

|  |
| --- |
| Trabalho Prático Nº1  Integração de Sistemas de Informação |
|  |
| Da autoria de:  Paulo Costa Nº29851 |



Índice

[Introdução 4](#_Toc211607466)

[Ferramentas Utilizadas 5](#_Toc211607467)

[Job “meteorologia\_job.kjb” 5](#_Toc211607468)

[Transformação “dados\_meteorologicos.ktr” 5](#_Toc211607469)

[Input de dados em tempo real 6](#_Toc211607470)

[Input de dados de previsões 7](#_Toc211607471)

[“Merge Join” e “Filtra Portugal” 8](#_Toc211607472)

[Criação do XML estruturado 10](#_Toc211607473)

[Vídeo com demonstração 12](#_Toc211607474)

[Conclusão 13](#_Toc211607475)

Índice de Figuras

[Figura 1 - meteorologia\_job.kjb 5](#_Toc211607476)

[Figura 2 - dados\_meteorologicos.ktr 5](#_Toc211607477)

[Figura 3 - Input de dados em tempo real 6](#_Toc211607478)

[Figura 4 - Pasta "dados" 6](#_Toc211607479)

[Figura 5 - Corre todos os ficheiros Json em dados 7](#_Toc211607480)

[Figura 6 - Dados Atuais 7](#_Toc211607481)

[Figura 7 - Sort atual 7](#_Toc211607482)

[Figura 8 - Input de dados de previsões 7](#_Toc211607483)

[Figura 9 - Dados de previsões 7](#_Toc211607484)

[Figura 10 - Recebe dados do passo anterior 8](#_Toc211607485)

[Figura 11 - Dados extraidos do array 8](#_Toc211607486)

[Figura 12 - Dados de csv\_previsoes.csv 8](#_Toc211607487)

[Figura 13 - Resultado csv\_previsoes.csv 8](#_Toc211607488)

[Figura 14 - “Merge Join” e “Filtra Portugal” 8](#_Toc211607489)

[Figura 15 - Merge Join 9](#_Toc211607490)

[Figura 16 – Dados join.json 9](#_Toc211607491)

[Figura 17 - Conteudo join.json 9](#_Toc211607492)

[Figura 18 - Filtra Portugal 9](#_Toc211607493)

[Figura 19 - Criação do XML estruturado 10](#_Toc211607494)

[Figura 20 - Criação do campo previsao\_xml 10](#_Toc211607495)

[Figura 21 - Agrupa previsões por cidade 10](#_Toc211607496)

[Figura 22 - Criação do cidade\_xml 11](#_Toc211607497)

[Figura 23 - Header e footer do XML 11](#_Toc211607498)

[Figura 24 - Escrever header do xml (content) 11](#_Toc211607499)

[Figura 25 - Escrever header do xml (fields) 11](#_Toc211607500)

[Figura 26 - Escrever cidade\_xml (content) 12](#_Toc211607501)

[Figura 27 - Escrever cidade\_xml (content) 12](#_Toc211607502)

[Figura 28 - Fecha a tag "cidades" 12](#_Toc211607503)

[Figura 29 - Video Demo 12](#_Toc211607504)

# 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.

Uma imagem com diagrama, file, Tipo de letra

Os conteúdos gerados por IA podem estar incorretos.

Figura - meteorologia\_job.kjb

# Transformação “dados\_meteorologicos.ktr”

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

Uma imagem com captura de ecrã, file, diagrama, texto

Os conteúdos gerados por IA podem estar incorretos.

Figura - 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

Uma imagem com diagrama, captura de ecrã, file, Tipo de letra

Os conteúdos gerados por IA podem estar incorretos.

Figura - Input de dados em tempo real

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

Uma imagem com texto, captura de ecrã, menu

Os conteúdos gerados por IA podem estar incorretos.

Figura - Pasta "dados"

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

|  |  |
| --- | --- |
| Figura - Corre todos os ficheiros Json em dados | Figura - Dados Atuais |

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

Uma imagem com texto, captura de ecrã, número, file

Os conteúdos gerados por IA podem estar incorretos.

Figura - Sort atual

## Input de dados de previsões

Uma imagem com diagrama, file, Tipo de letra, captura de ecrã

Os conteúdos gerados por IA podem estar incorretos.

Figura - Input de dados de previsões

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

Uma imagem com texto, captura de ecrã, Tipo de letra, file

Os conteúdos gerados por IA podem estar incorretos.

Figura - 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 - Recebe dados do passo anterior | Figura - Dados extraidos do array |

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

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Os conteúdos gerados por IA podem estar incorretos.

Figura - Dados de csv\_previsoes.csv

Sendo o resultado algo como isto:

Uma imagem com texto, captura de ecrã, número, Tipo de letra

Os conteúdos gerados por IA podem estar incorretos.

Figura - Resultado csv\_previsoes.csv

## “Merge Join” e “Filtra Portugal”

Uma imagem com diagrama, file, captura de ecrã, Tipo de letra

Os conteúdos gerados por IA podem estar incorretos.

Figura - “Merge Join” e “Filtra Portugal”

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

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Os conteúdos gerados por IA podem estar incorretos.

Figura - Merge Join

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

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Os conteúdos gerados por IA podem estar incorretos.

Figura – Dados join.json

Sendo o resultado algo como isto:

Uma imagem com texto, captura de ecrã, Tipo de letra, menu

Os conteúdos gerados por IA podem estar incorretos.

Figura - Conteudo join.json

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

Uma imagem com texto, captura de ecrã, file, Tipo de letra

Os conteúdos gerados por IA podem estar incorretos.

Figura - 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

Uma imagem com texto, diagrama, captura de ecrã, file

Os conteúdos gerados por IA podem estar incorretos.

Figura - 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”.

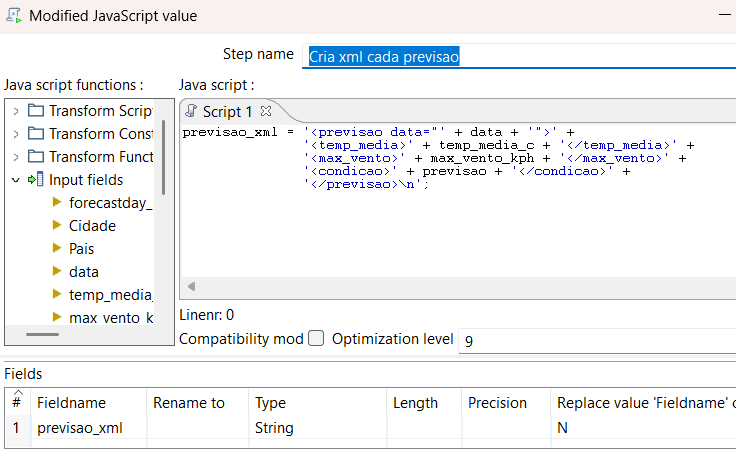


Figura - Criação do campo previsao\_xml

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”.

Uma imagem com texto, captura de ecrã, número, Tipo de letra

Os conteúdos gerados por IA podem estar incorretos.

Figura - 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”.

Uma imagem com texto, captura de ecrã, software, ecrã

Os conteúdos gerados por IA podem estar incorretos.

Figura - 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*.

Uma imagem com texto, captura de ecrã, ecrã, software

Os conteúdos gerados por IA podem estar incorretos.

Figura - 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 - Escrever header do xml (content) | Figura - Escrever header do xml (fields) |

De seguida, insere os dados da “cidade\_xml”.

|  |  |
| --- | --- |
| Figura - Escrever cidade\_xml (content) | Figura - Escrever cidade\_xml (content) |

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

Uma imagem com texto, captura de ecrã, Tipo de letra, file

Os conteúdos gerados por IA podem estar incorretos.

Figura - Fecha a tag "cidades"

# Vídeo com demonstração

Uma imagem com padrão, texto, coser

Os conteúdos gerados por IA podem estar incorretos.

Figura - 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.