Trabalho Prático Nº1

Integração de Sistemas de Informação

Da autoria de:

Paulo Costa Nº29851

IPCA
INSTITUTO POLITÉCNICO DO CÁVADO E DO AVE
ESCOLA SUPERIOR DE TECNOLOGIA

Índice

Introdução 4

Ferramentas Utilizadas 5

Job "meteorologia_job.kjb" 5

Transformação "dados_meteorologicos.ktr" 5

- **Input de dados em tempo real** 6
- **Input de dados de previsões** 7
- **"Merge Join" e "Filtra Portugal"** 8

Criação do XML estruturado 10

Vídeo com demonstração 12

Conclusão 13

Índice de Figuras

Figura 1 - meteorologia_job.kjb 5
Figura 2 - dados_meteorologicos.ktr 5
Figura 3 - Input de dados em tempo real 6
Figura 4 - Pasta "dados" 6
Figura 5 - Corre todos os ficheiros Json em dados 7
Figura 6 - Dados Atuais 7
Figura 7 - Sort atual 7
Figura 8 - Input de dados de previsões 7
Figura 9 - Dados de previsões 7
Figura 10 - Recebe dados do passo anterior 8
Figura 11 - Dados extraidos do array 8
Figura 12 - Dados de csv_previsoes.csv 8
Figura 13 - Resultado csv_previsoes.csv 8
Figura 14 - “Merge Join” e “Filtra Portugal” 8
Figura 15 - Merge Join 9
Figura 16 – Dados join.json 9
Figura 17 - Conteudo join.json 9
Figura 18 - Filtra Portugal 9
Figura 19 - Criação do XML estruturado 10
Figura 20 - Criação do campo previsao_xml 10
Figura 21 - Agrupa previsões por cidade 10
Figura 22 - Criação do cidade_xml 11
Figura 23 - Header e footer do XML 11
Figura 24 - Escrever header do xml (content) 11
Figura 25 - Escrever header do xml (fields) 11
Figura 26 - Escrever cidade_xml (content) 12
Figura 27 - Escrever cidade_xml (content) 12
Figura 28 - Fecha a tag "cidades" 12
Figura 29 - Video Demo 12

Introdução

O presente relatório descreve a implementação de um processo ETL (Extract, Transform, Load) utilizando o Pentaho Data Integration (Kettle), com o objetivo de integrar dados meteorológicos provenientes de diferentes fontes. Foram utilizados vários ficheiros JSON contendo os dados meteorológicos atuais e previsões de três dias para diversas cidades em todo o mundo.

O processo desenvolvido consistiu na extração destes dados, consolidação em CSV, combinação através de merge join, armazenamento em JSON e filtragem de cidades portuguesas com exportação para XML, de forma a obter um conjunto de dados organizado e facilmente consultável. Este trabalho ilustra não só a capacidade do Kettle em manipular diferentes formatos de dados, como também a importância da integração de sistemas para a construção de informação consistente e útil.

Ferramentas Utilizadas

Na realização deste trabalho prática foram utilizadas as seguintes ferramentas:

- **Pentaho Data Integration (Kettle) General Availability Release versão 10.2** – Como ambiente de desenvolvimento do projeto;
- **Weather API** – Utilizada como fonte dos dados meteorológicos utilizados como input.

Job "meteorologia_job.kjb"

O job *meteorologia_job.kjb* tem como objetivo automatizar a execução da transformação ETL principal do projeto. Permite assim que o processo possa ser repetido de forma simples e confiável, sem necessidade de interação manual a cada etapa.

Figura 1 - meteorologia_job.kjb

Transformação "dados_meteorologicos.ktr"

Esta transformação tem como objetivo integrar, transformar e exportar dados meteorológicos obtidos a partir da API *WeatherAPI*.

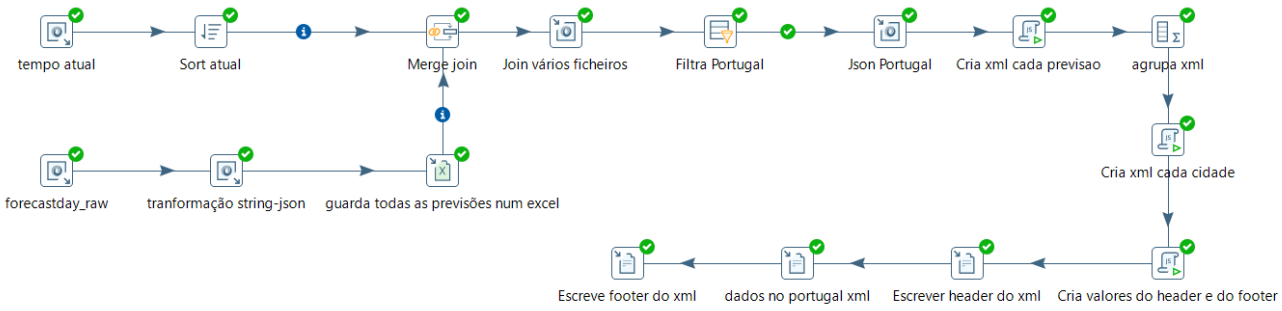

Figura 2 - dados_meteorologicos.ktr

A transformação recebe dados de diversas cidades espalhadas por todo o mundo, e guarda os dados devidamente organizados em tempo real e as previsões meteorológicas de até 3 dias, estes dados estão em flows diferentes.
O kettle guarda em csv todas as previsões de todas as cidades e depois junta as duas tabelas utilizando um *Merge join* ficando cada dia com uma previsão e com os dados em tempo real da cidade e extrai para um ficheiro json.
Ela filtra todas as previsões de cidades portuguesas e extrai para um um ficheiro Json e para um ficheiro XML todas as previsões devidamente formatadas.

Input de dados em tempo real

Figura 3 - Input de dados em tempo real

Os dados meteorológicos estão todos com cidades em ficheiros diferentes numa pasta chamada “dados”.

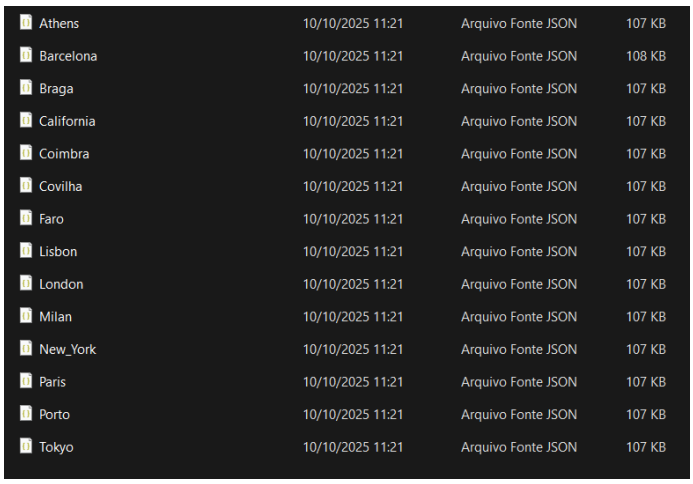

Figura 4 - Pasta "dados"

No *JSON input* “tempo atual” ele corre todos os ficheiros Json dentro da pasta dados e guarda os dados mais relevantes.

Figura 5 - Corre todos os ficheiros Json em dados

#	Name	Path	Type
1	Cidade	$.location.name	String
2	Pais	$.location.country	String
3	LocalTime	$.location.localtime	String
4	TempAtual	$.current.temp_c	Number
5	CondicaoAtual	$.current.condition.text	String
6	HumidadeAtual	$.current.humidity	Number
7	VelocidadeVento	$.current.wind_kph	Number

Figura 6 - Dados Atuais

No passo “Sort atual” organiza por cidades de ordem ascendente

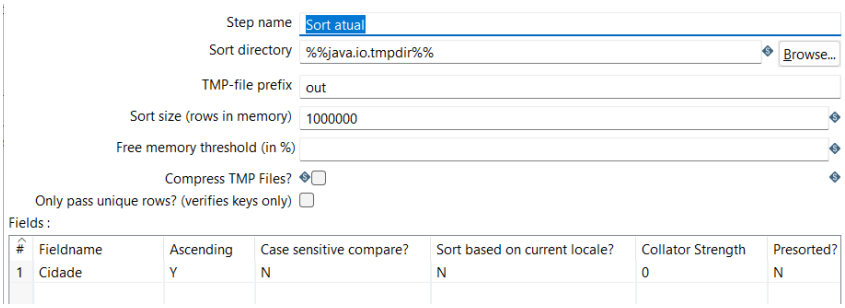

Figura 7 - Sort atual

Input de dados de previsões

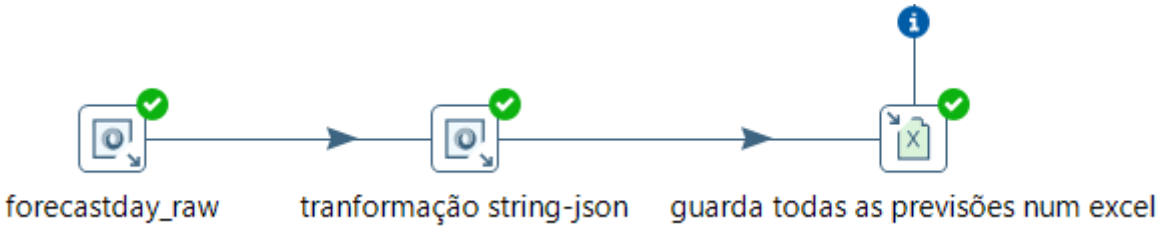

Figura 8 - Input de dados de previsões

A maneira de aceder aos dados das previsões é igual á dos dados atuais, mudando apenas os campos recebidos.

1	forecastday_raw	forecast.forecastday	String
2	Cidade	$.location.name	String
3	Pais	$.location.country	String

Figura 9 - Dados de previsões

Para organizar os dados das previsões, tive de guardar um campo com o array de json puro, para no passo seguinte processar esse array e guardar os dados em rows diferentes.

Figura 10 - Recebe dados do passo anterior

#	Name	Path	Type
1	data	$.[*].date	String
2	temp_media_c	$.[*].day.avgtemp_c	Number
3	max_vento_kph	$.[*].day.maxwind_kph	Number
4	previsao	$.[*].day.condition.text	String
5	humidade_media	$.[*].day.avghumidity	Number
6	chance_chuva	$.[*].day.daily_chance_of_rain	Number

Figura 11 - Dados extraidos do array

Após receber os dados das previsões todas, guarda tudo num ficheiro CSV com os dados:

#	Name	Type
1	Cidade	String
2	Pais	String
3	data	String
4	temp_media_c	Number
5	max_vento_kph	Number
6	previsao	String
7	humidade_media	Number
8	chance_chuva	Number

Figura 12 - Dados de csv_previsoes.csv

Sendo o resultado algo como isto:

Cidade	Pais	data	temp_med	max_vento	previsao	humidade_	chance_chuva
Athens	Greece	2025-10-08	15,9	15,8	Patchy rain nearby	58	82
Athens	Greece	2025-10-09	17,1	11,9	Sunny	57	0
Athens	Greece	2025-10-10	19,4	10,1	Sunny	56	0
Barcelona	Spain	2025-10-08	21,9	11,5	Patchy rain nearby	71	86
Barcelona	Spain	2025-10-09	21,9	13,3	Moderate rain	68	87
Barcelona	Spain	2025-10-10	21,1	30,6	Patchy rain nearby	64	84
Braga	Portugal	2025-10-08	18	12,2	Partly Cloudy	76	0
Braga	Portugal	2025-10-09	19,6	10,4	Sunny	64	0
Braga	Portugal	2025-10-10	19,7	15,1	Sunny	48	0
California	United Sta	2025-10-08	21,1	24,1	Sunny	24	0
California	United Sta	2025-10-09	19,3	13,3	Sunny	25	0
California	United Sta	2025-10-10	20,8	22,3	Partly Cloudy	53	0
Coimbra	Portugal	2025-10-08	17,7	16,6	Cloudy	74	0
Coimbra	Portugal	2025-10-09	19	16,9	Sunny	68	0
Coimbra	Portugal	2025-10-10	20,3	14,4	Sunny	48	0

Figura 13 - Resultado csv_previsoes.csv

“Merge Join” e “Filtra Portugal”

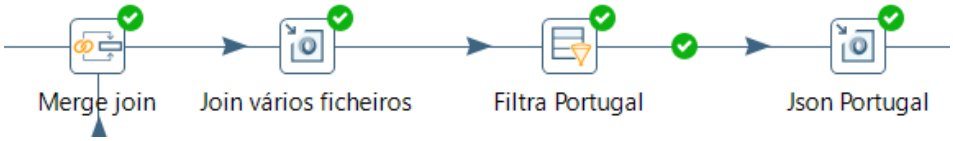

Figura 14 - “Merge Join” e “Filtra Portugal”

O *Merge Join* junta os dados atuais com as previsões organizando pelo campo “Cidade”.

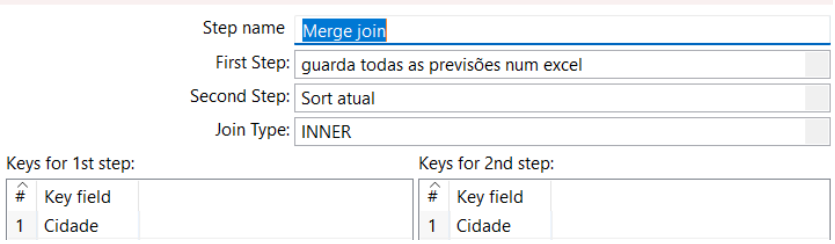

Figura 15 - Merge Join

Depois de juntar os dados, guarda-os num ficheiro Json composto por um array, juntando estes dados:

#	Fieldname	Element name
1	Cidade	Cidade
2	Pais	Pais
3	LocalTime	LocalTime
4	TempAtual	TempAtual
5	CondicaoAtual	CondicaoAtual
6	HumidadeAtual	HumidadeAtual
7	VelocidadeVento	VelocidadeVento
8	data	data
9	temp_media_c	temp_media_c
1..	max_vento_kph	max_vento_kph
1..	previsao	previsao
1..	humidade_media	humidade_media
1..	chance_chuva	chance_chuva

Figura 16 – Dados join.json

Sendo o resultado algo como isto:

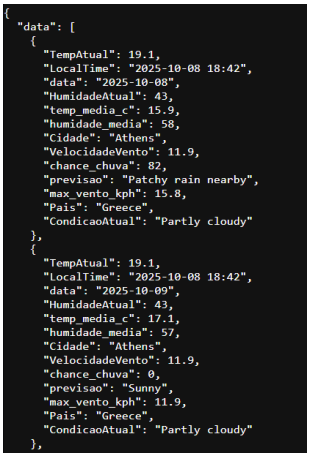

Figura 17 - Conteudo join.json

Depois utiliza um *Filter Rows* que filtra todos os dados para só existirem dados de Portugal.

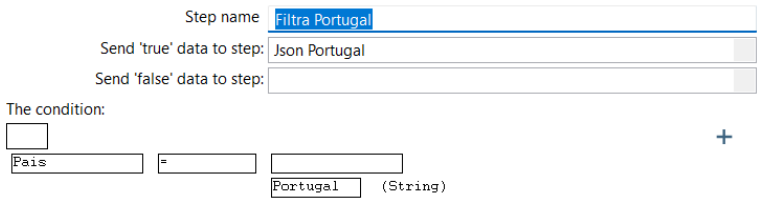

Figura 18 - Filtra Portugal

Após a filtragem, guarda os dados todos num Json outra vez, seguindo a mesma estrutura de quando guardou depois do *Merge Join*.

Criação do XML estruturado

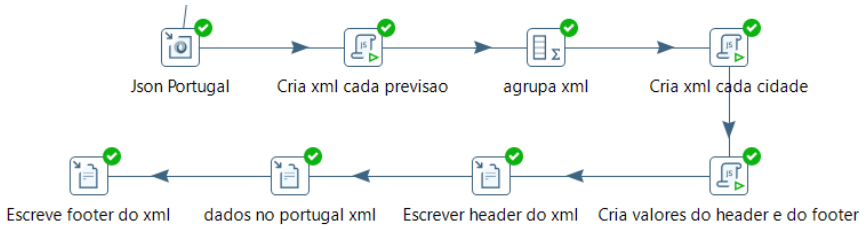

Figura 19 - Criação do XML estruturado

Esta é a parte do projeto com mais componentes complexos, pois o maior desafio foi fazer um XML com uma estrutura correta.
Para fazer isso, comecei por estruturar cada previsão, visto que cada linha da tabela correspondia a uma previsão. Para isso, utilizei um *Modified Javascript Value* que criou uma estrutura em XML e guardou em cada linha numa coluna chamada “previsao_xml”.

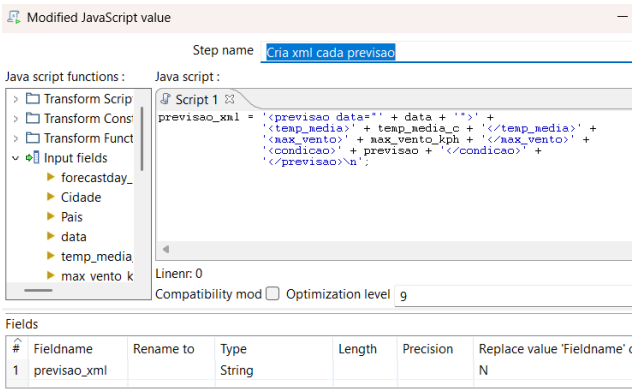

Figura 20 - Criação do campo previsao_xml

De seguida, utilizei um *Group By* para juntar as strings pertencentes aos campos “previsao_xml” e agrupar por cidades, guardando essas previsões todas no campo “previsoes_concat”.

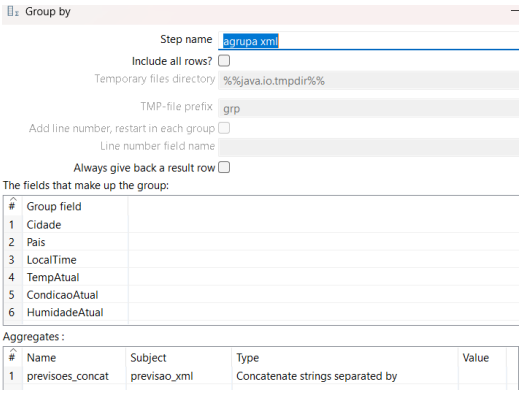

Figura 21 - Agrupa previsões por cidade

Após juntar todas as previsões de uma cidade, utiliza mais uma vez um *Modified Javascript Value* para desta vez criar a parte do xml com os dados atuais da cidade e concatenar os dados já em XML das previsões, guardando tudo no campo “cidade_xml”.

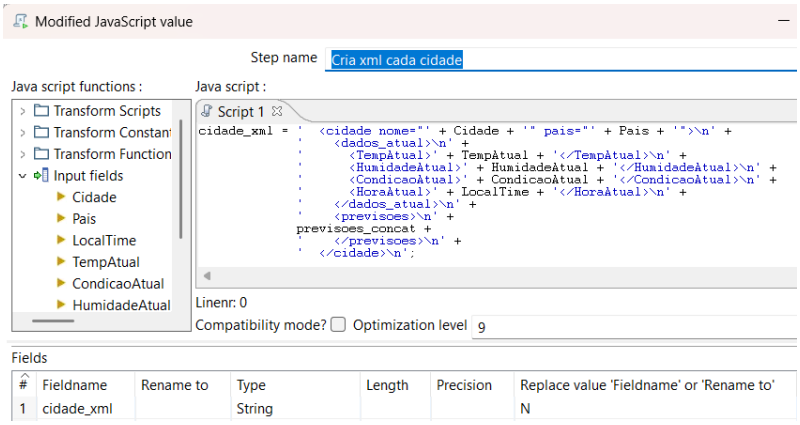

Figura 22 - Criação do cidade_xml

Como os ficheiros XML têm de ter um *header* e como os dados das cidades irão ficar dentro da tag “<cidades></cidades>”, utilizei mais um *Modified Javascript Value* que guarda num campo o valor de *header* e de *footer*.

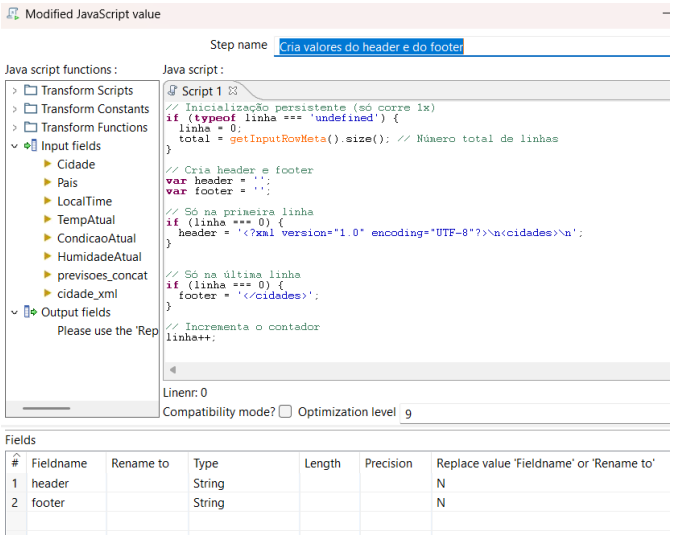

Figura 23 - Header e footer do XML

Depois falta a parte de construir realmente o ficheiro XML. Começando por criar o *header* previamente guardado no campo *header*.

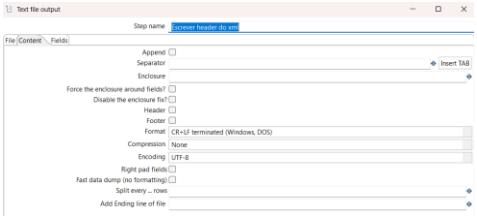

Figura 24 - Escrever header do xml (content)

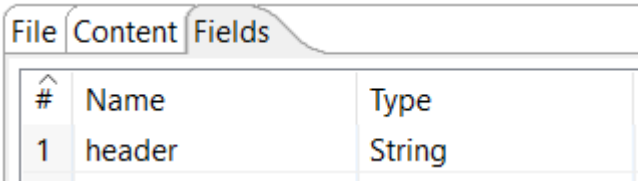

Figura 25 - Escrever header do xml (fields)

De seguida, insere os dados da “cidade_xml”.

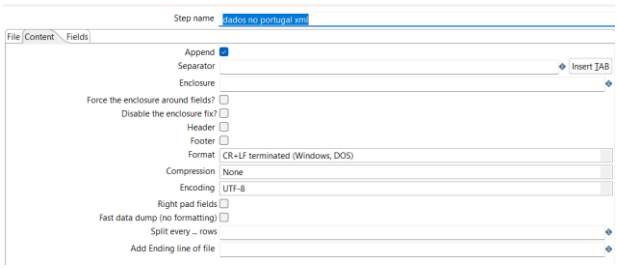

Figura 26 - Escrever cidade_xml (content)

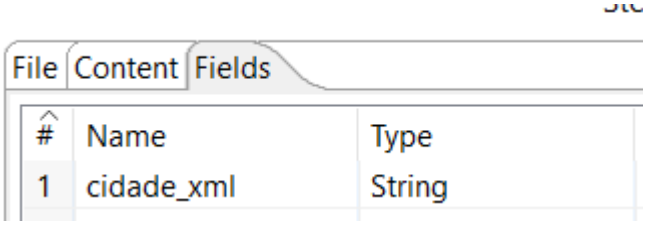

Figura 27 - Escrever cidade_xml (content)

E, por fim, escreve o fecho da tag *cidades* com a mesma configuração de contente do passo anterior.

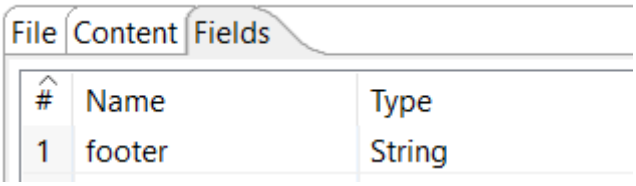

Figura 28 - Fecha a tag "cidades"

Vídeo com demonstração

Figura 29 - Video Demo

Conclusão

Com este trabalho foi possível compreender como funciona um processo de integração de dados utilizando o Pentaho Data Integration (Kettle).

Durante o desenvolvimento, foram aplicados passos de extração, transformação e carregamento (ETL), como a leitura de ficheiros JSON, a junção e agrupamento de dados e a escrita de um resultado final num formato personalizado e estruturado. Também foi criado um Job para automatizar todo o processo, garantindo que a transformação é executada de forma simples e controlada.

Em resumo, o trabalho permitiu praticar os conceitos de integração de sistemas e demonstrar como o Kettle pode ser usado para tratar e organizar dados de diferentes fontes de forma eficiente.