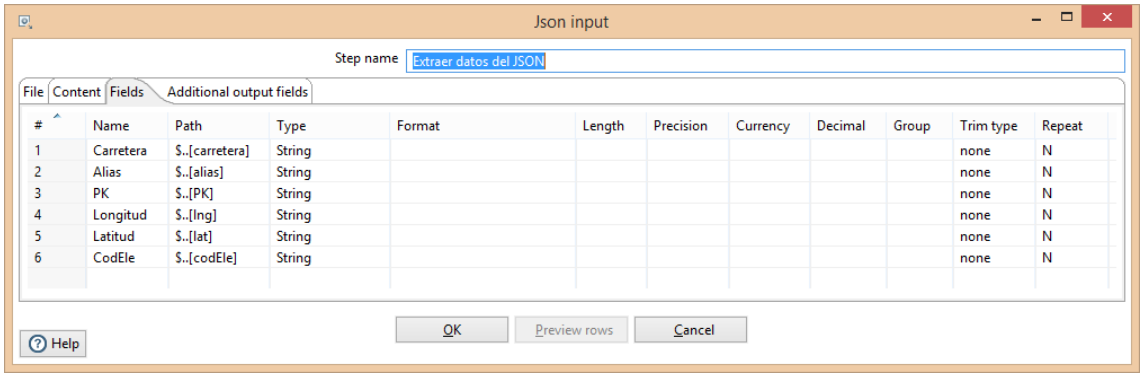


El segundo paso es un “HTTP Client”. Lo configuramos para obtener la URL de un campo y generar el resultado en otro campo que llamaremos “result”

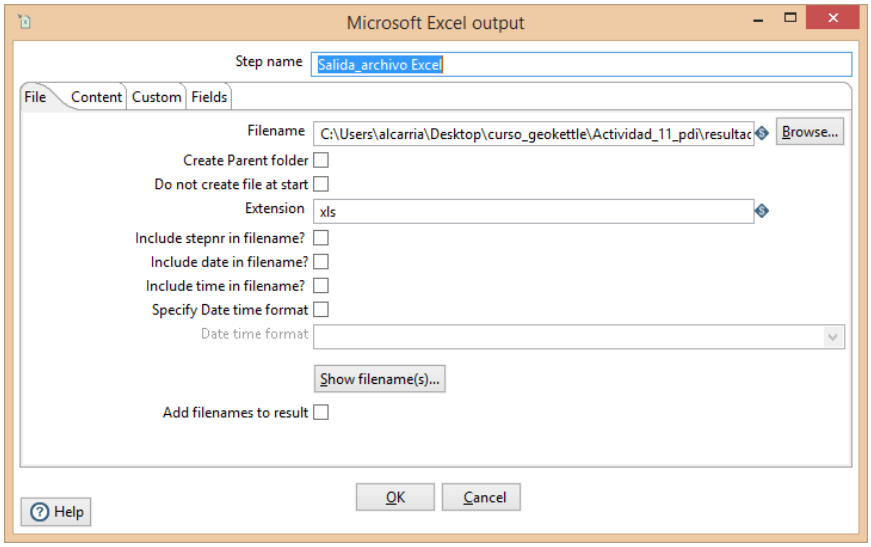


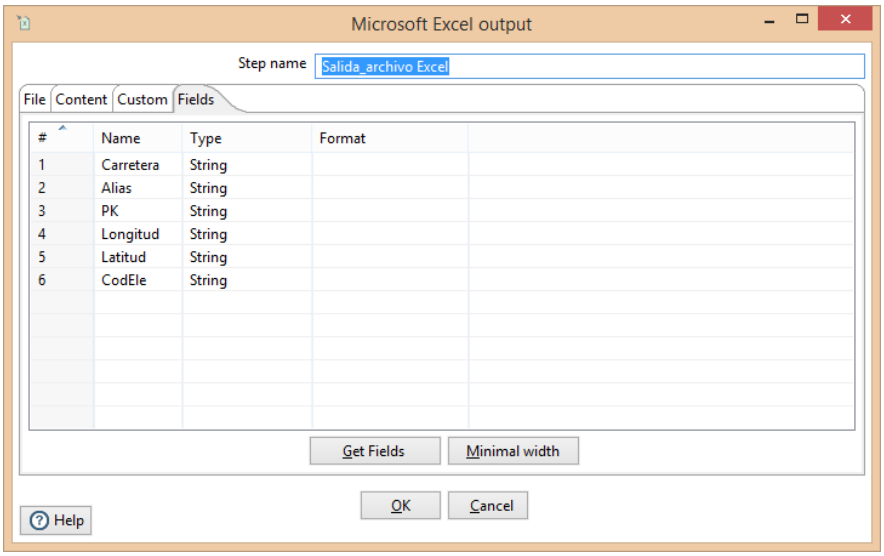
El tercer paso es un JSON input. Seleccionamos que la fuente de datos está definida en un campo y seleccionamos como campo “result”.

Indicamos los campos que queremos salvar:

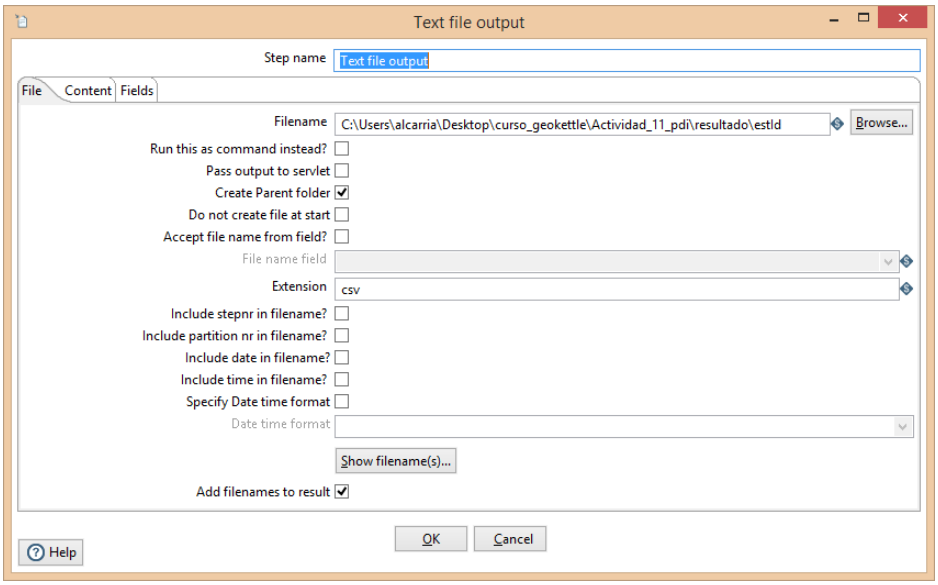


Como últimos pasos seleccionamos una salida a fichero de Excel y definimos los campos que queremos escribir:

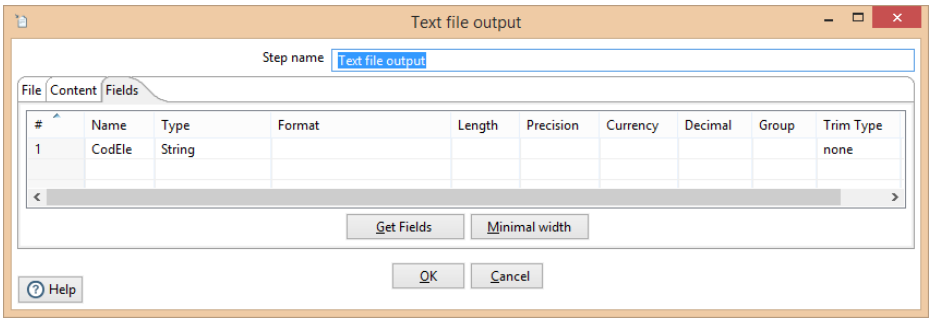




En paralelo seleccionamos una salida en fichero de texto y definimos un fichero de salida en formato csv:



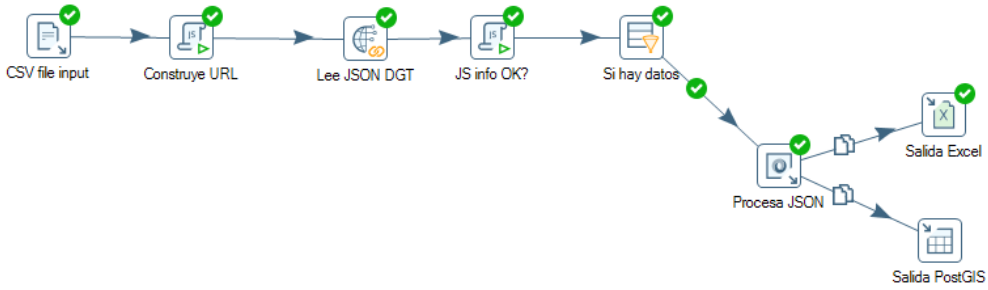
Que contenga únicamente el campo CodEle:



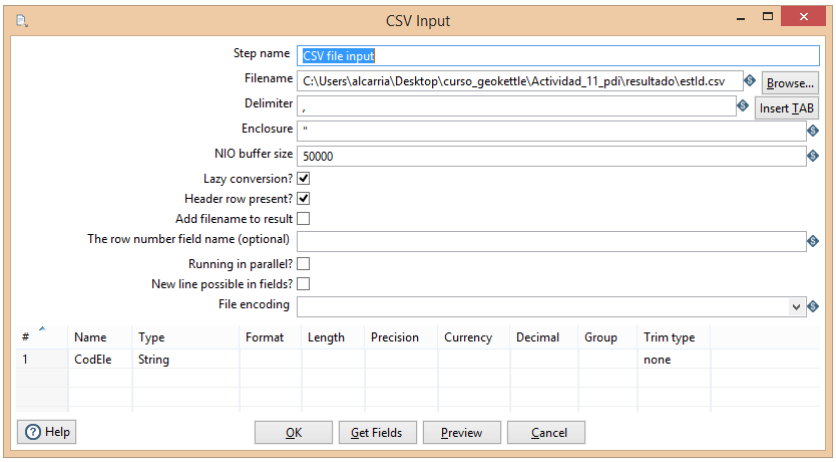
Segunda transformación:

Creamos una segunda transformación que recupere el *csv* con los nombres de las estaciones de meteorología y vaya haciendo peticiones HTTP para recuperar la información de cada estación. Finalmente guardaremos la información en un fichero Excel y en una tabla de PostGIS.

La transformación tiene este aspecto:

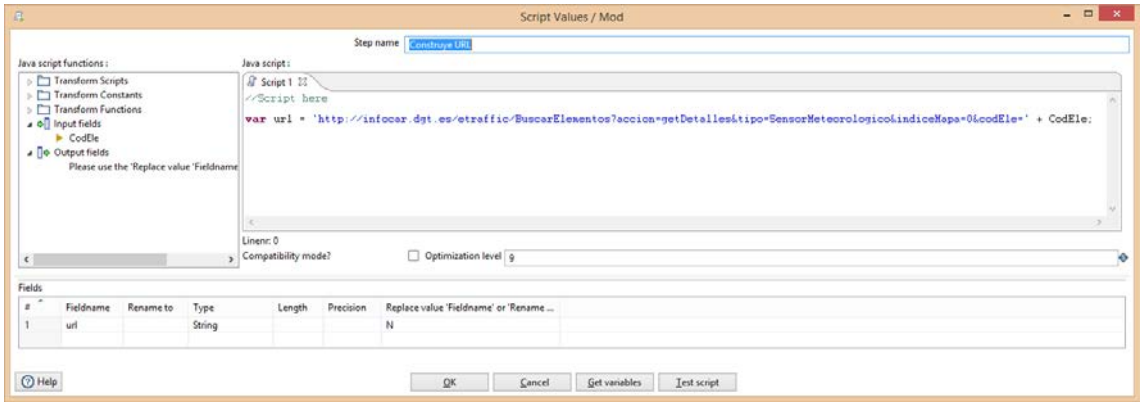


El primer paso es recuperar la información del fichero csv a través de un "CSV file input".

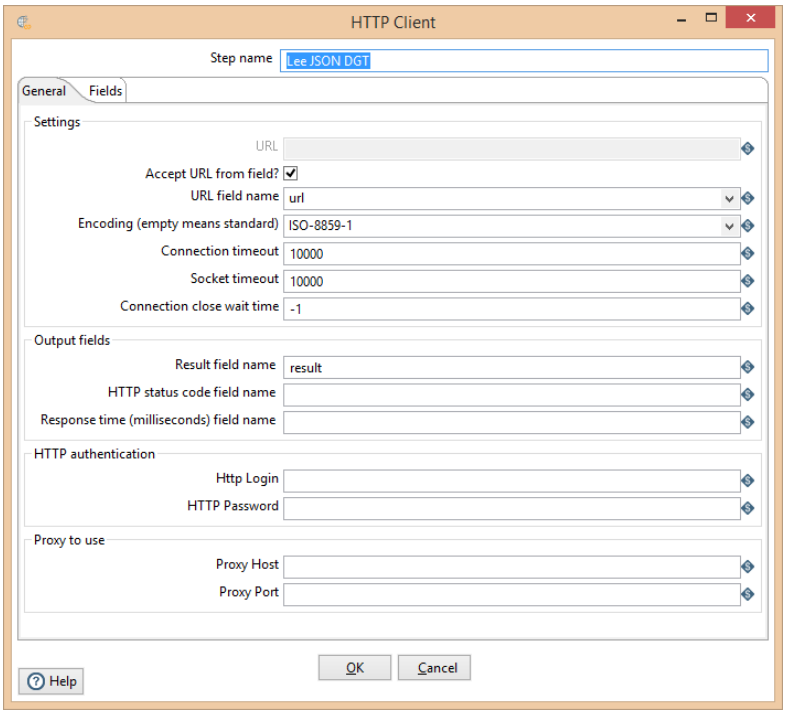


Construimos la URL con un paso de "Modified Java Script Value": Definimos la variable url como:

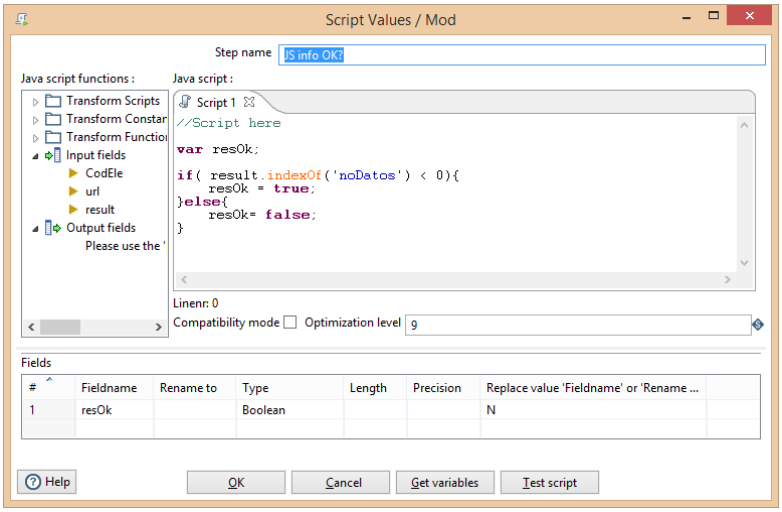
```
var url =
'http://infocar.dgt.es/etraffic/BuscarElementos?accion=getDetalles&tipo=Sensor
Meteorologico&indiceMapa=0&codEle=' + CodEle;
```



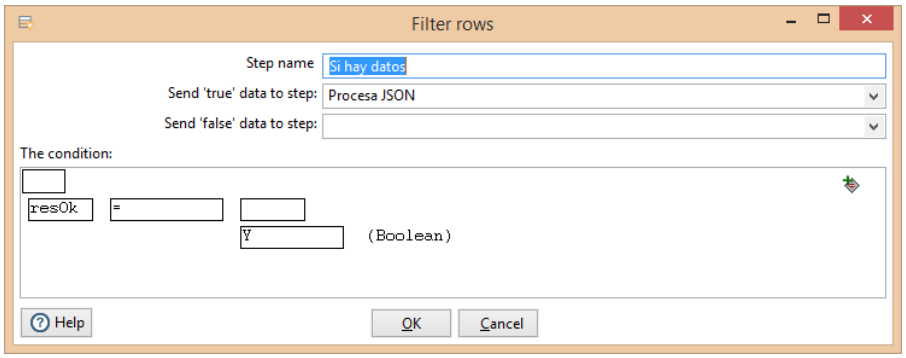
En un paso HTTP Client construimos las peticiones HTTP seleccionando el campo url como el proveedor de las direcciones:



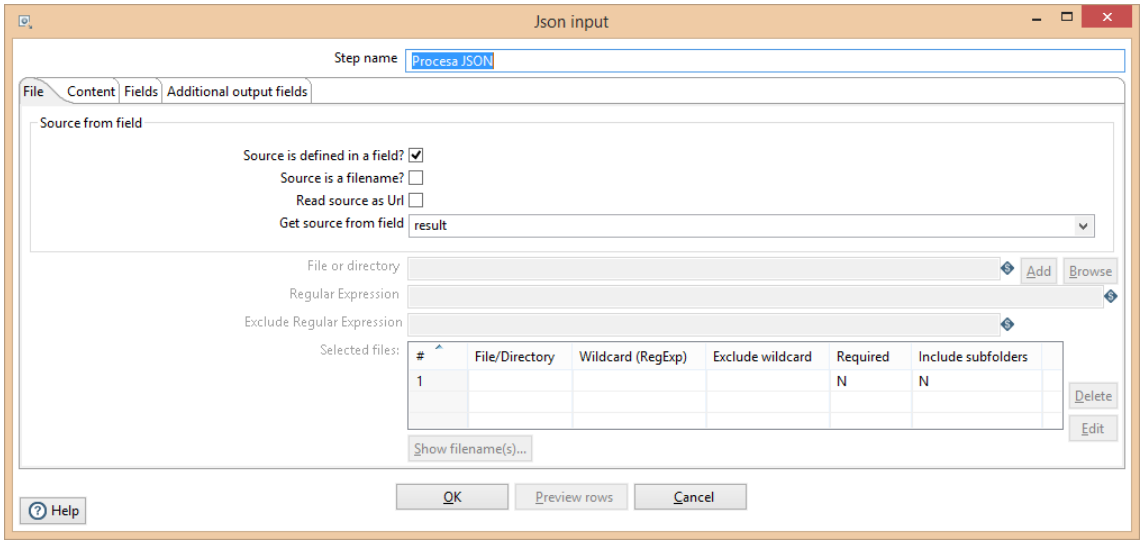
Tenemos que ver si el mensaje de respuesta contiene el texto “noDatos”. En ese caso debemos descartarlo. Para ello definimos un script donde establecemos la variable resOk a verdadero o a falso dependiendo de si el texto “noDatos” está presente en el resultado.

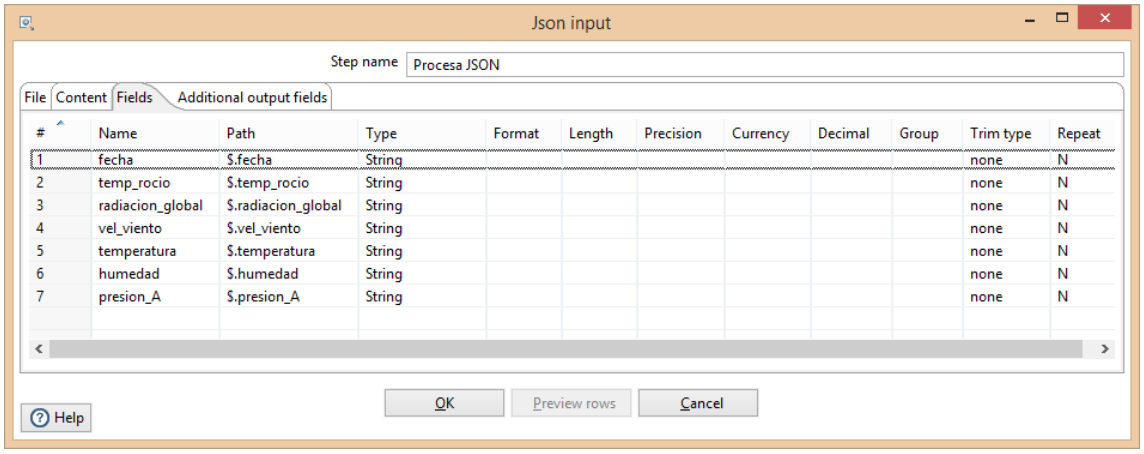


Añadimos un paso de “filter rows” que llame al siguiente paso cuando resOk sea verdadero:



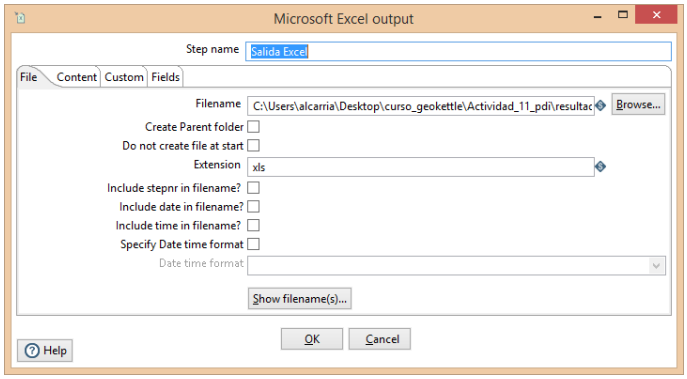
Definimos un paso “Json input” que recupera del campo resultado los datos necesarios:

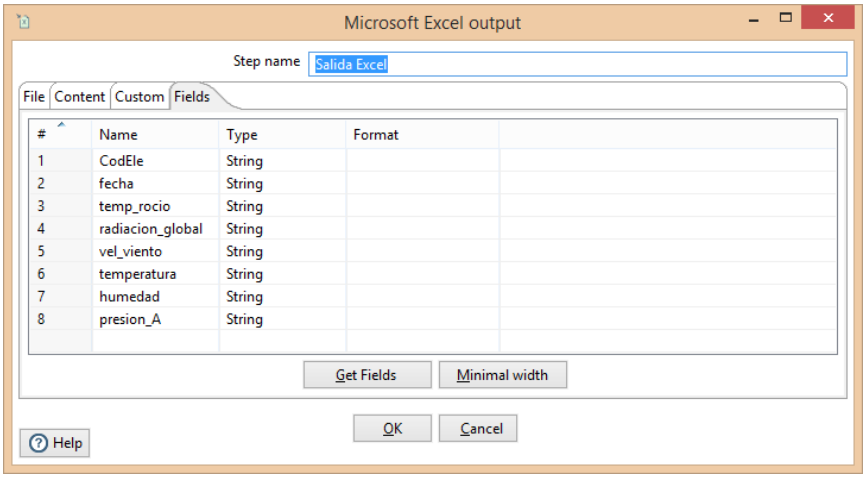




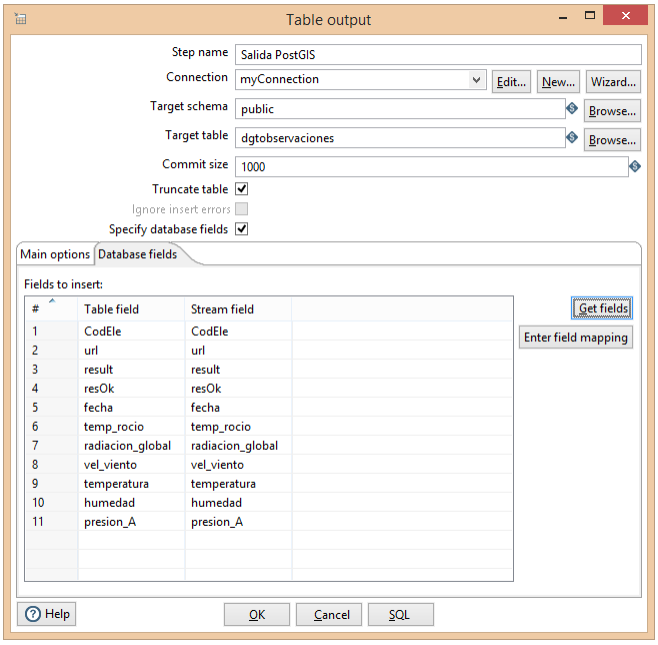
Por último definimos las salidas:

Un paso de salida Excel con todos los campos:





Y un paso de salida a base de datos:

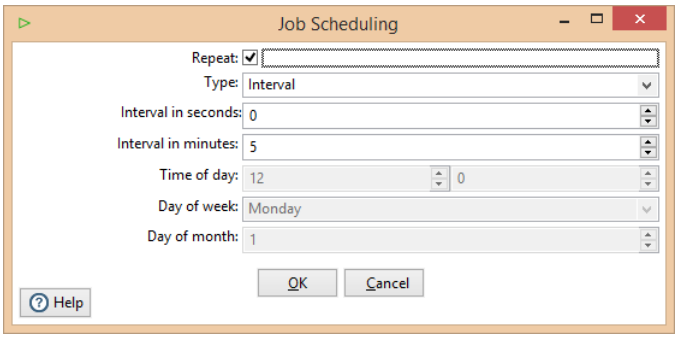


Extensión:

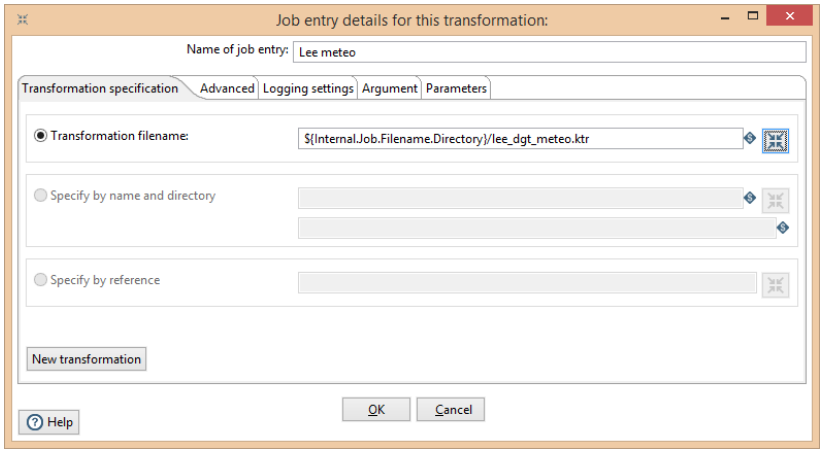
Vamos a extender la actividad anterior para que el proceso de obtención de estaciones y de valores para las estaciones se realice cada cinco minutos. Para ello crearemos un trabajo que llame a estas dos transformaciones de forma periódica.

Como primera acción crearemos un nuevo trabajo en Pentaho. Utilizamos los siguientes pasos:

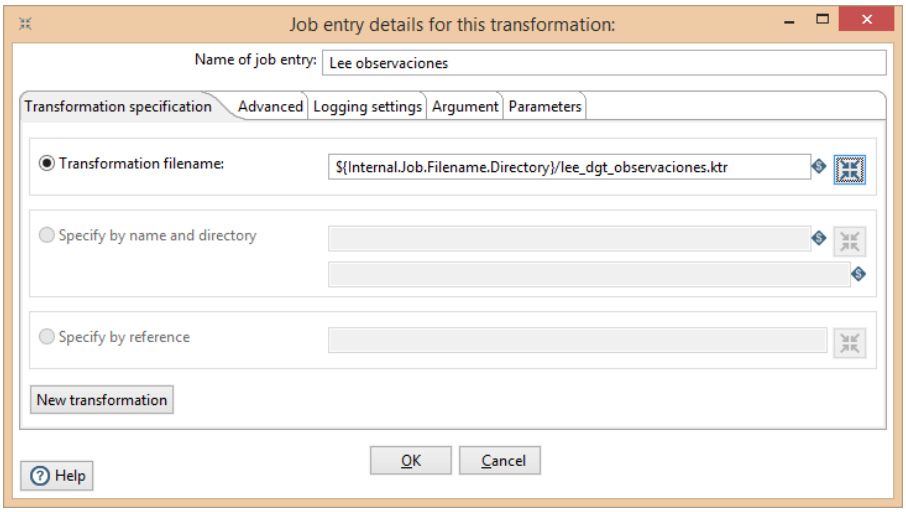
El primer paso es una actividad de Start, que debemos configurar para repetirse cada 5 minutos:



El primer paso de trasformación debe configurase para ejecutar la trasformación de lee_dgt_meteo:



El segundo paso de trasformación debe configurase para ejecutar la trasformación de lee_dgt_observaciones:



Si ejecutamos el trabajo vemos que hay que esperar los primeros 5 minutos para que se empiecen a ejecutar las dos transformaciones. También se puede observar como hasta que no termina la última transformación no empiezan a contar los siguientes 5 minutos para repetir la ejecución del trabajo.

Por último ejecutaremos el trabajo fuera de *Spoon* y aprenderemos a como ejecutar esta tarea automáticamente cuando se arranque el ordenador, a través del programador de tareas de Windows.

Para ejecutar trabajos “Jobs” fuera de Spoon se utiliza el programa Kitchen.bat, integrado en la suite de Pentaho.

Un manual de uso básico de Kitchen se enlaza aquí:

http://wiki.pentaho.com/display/EAI/Kitchen+User+Documentation

Tenemos que crear un archivo con extensión .bat y abrirlo con un archivo de texto. Introduciremos el siguiente texto:

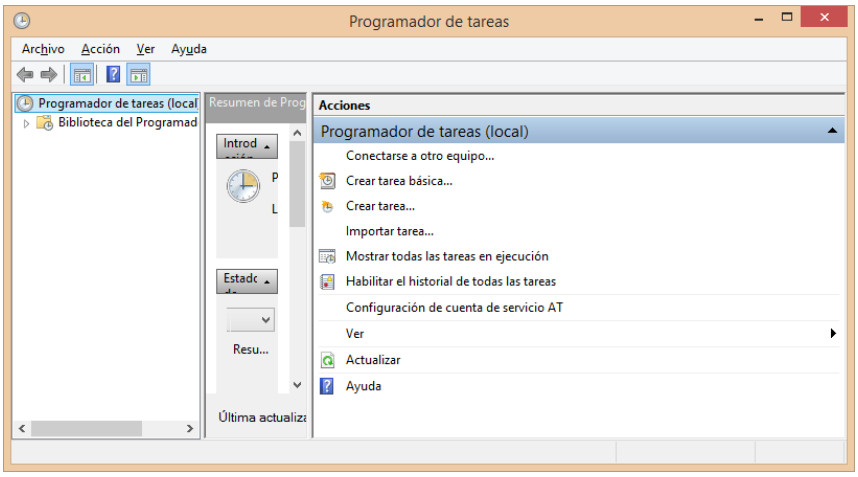
cd <data-integration>

call Kitchen.bat /file:<direccionJob>\act_11_job.kjb /level:Basic

Siendo <data-integration> la carpeta de instalación del pentaho, que es a su vez la dirección donde tenemos el archivo Kitchen.bat

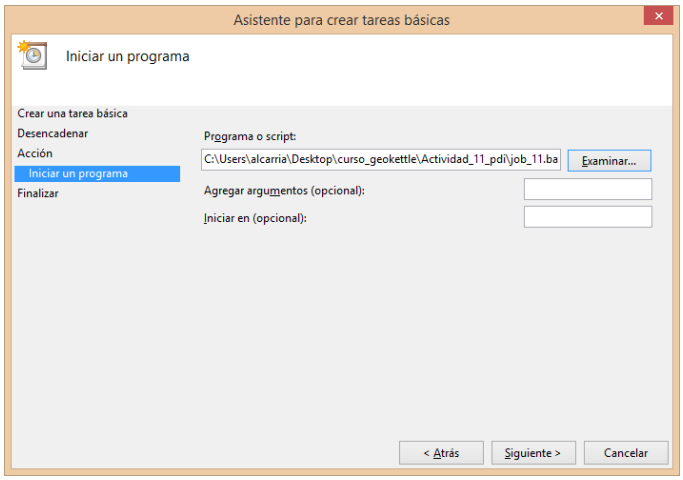
Y <direccionJob> donde tenemos el archivo Job de la actividad 11.

Por último abrimos el programador de tareas:

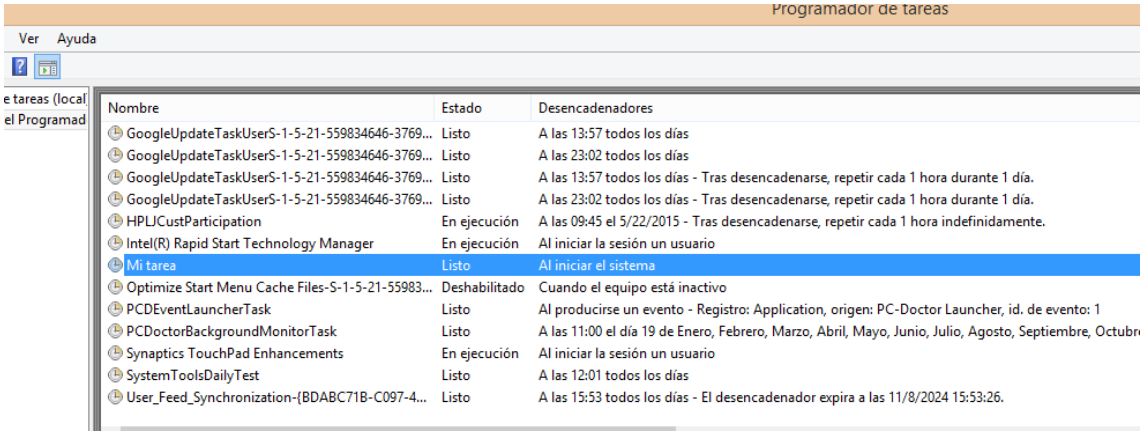


Pulsamos en crear tarea básica y elegimos que se ejecute al iniciar el equipo.

La acción será iniciar un programa y seleccionaremos el fichero bat que hemos creado:



Podremos encontrar la tarea en la lista de tareas, y ejecutarla de prueba


Programador de tareas
Ver
Ayuda
e tareas (local
el Programad
Nombre
Estado
Desencadenadores
GoogleUpdateTaskUserS-1-5-21-559834646-3769...
Listo
A las 13:57 todos los días
GoogleUpdateTaskUserS-1-5-21-559834646-3769...
Listo
A las 23:02 todos los días
GoogleUpdateTaskUserS-1-5-21-559834646-3769...
Listo
A las 13:57 todos los días - Tras desencadenarse, repetir cada 1 hora durante 1 día.
GoogleUpdateTaskUserS-1-5-21-559834646-3769...
Listo
A las 23:02 todos los días - Tras desencadenarse, repetir cada 1 hora durante 1 día.
HPLJCustParticipation
En ejecución
A las 09:45 el 5/22/2015 - Tras desencadenarse, repetir cada 1 hora indefinidamente.
Intel(R) Rapid Start Technology Manager
En ejecución
Al iniciar la sesión un usuario
Mi tarea
Listo
Al iniciar el sistema
Optimize Start Menu Cache Files-S-1-5-21-55983...
Deshabilitado
Cuando el equipo está inactivo
PCDEventLauncherTask
Listo
Al producirse un evento - Registro: Application, origen: PC-Doctor Launcher, id. de evento: 1
PCDoctorBackgroundMonitorTask
Listo
A las 11:00 el día 19 de Enero, Febrero, Marzo, Abril, Mayo, Junio, Julio, Agosto, Septiembre, Octubr
Synaptics TouchPad Enhancements
En ejecución
Al iniciar la sesión un usuario
SystemToolsDailyTest
Listo
A las 12:01 todos los días
User_Feed_Synchronization-{BDABC71B-C097-4...
Listo
A las 15:53 todos los días - El desencadenador expira a las 11/8/2024 15:53:26.