Relatório ISI

No.27975 – Diogo Manuel Pereira Abreu
No.27966 – Paulo Ricardo Fernandes Gonçalves

Licenciatura em Engenharia Sistemas Informáticos

3ºano

Barcelos | outubro, 2025

Lista de Abreviaturas e Siglas

ETL	Extract, Transform, Load (Extração, Transformação e Carga)
CSV	Comma-Separated Values
XML	Extensible Markup Language
ER	Expressões Regulares
PDI	Pentaho Data Integration
kettle	Pentaho Data Integration (nome popular da ferramenta)
QR Code	Quick Response Code
MM/DD/YYYY	Mês/Dia/Ano (Formato de data)
dd/mm/yyyy	Dia/Mês/Ano (Formato de data)

Índice de Figuras

Índice

1. Introdução

O presente documento constitui o Relatório Técnico referente ao trabalho desenvolvido no âmbito da Unidade Curricular de ISI - Integração de Sistemas de Informação da Licenciatura em Engenharia de Sistemas Informáticos, no 3º ano, pelos alunos Diogo Manuel Pereira Abreu (No.27975) e Paulo Ricardo Fernandes Gonçalves (No.27966).

Este relatório documenta a conceção, implementação e demonstração de um sistema de Extração, Transformação e Carga (ETL). O projeto surge da necessidade de converter dados de clientes de um ficheiro CSV para o formato XML, aplicando processos cruciais de Data Quality e preparação.

- Os principais objetivos do trabalho são:

- Demonstrar um fluxo ETL completo utilizando ferramentas como o Pentaho Data Integration e o Knime.

- Cumprir integralmente os critérios do enunciado do projeto.

- Explorar as capacidades robustas de transformação de dados das ferramentas ETL utilizadas.

1. Enquadramento do problema

1.1. Descrição do Cenário

Conversão de dados de clientes de um ficheiro CSV para formato XML, aplicando processos de:

- Limpeza e normalização de dados

- Validação de campos usando expressões regulares

- Visualização de gráficos com o ficheiro JSON

- Geração de jobs

1.2. Objetivos Específicos

- Demonstrar um fluxo ETL completo com o Pentaho e no Knime

- Cumprir os critérios do enunciado

- Explorar as capacidades de transformação da ferramenta

1.3. Organização do trabalho

Ficheiros Incluídos no Repositório (Conteúdo de data/int/, data/input/, e data/output/):

-ISI25_TP01_27975_27966/

- README.md

- doc/ISI25_TP01_27975_27966_doc.pdf - numeroaluno_PlataformaUsada/ (27966 - Knime ou 27975_Pentaho_Data_Integration)

- data/ - transformações e jobs usados para o ETL (data-integration)

- data/input/ - ficheiros com os dados de entrada

- data/output/ - dados de saída... para onde devem ser encaminhados os ficheiros de saída...

2. Projeto Pentaho Kettle

2.1. Ficheiros de Transformações

2.1.1. Transformação de CSV para XML

Figura 1 – Transformação da passagem de csv para xml

O processo de transformação de CSV para XML mostrado na imagem é um passo fundamental em fluxos de trabalho de integração de dados e ETL (Extract, Transform, Load). Ele representa a conversão de dados de um formato de ficheiro simples e tabular (CSV) para um formato hierárquico e estruturado (XML).

- **Entrada (Input):** O processo começa com a leitura de um ficheiro CSV (Comma-Separated Values). O CSV é um formato de texto simples onde os dados são organizados em linhas, e os valores em cada linha são separados por um delimitador, como uma vírgula ou ponto e vírgula. A primeira linha do ficheiro geralmente contém os nomes das colunas (cabeçalhos).

- **Transformação (Transformation):** O software de ETL, como o que parece ser usado na imagem, lê o ficheiro CSV e realiza a conversão. Cada linha do ficheiro CSV é transformada em um elemento XML, e os dados de cada coluna se tornam subelementos ou atributos dentro desse elemento. É nesse ponto que se define a estrutura hierárquica do XML, como os nomes dos elementos raiz e dos registos.

- **Saída (Output):** O resultado do processo é um ficheiro XML (Extensible Markup Language), que utiliza tags para descrever a estrutura e o significado dos dados. A saída em XML é um formato mais versátil e legível por máquinas, especialmente útil para sistemas que precisam de dados com uma estrutura mais complexa ou para integração com outras aplicações.

Esse tipo de transformação é comum quando é necessário alimentar um sistema que só aceita dados em XML, ou quando a estrutura de dados precisa ser mais detalhada do que a que um ficheiro CSV pode fornecer.

2.1.2. Transformação de criação de tabelas

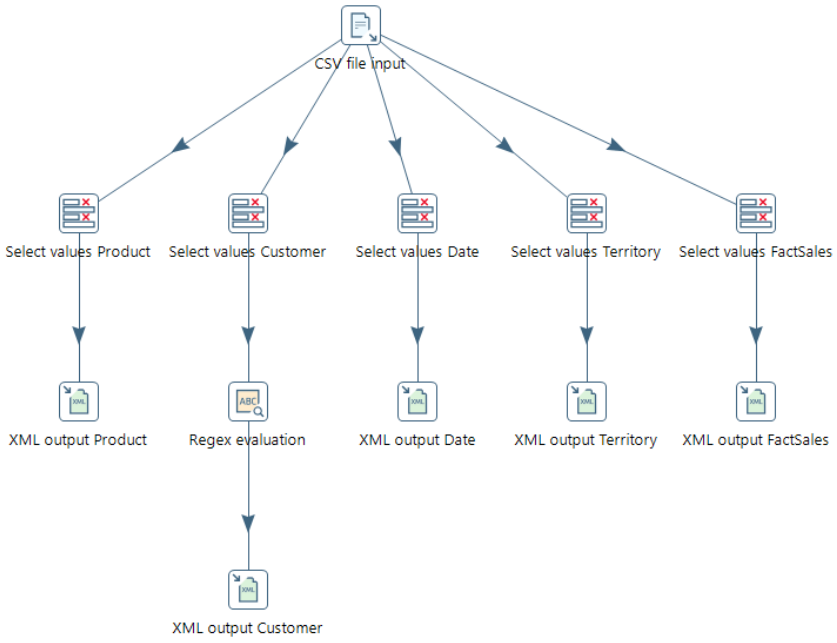

Figura 2 – Transformação da Criação de Tabelas

Com base na imagem, esta transformação representa um processo de ETL (Extração, Transformação e Carga) que converte dados de um único ficheiro CSV para múltiplos ficheiros XML, designando-se assim de tabelas

Aqui está a descrição de cada etapa do processo:

1. Entrada de Dados (CSV file input)

O processo começa com a leitura de um ficheiro CSV (Comma-Separated Values). Este ficheiro atua como a fonte de dados, contendo informações em formato tabular que serão processadas.

2. Seleção e Transformação de Campos (Select values)

Os dados do ficheiro CSV são divididos em várias ramificações para processamento paralelo, com cada uma a lidar com um conjunto específico de dados:

- **Select values Product:** Seleciona os dados relacionados ao produto. Selecionado apenas os campos PRODUCTCODE, PRODUCTLINE e MSRP

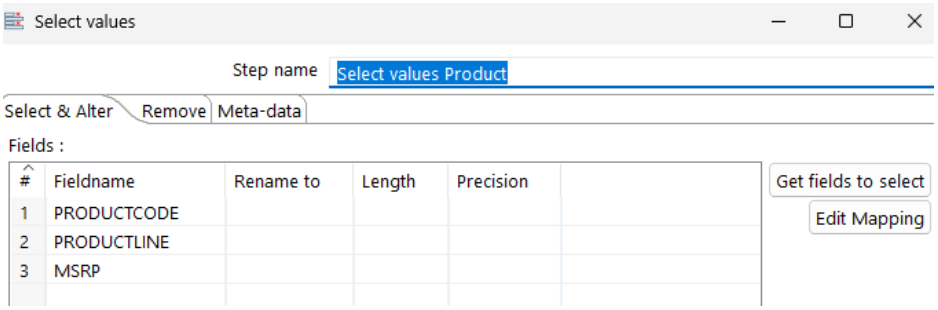

Figura 3 – Selecionar os campos para criar a tabela Product

- **Select values Customer:** Seleciona os dados dos clientes. Selecionado apenas os campos CUSTOMERNAME, PHONE, ADDRESSLINE1, ADDRESSLINE2, CITY, STATE, POSTALCODE, COUNTRY.

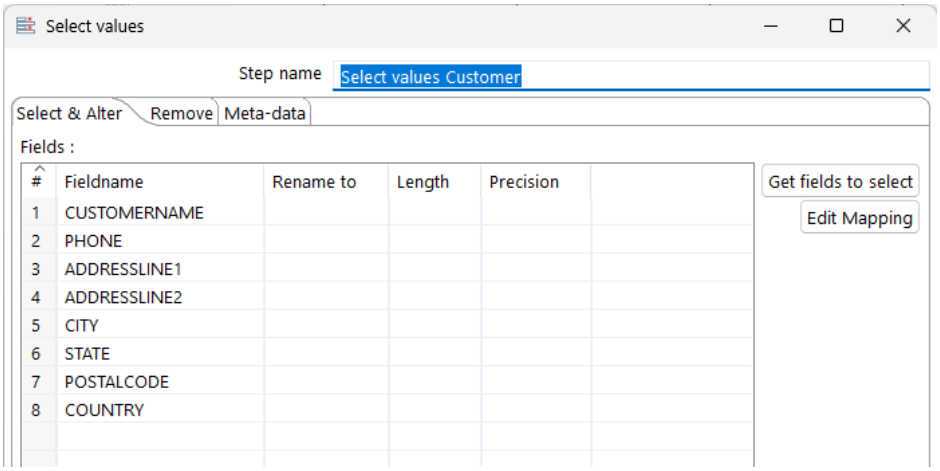

Figura 4 - Selecionar os campos para criar a tabela Customer

- **Select values Date:** Seleciona os dados de data. Selecionando apenas os campos ORDERDATE, QTR_ID, MONTH_ID, YEAR_ID

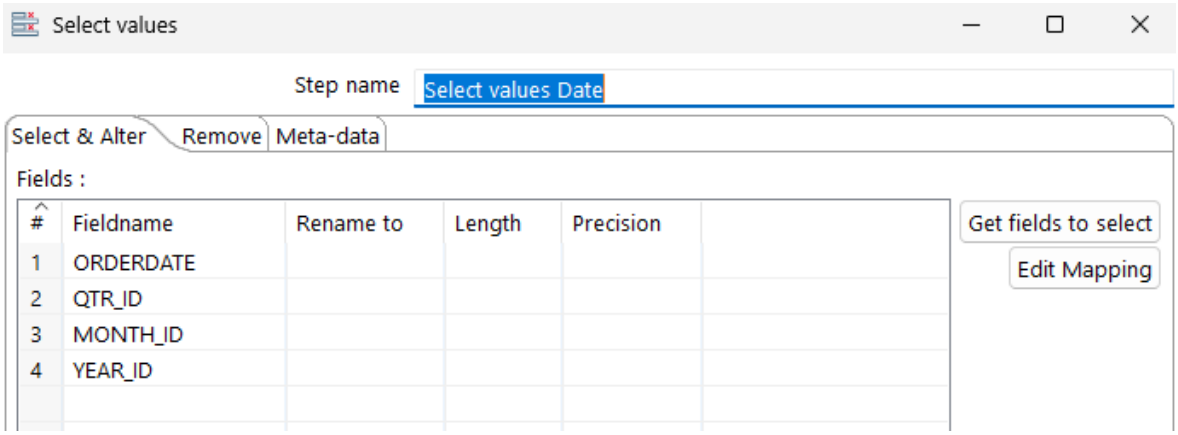

Figura 5 - Selecionar os campos para criar a tabela Date

- **Select values Territory:** Seleciona os dados de território. Selecionando apenas os campos COUNTRY, TERRITORY

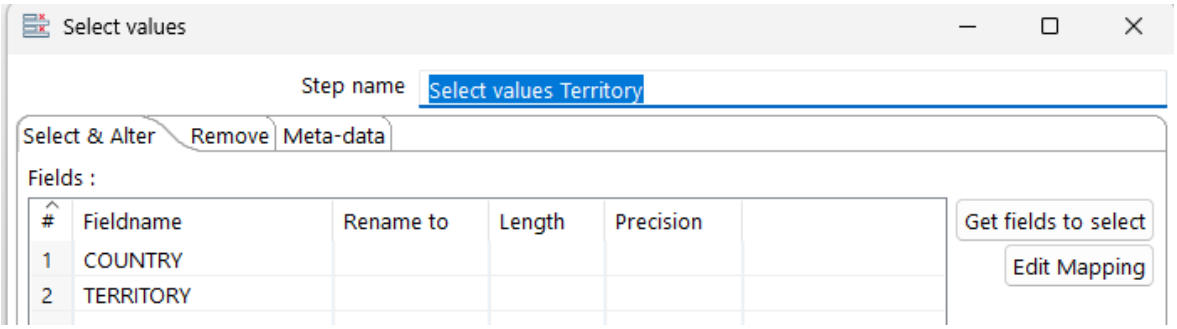

Figura 6 - Selecionar os campos para criar a tabela Territory

- **Select values FactSales:** Seleciona os dados de vendas. Selecionando apenas os campos SALES, CUSTOMERNAME, PRODUCTCODE, QTR_ID, MONTH_ID, YEAR_ID, TERRITORY, ORDERNUMBER, QUANTITYORDERED, STATUS, DEALSIZE.

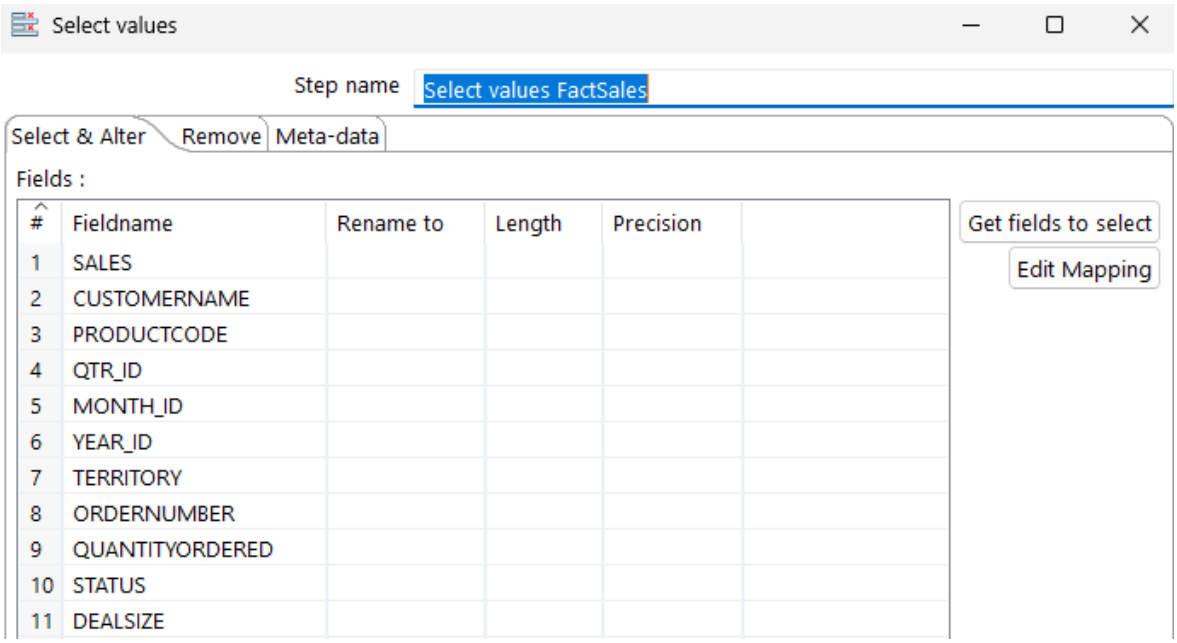

Figura 7 - Selecionar os campos para criar a tabela FactSales

3. Validação de Dados (Regex evaluation)

No ramo de "Select values Customer", é aplicada uma etapa de "Regex evaluation".

Esta etapa é crucial para a qualidade dos dados. Conforme especificado, a expressão regular **^[0-9]+** é usada para validar um campo, como o do número de telefone.

Esta regex garante que o valor do campo começa com um ou mais dígitos numéricos, filtrando ou rejeitando quaisquer registos que não sigam este padrão e, assim, assegurando a integridade dos dados antes da sua exportação.

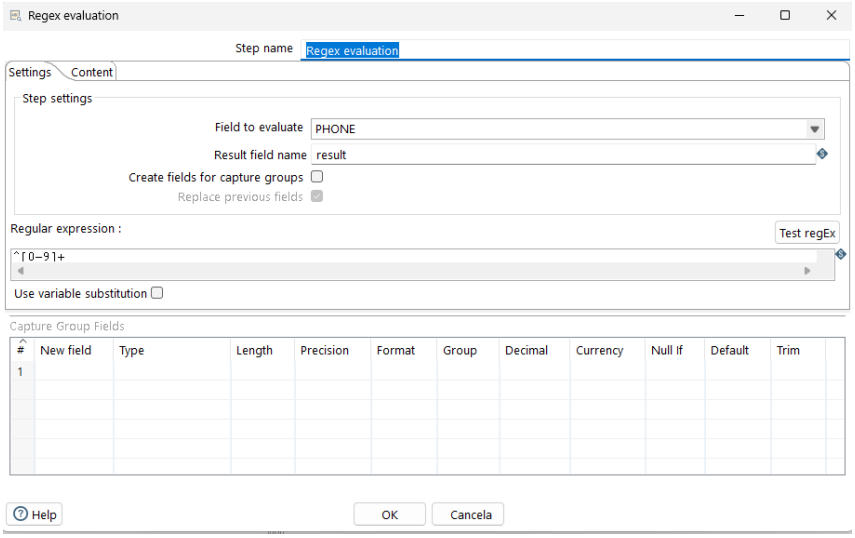

Figura 8 – Instrução regex

4. Saída de Dados (XML output)

Após a seleção e, em alguns casos, a validação dos dados, cada ramificação converte os dados filtrados para o formato XML (Extensible Markup Language), um formato hierárquico e estruturado que é amplamente utilizado para troca de dados entre sistemas.

- **XML output Product:** Cria um ficheiro XML com os dados de produto.
- **XML output Customer:** Cria um ficheiro XML com os dados de cliente validados.
- **XML output Date:** Cria um ficheiro XML com os dados de data.
- **XML output Territory:** Cria um ficheiro XML com os dados de território.
- **XML output FactSales:** Cria um ficheiro XML com os dados de vendas.

Em resumo, o diagrama ilustra um fluxo de trabalho que extrai dados de um ficheiro de origem CSV, os divide e os transforma, realizando a validação de um campo específico com uma expressão regular, e, finalmente, gera múltiplos ficheiros XML como resultado final, prontos para serem usados em outros sistemas.

2.1.3. Transformação de ordenação de tabelas

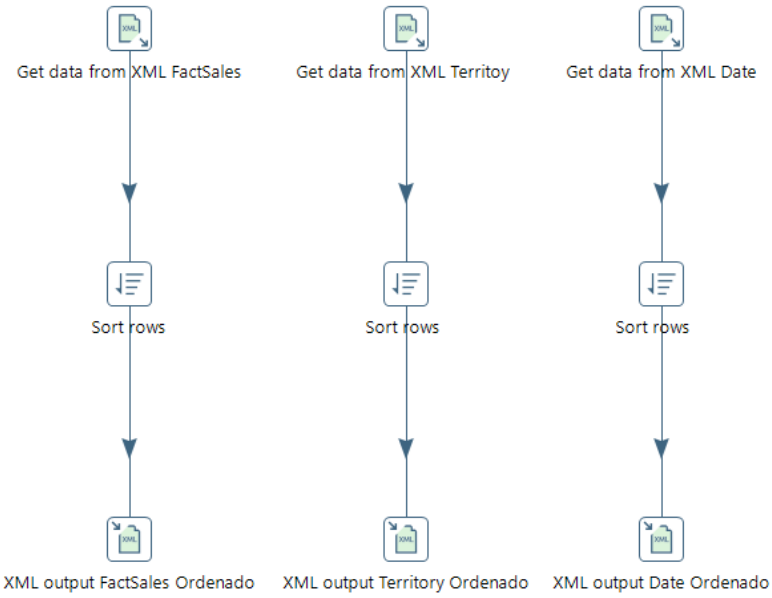

Figura 9 – Transformação da ordenação dos campos das tabelas FactSales, Territory, Date

A transformação apresentada ilustra três fluxos de trabalho de processamento de dados paralelos, com o objetivo principal de extrair dados de ficheiros XML existentes, ordená-los e, em seguida, exportá-los para novos ficheiros XML ordenados.

Cada fluxo é independente e segue os seguintes passos:

1. **Entrada de Dados XML:**

1. Get data from XML FactSales: Extrai dados de vendas de um ficheiro XML de origem.

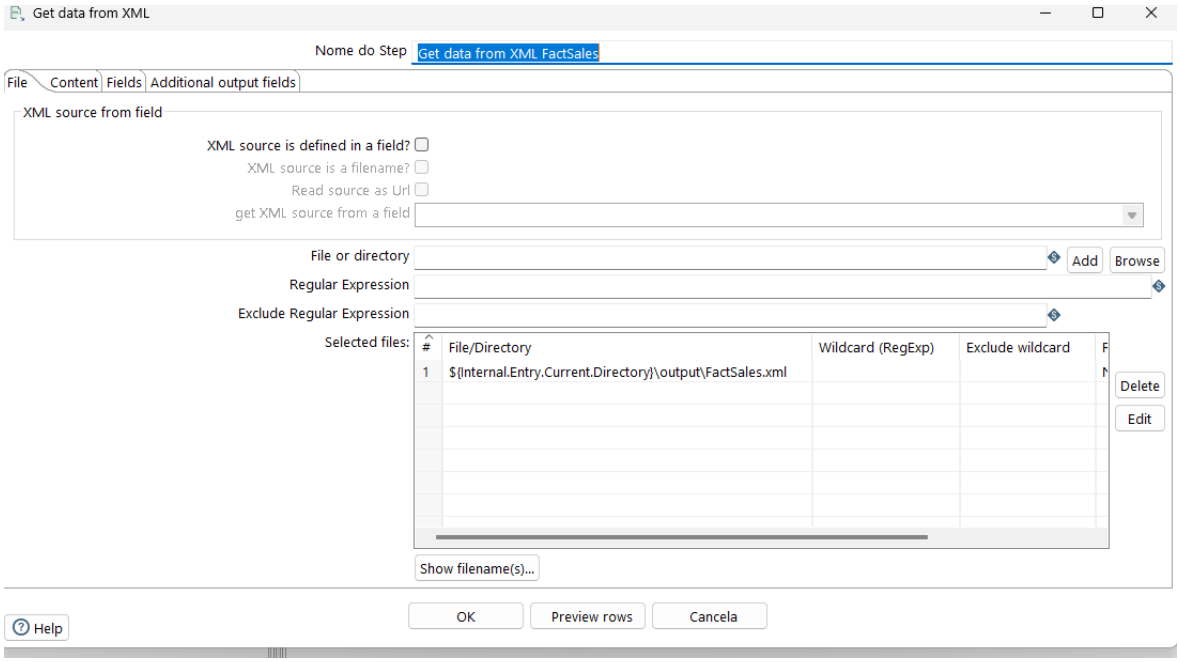

Figura 10 – Get do XML FacSale

2. Get data from XML Territoy: Extrai dados de território de um ficheiro XML de origem.

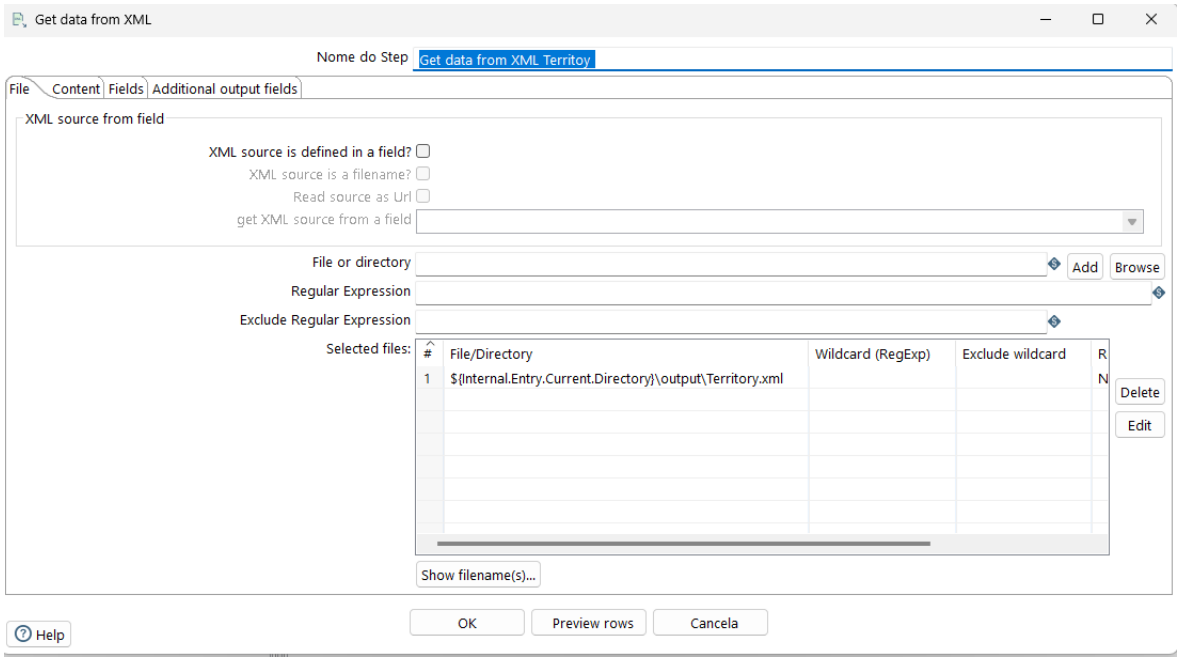

Figura 11 - Get do XML Territory

3. Get data from XML Date: Extrai dados de data de um ficheiro XML de origem.

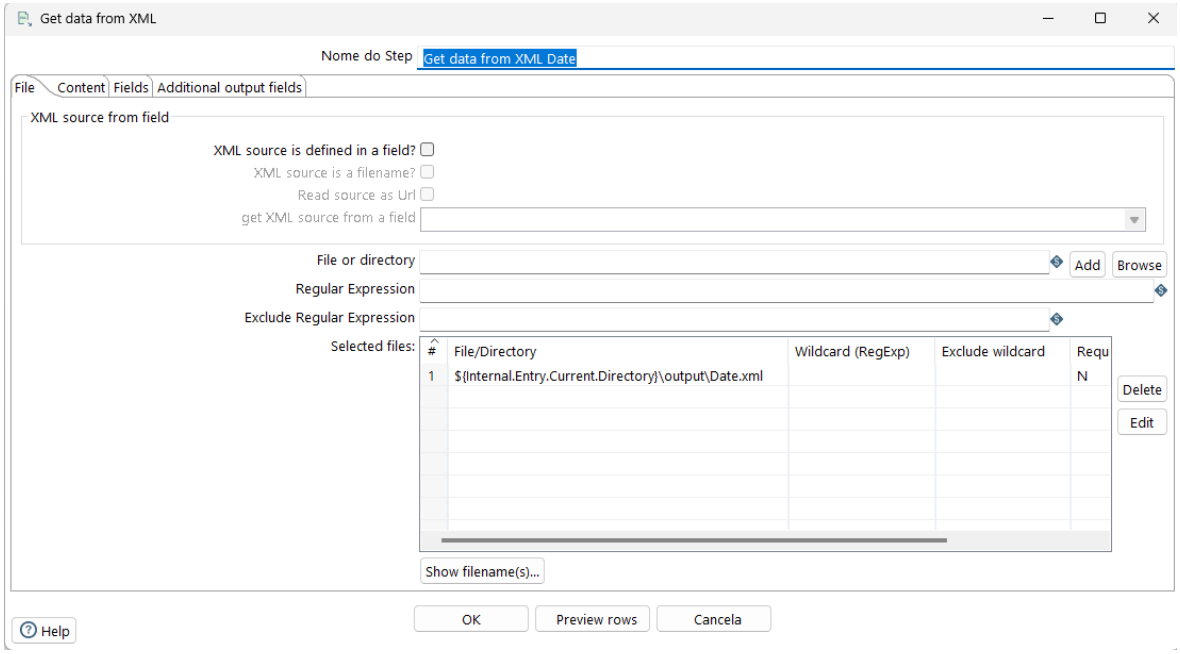

Figura 12 - Get do XML Date

2. Ordenação de Dados:

1. Sort rows: Esta etapa processa os dados extraídos, organizando-os em uma ordem específica (por exemplo, ascendente ou descendente) com base em um ou mais campos.

No caso de FactSales podemos ver como está a ser ordenado

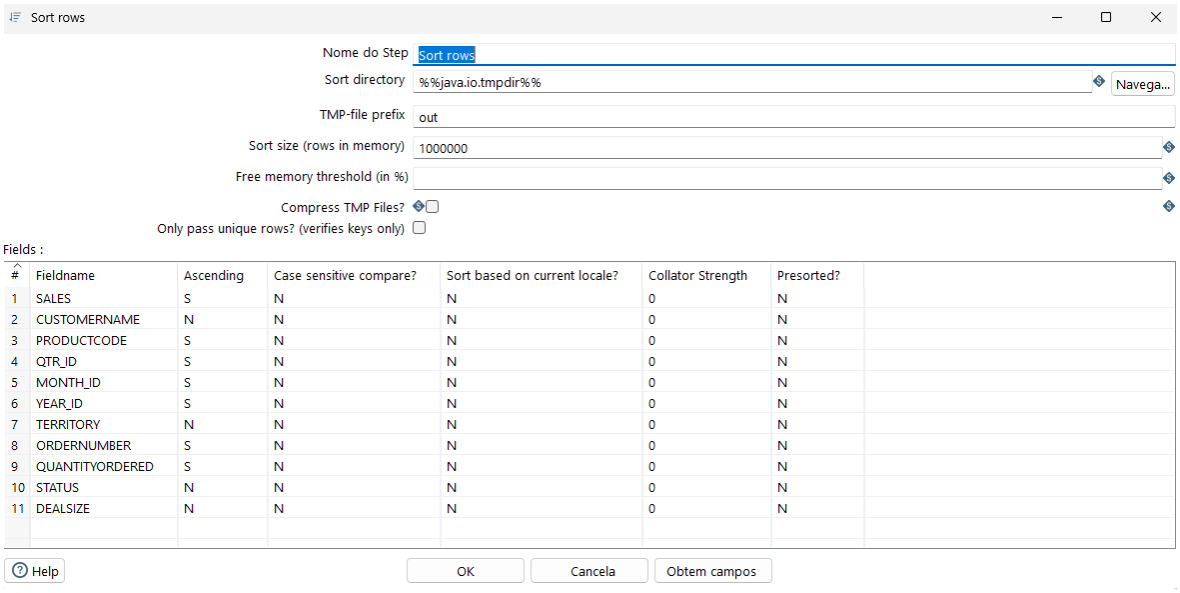

Figura 13 – Ordenação das linhas da tabela FactSales

No caso de Territory podemos ver como está a ser ordenado

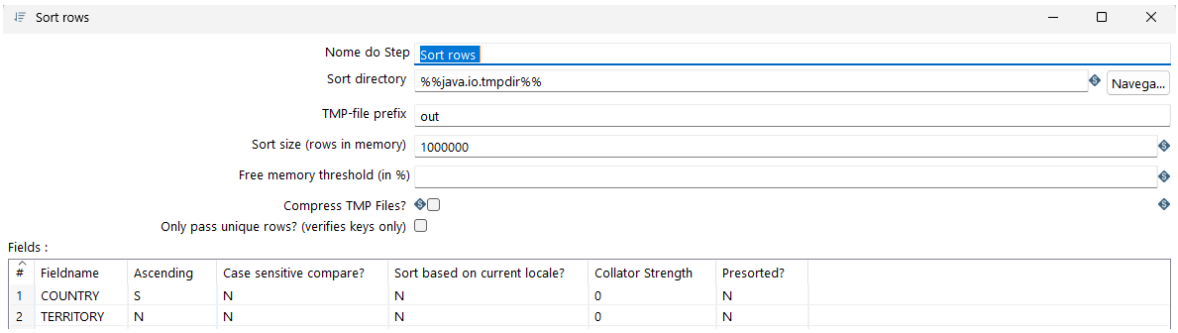

Figura 14 - Ordenação das linhas da tabela Territory

No caso de Date podemos ver como está a ser ordenado

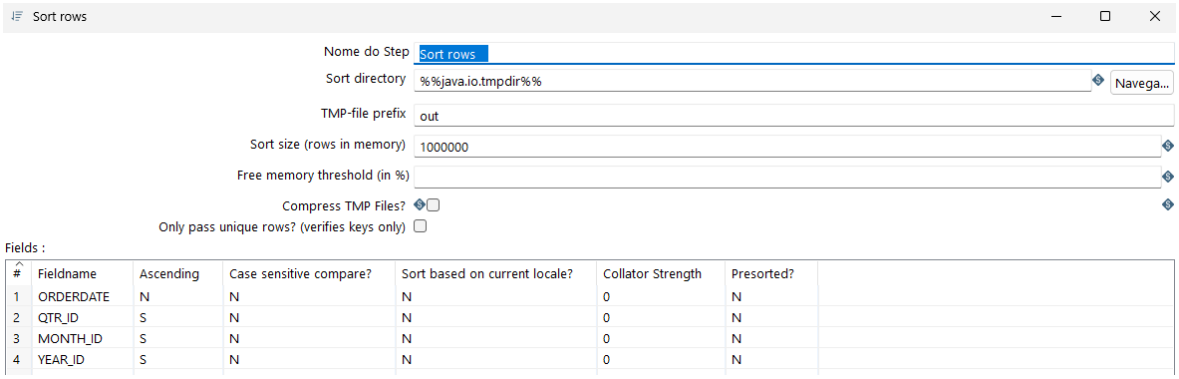

Figura 15 - Ordenação das linhas da tabela Date

3. Saída de Dados XML Ordenados:

1. XML output FactSales Ordenado: Gera um novo ficheiro XML que contém os dados de vendas já ordenados.

2. XML output Territory Ordenado: Gera um novo ficheiro XML com os dados de território ordenados.

3. XML output Date Ordenado: Gera um novo ficheiro XML com os dados de data ordenados.

Em sintese, a transformação processa de forma independente três conjuntos de dados distintos (vendas, território e datas) lidos de ficheiros XML, aplica uma operação de ordenação a cada um e salva o resultado em novos ficheiros XML, mantendo a integridade dos dados e a sua estrutura original.

2.1.4. Transformação de preparação dos dados date em formato mm/dd/yyyy

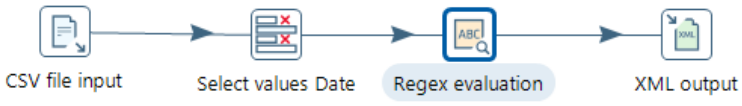

Figura 16 – Transformação do CSV em XML com o formato de data diferente e uma instrução regex

O diagrama apresentado descreve um fluxo de trabalho de processamento de dados linear e simples, com o objetivo de extrair, filtrar, validar e exportar dados de data.

O processo pode ser descrito da seguinte forma:

1. **Entrada de Dados (CSV file input):** O fluxo de trabalho começa com a extração de dados de um ficheiro de texto no formato CSV (Comma-Separated Values).

2. **Seleção de Campos (Select values Date):** De entre todos os dados lidos do ficheiro CSV, esta etapa foca-se na seleção e processamento dos campos que contêm informações de data.

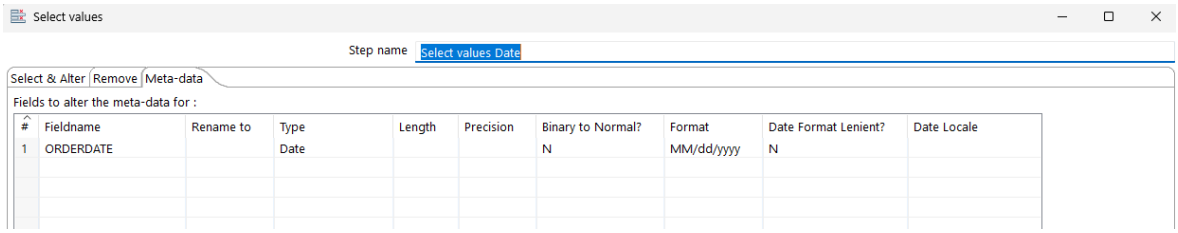

Figura 17 – Alteração do formato date

3. **Validação de Dados (Regex evaluation):** Os dados de data selecionados são submetidos a uma validação por meio de uma expressão regular (Regex). Esta etapa é essencial para garantir que o formato das datas está correto e consistente com as regras de negócio definidas.

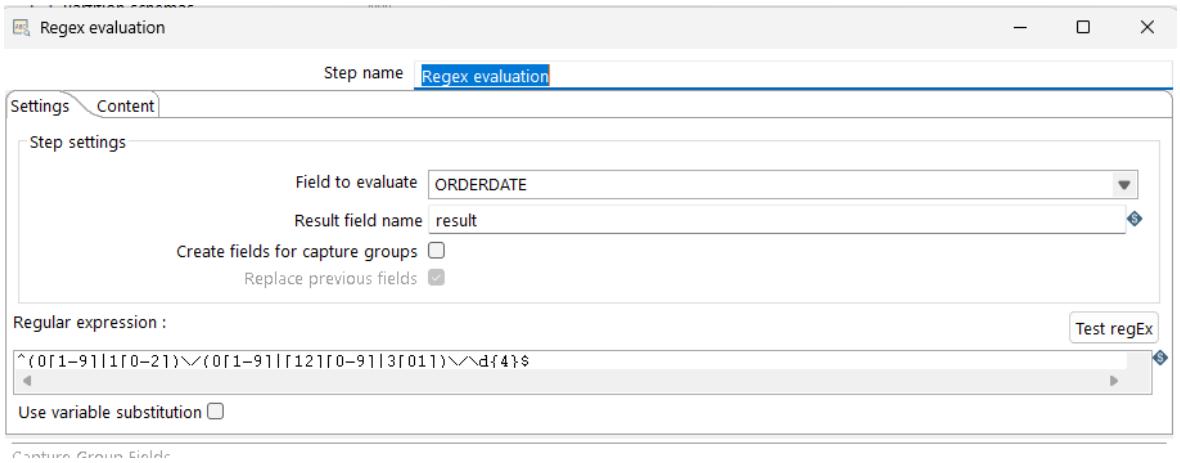

Figura 18 – Instrução Regex

A expressão regular ^(0[1-9]|1[0-2])\/(0[1-9]|[12][0-9]|3[01])\/\d{4}$ é utilizada para validar uma data no formato **MM/DD/YYYY** (Mês/Dia/Ano).

A sua função pode ser explicada da seguinte forma, analisando cada parte:

- ^: Indica o início da string. A correspondência deve começar no primeiro caractere.
- (0[1-9]|1[0-2]): Esta parte valida o **mês (MM)**.
    - 0[1-9]: Corresponde aos meses de 01 a 09.
    - |: Atua como um operador "ou".
    - 1[0-2]: Corresponde aos meses de 10, 11 ou 12.
    - Isto assegura que o mês seja um valor válido entre 01 e 12.
- \/: Corresponde ao caractere barra (/), que atua como separador entre o mês e o dia. O caractere de barra tem que ser "escapado" com a barra invertida, pois é um caractere especial em regex.
- (0[1-9]|[12][0-9]|3[01]): Esta parte valida o **dia (DD)**.
    - 0[1-9]: Corresponde aos dias de 01 a 09.
    - |: Operador "ou".
    - [12][0-9]: Corresponde aos dias de 10 a 29.
    - |: Operador "ou".

- 3[01]: Corresponde aos dias 30 ou 31.
- Isto garante que o dia seja um valor entre 01 e 31.

- \/: Corresponde novamente ao caractere barra (/), separando o dia do ano.
- \d{4}: Esta parte valida o **ano (YYYY)**.
  - \d: Corresponde a qualquer dígito numérico (0-9).
  - {4}: Indica que o dígito anterior (\d) deve ocorrer exatamente 4 vezes.
  - Isto assegura que o ano seja um número de 4 dígitos.
- $: Indica o fim da string. A correspondência deve terminar no último caractere.

Esta expressão regular verifica se uma string inteira corresponde ao formato MM/DD/YYYY, com valores de mês de 01-12 e valores de dia de 01-31. No entanto, não valida a lógica de datas, por exemplo, se um mês tem 31 dias ou 28/29.

4. **Saída de Dados (XML output):** Por fim, os dados de data que foram validados com sucesso são convertidos e exportados para um ficheiro no formato XML (Extensible Markup Language), que é um formato hierárquico e estruturado.

Em suma, este fluxo de trabalho extrai dados de um ficheiro CSV, filtra as colunas de data, valida-as com uma expressão regular e, por fim, gera um ficheiro XML com as datas validadas.

2.1.1. Transformação de XML para JSON

Figura 19 - Transformação do CSV em JSON

O diagrama apresentado ilustra um fluxo de trabalho de processamento de dados linear e simples. O objetivo é extrair dados de um ficheiro XML e criar o ficheiro JSON para ver a página com os gráficos.

2.2. Página com os Gráficos

Com base nas três imagens apresento a página conhecida como **Dashboard de Análise de Vendas** abrangente, que apresenta uma visão geral do desempenho de vendas de uma empresa ao longo do tempo (2003-2005). Vamos decompor a informação presente em cada ecrã:

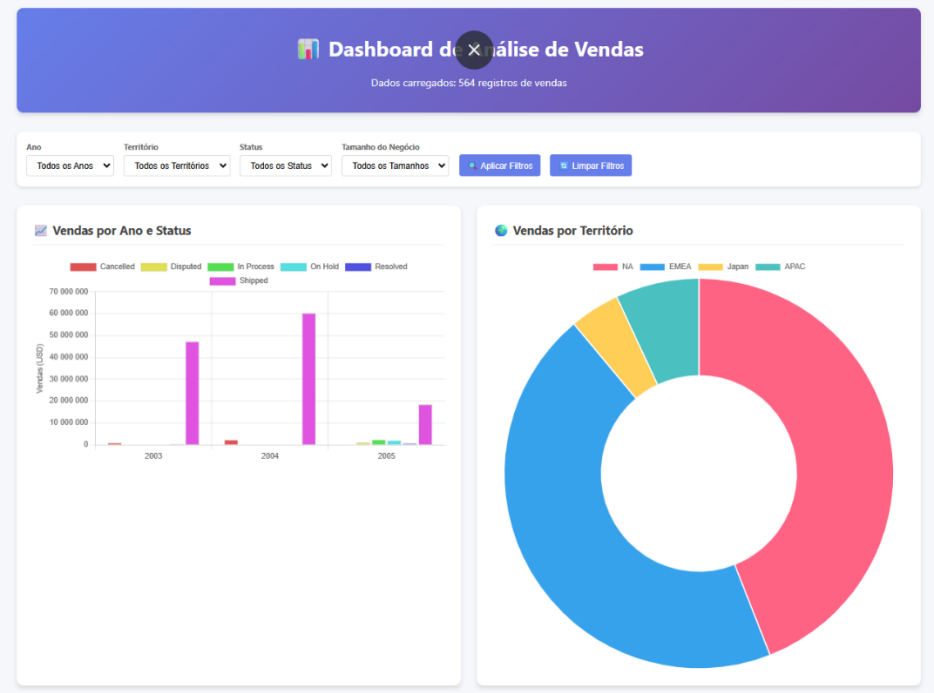

Figura 20 - Parte Inicial da Dashboard

- **Dados Carregados:** O conjunto de dados tem **564 registos de vendas**.
- **Vendas por Ano:** O gráfico de barras mostra as vendas totais (em USD) para os anos de **2003, 2004 e 2005**. Houve um crescimento significativo de 2003 para 2004, seguido de uma estabilização ou ligeiro aumento em 2005.
- **Status dos Pedidos:** Um gráfico de "donut" ou pizza mostra a distribuição dos pedidos por estado (ex: "Canceled", "In Process", "Resolved"). Isto ajuda a perceber a eficiência do processo de vendas.
- **Vendas por Território:** Outro gráfico mostra a distribuição das vendas por regiões geográficas: **NA** (América do Norte), **EMEA** (Europa, Médio Oriente e África), **Japan** e **APAC** (Ásia-Pacífico). A NA parece ser a maior região.

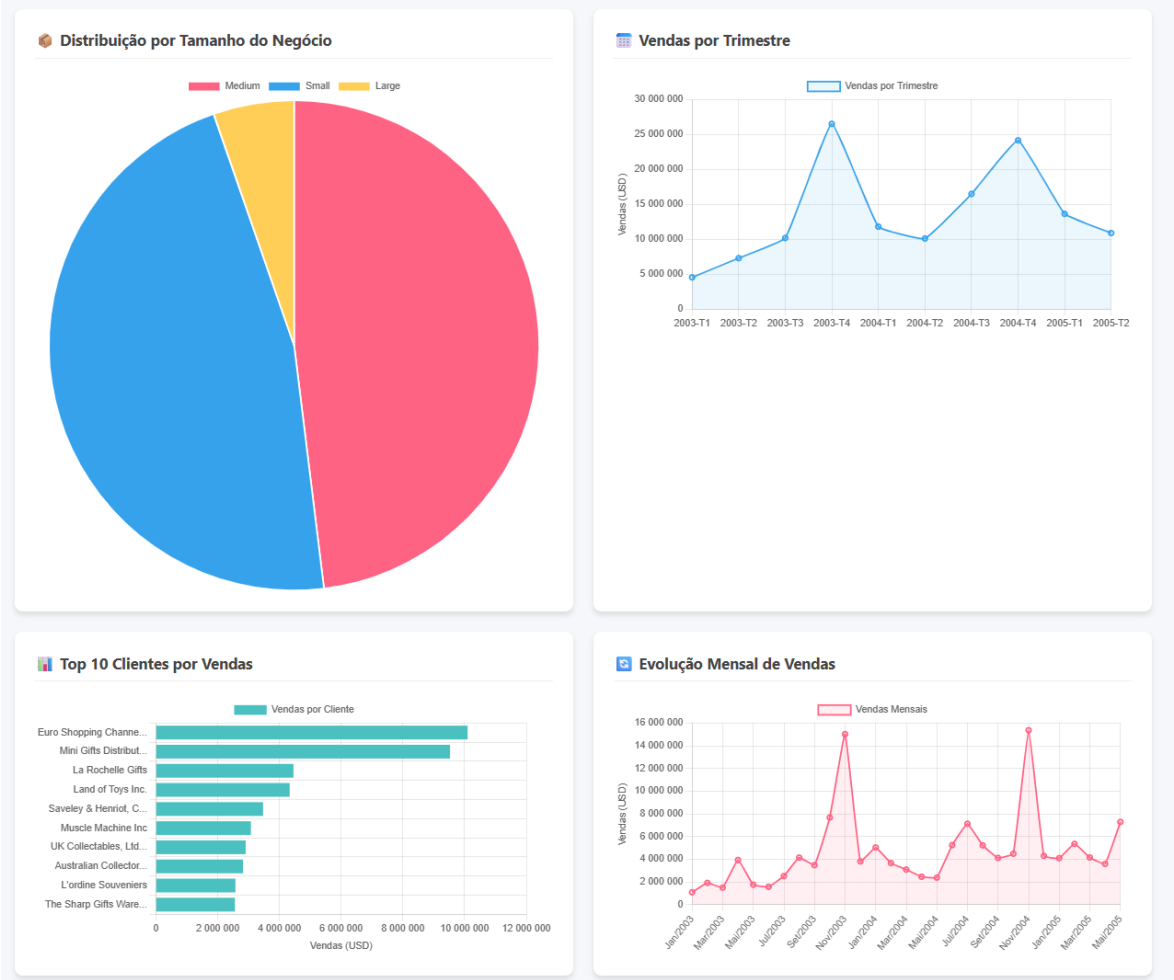

Figura 21 - Parte do meio da Dashboard

Esta parte foca-se na performance ao longo do tempo e nos principais clientes.

- **Distribuição por Tamanho do Negócio:** A empresa categoriza os seus clientes como "Medium", "Small" e "Lapp" (provavelmente um erro para "Large" - Grande).
- **Top 10 Clientes por Vendas:** Um gráfico de barras horizontal lista os clientes que mais contribuíram para o volume de vendas. O **"Euro Shopping Channel..."** é claramente o maior cliente, com vendas muito superiores aos restantes.
- **Vendas por Trimestre:** Um gráfico de linhas mostra a evolução trimestral das vendas desde o 1º trimestre de 2003 (2003-T1) até ao 2º trimestre de 2005 (2005-T2). Revela uma **sazonalidade clara**, com picos consistentes no 4º trimestre de cada ano (T4 - provavelmente associado às vendas de final de ano).
- **Evolução Mensal de Vendas:** Um gráfico de barras mais detalhado que confirma a tendência de sazonalidade, mostrando os picos mensais dentro de cada trimestre.

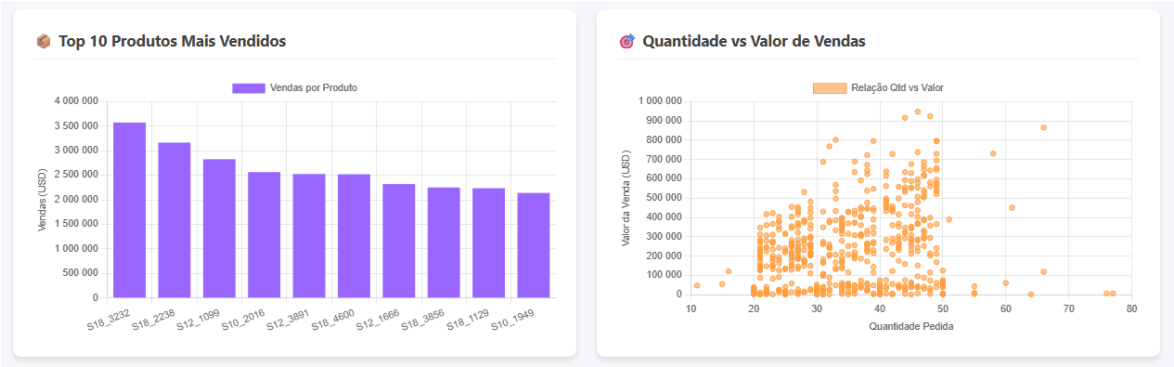

Figura 22 - Parte final da Dashboard

Esta parte é dedicado ao desempenho dos produtos.

- **Top 10 Produtos Mais Vendidos:** Um gráfico de barras vertical mostra os produtos (identificados por códigos como "S18_3232") que geraram mais receita. O primeiro produto tem vendas significativamente mais altas que os outros.
- **Quantidade vs Valor de Vendas:** Este é um gráfico de dispersão que explora a relação entre o número de unidades vendidas ("Quantidade Pedida") e o valor total gerado ("Valor da Venda"). Isto é crucial para identificar:
    - **Produtos de Alto Valor/Baixo Volume:** Itens caros que vendem poucas unidades (canto superior esquerdo).
    - **Produtos de Alto Volume/Baixo Valor:** Itens baratos que vendem em grande quantidade (canto inferior direito).
    - **Produtos "Ideais":** Itens que vendem bem em quantidade e têm um bom valor (canto superior direito).

2.3. Job

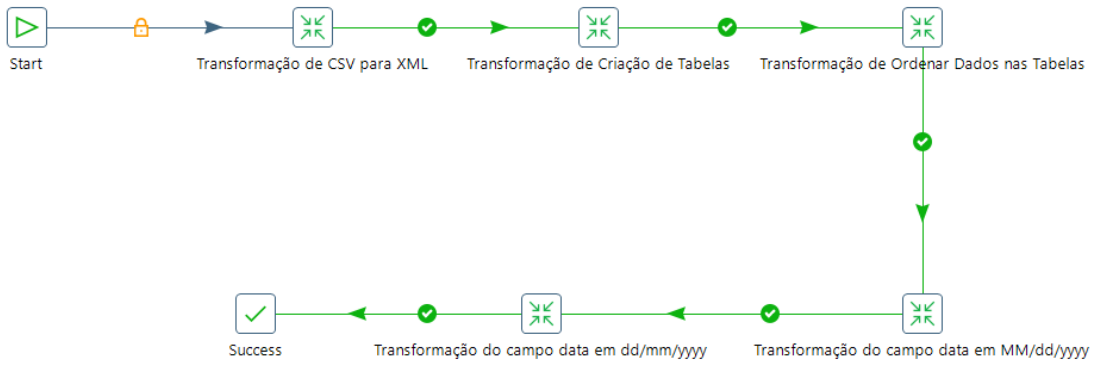

Figura 23 - Job

Este diagrama representa um fluxo de trabalho de processamento de dados, criado com uma ferramenta ETL (Extração, Transformação, Carga) como o Pentaho Data Integration (PDI), também conhecido como Kettle.

A descrição do fluxo de trabalho é a seguinte:

- **Início (Start):** O processo é iniciado.
- **Transformação de CSV para XML:** Os dados de um ficheiro no formato CSV (Comma-Separated Values) são transformados para o formato XML (Extensible Markup Language).
- **Transformação de Criação de Tabelas:** O processo continua com a criação de tabelas, o que pode ser uma etapa para preparar a base de dados para receber os dados transformados.
- **Transformação de Ordenar Dados nas Tabelas:** Os dados nas tabelas são ordenados.
- **Transformação do campo data em MM/dd/yyyy:** O formato do campo de data é alterado para Mês/Dia/Ano.
- **Transformação do campo data em dd/mm/yyyy:** O formato da data é alterado novamente, desta vez para Dia/Mês/Ano.
- **Sucesso (Success):** O processo termina com sucesso após a conclusão de todas as transformações.

3. Projeto KNIME

O Projeto foi divido em quatro partes, mas especificamente 4 job's utilizando *metanodes*, disponibilizados pelo *KNIME*:

- Preparação de dados
- Base de dados
- Email
- Gráficos

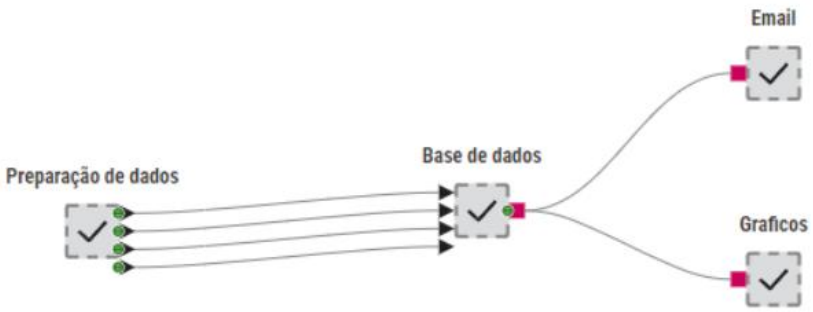

Figura 24 – Estrutura KNIME

3.1. Preparação de dados

3.1.1. Introdução dos dados

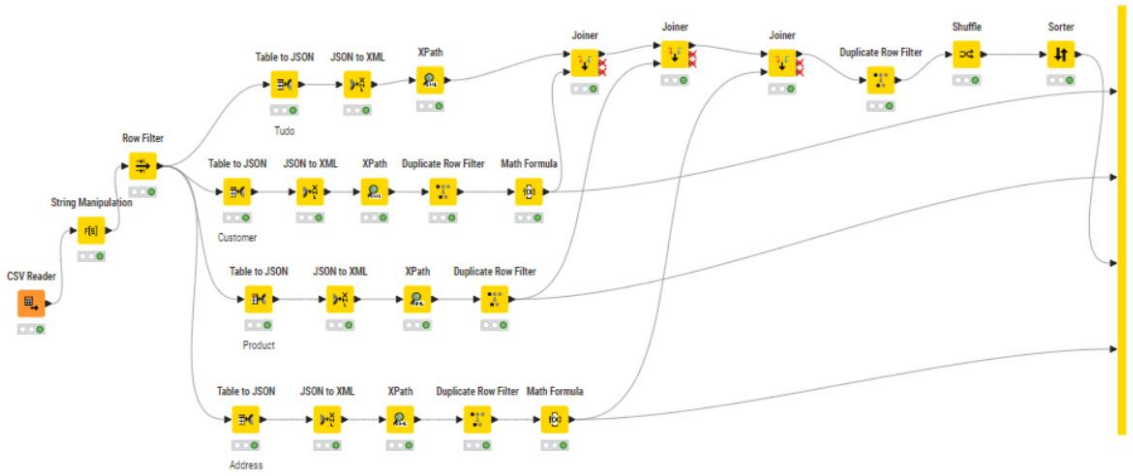

Figura 25 - Fluxo da Preparação de dados

O fluxo começa com a importação dos dados em CSV Atravez de um “CSV Reader”, apenas foi necessário preencher a localização do ficheiro de *input*

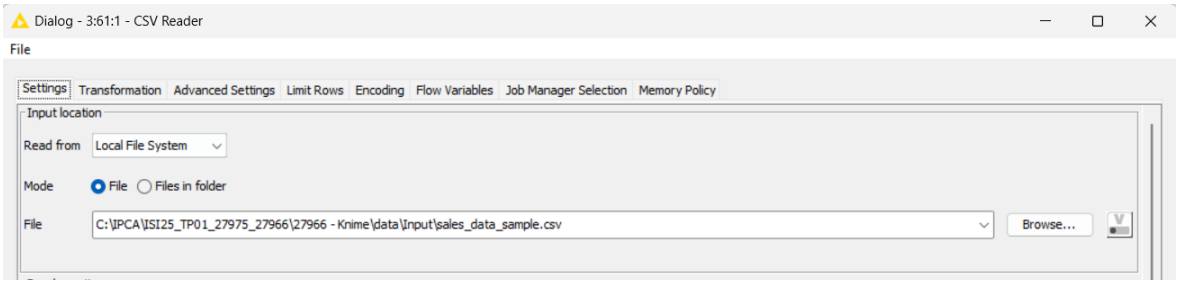

Figura 26 - Configuração CSV Reader

Em seguida foi feita a validação de alguns dados, como o número telefone, para isso utilizei o nó "String Manipulation" com um código em *regex* que apenas permite dados que comecem com + e em seguida tenha apenas números ou espaços:

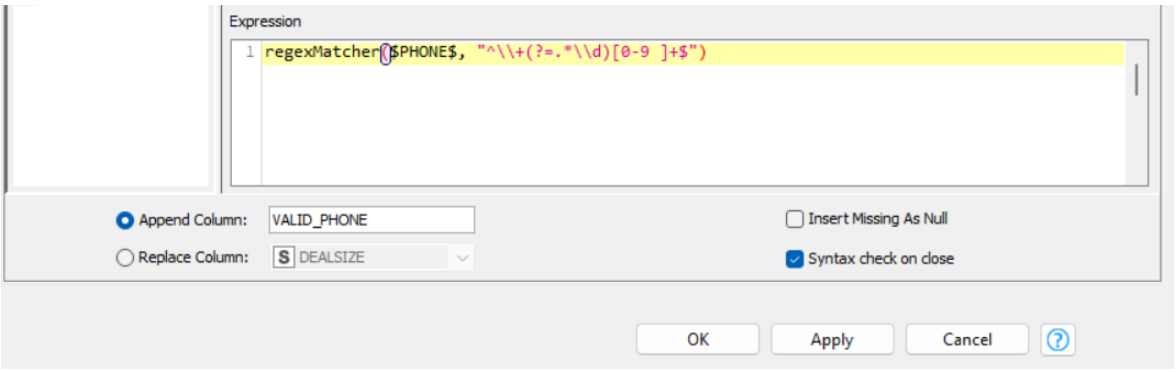

Figura 27 – Regex

A "String Manipulation" cria uma coluna nova "VALID_PHONE" onde, caso o número de telefone respeite o regex vai colocar o valor *true* e caso não esteja em conformidade vai colocar a *false*. No próximo nó, neste caso "Row Filter" vai ser eliminada a linha onde a coluna "VALID_PHONE" estiver com o valor *false*.

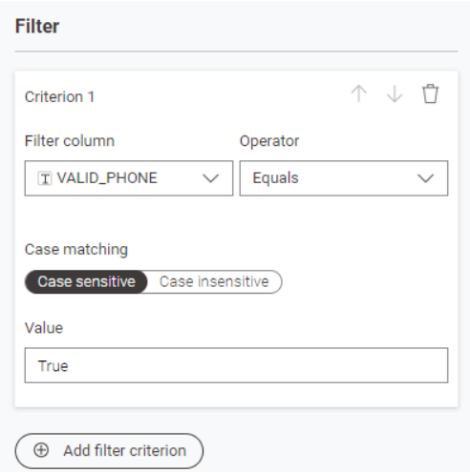

Figura 28 - Row Filter

3.1.2. Normalização

Depois de os dados estarem validados, foi necessário normalizar o ficheiro, pois os dados estavam concentrados em um ficheiro: cliente (Customer), produto (Product), compra (Order) e morada (Address), para isso separei em 4 fluxos diferentes um para cada tabela. Através do nó "Table to JSON" separei os dados do cliente, morada e produto para ficheiros *json* diferentes e mantive a compra com os dados íntegros para futuramente no fluxo conseguir juntar as tabelas com os identificadores(id), em seguida transformei os ficheiros json em xml para poder manipular

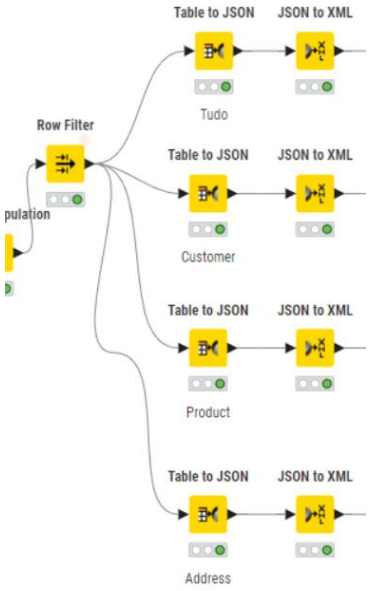

Figura 30 – Normalização

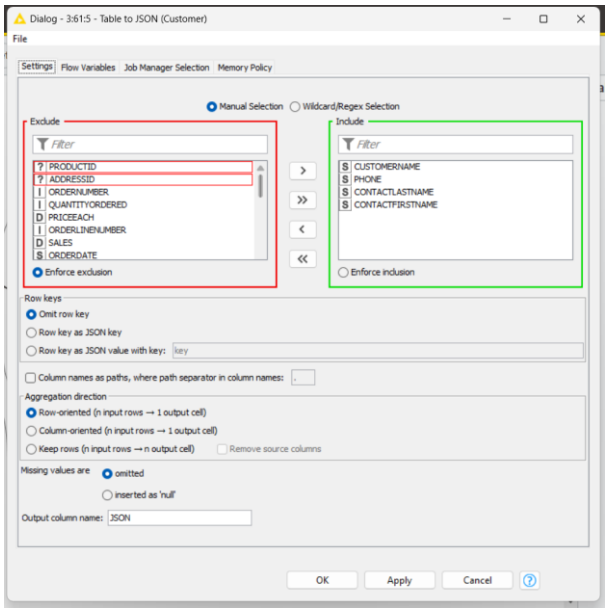

Figura 29 – Table to JSON

Para identificar os campos em um ficheiro XML foi necessário utilizar o nó “XPath”, em seguida eliminei as linhas duplicadas de cada tabela e inseri um ID utilizando o nó ”Math Formula” que, por cada linha criava um id que começava em 1.

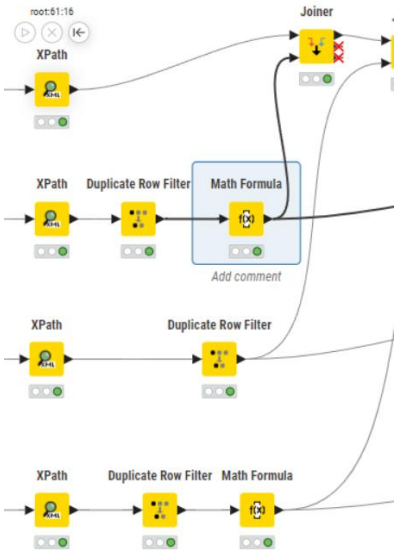

Figura 31 – Identificar

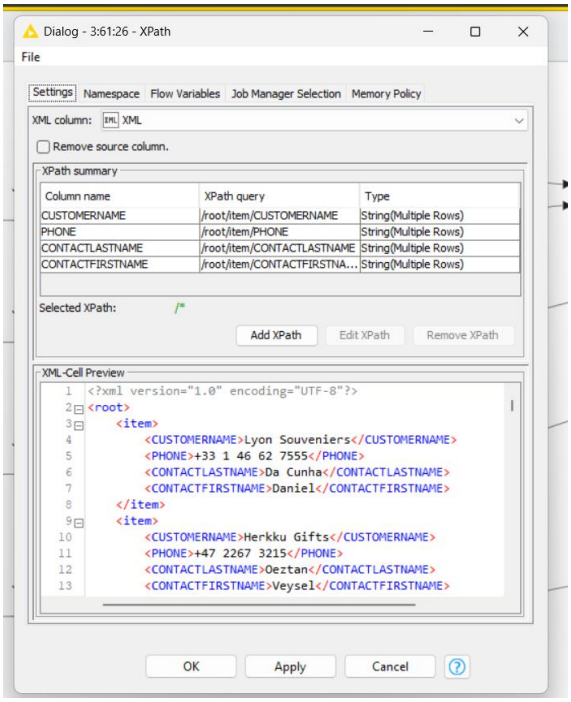

Figura 32 - XPath Customer

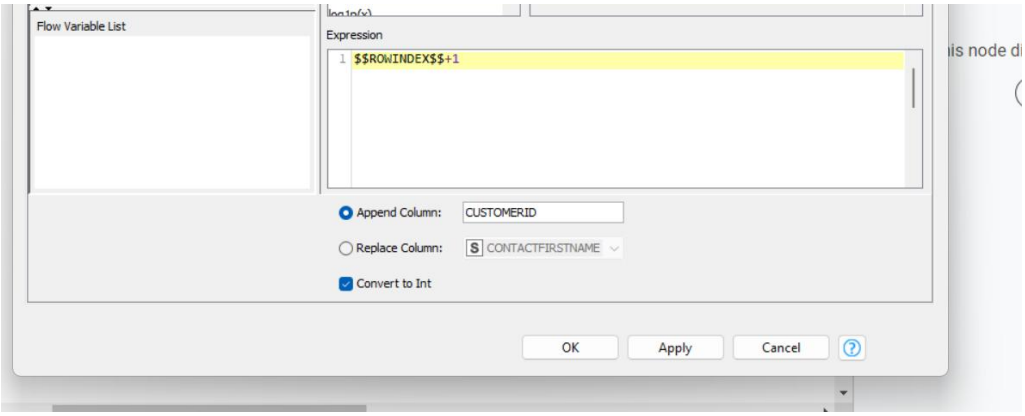

Figura 33 - Math Formula Criar ID

Neste caso só foi necessário criar os id's para a morada e para cliente pois o produto já tinha um identificador.

Com tudo devidamente validado, separado e identificado, utilizando o nó "Joiner" juntei todas as tabelas á tabela compra ainda tem os dados completos, para isso, na configuração do Joiner tirei as colunas do Cliente e adicionei o ID da outra tabela onde o nome do cliente na tabela compra for igual ao nome da tabela cliente, ficando assim com os dados do cliente numa tabela e apenas o id na tabela compra. Para as outras tabelas foi feito a mesma coisa terminando assim com a tabela compra apenas com o id do cliente, morada e produto, e os dados dos mesmo em outras tabelas separadas.

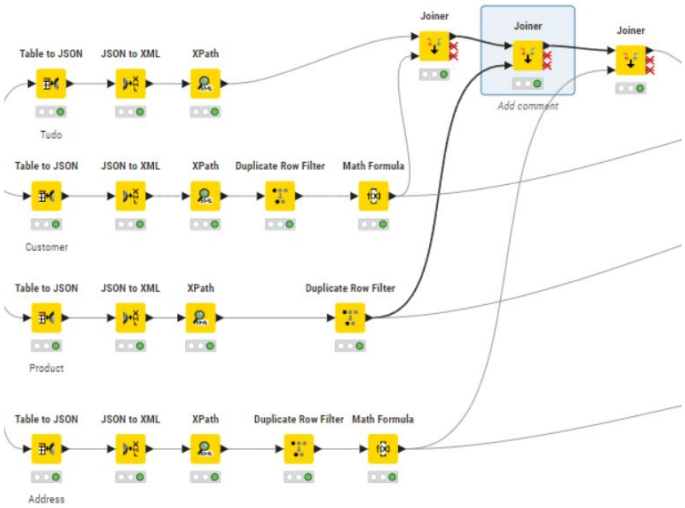

Figura 34 - Estrutura da Junçãodas tabelas

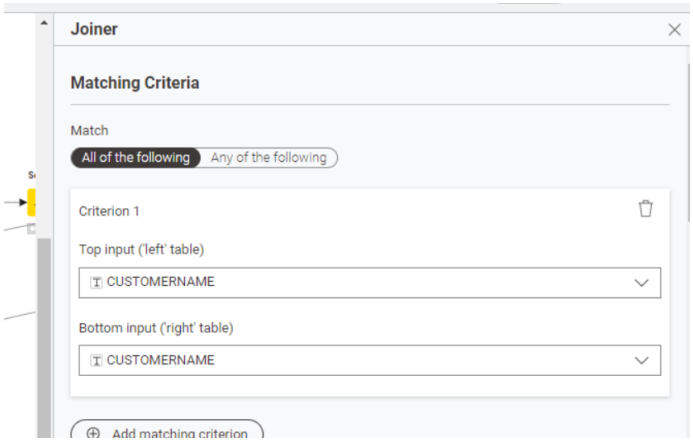

Figura 35 - Configuração do Joiner 1

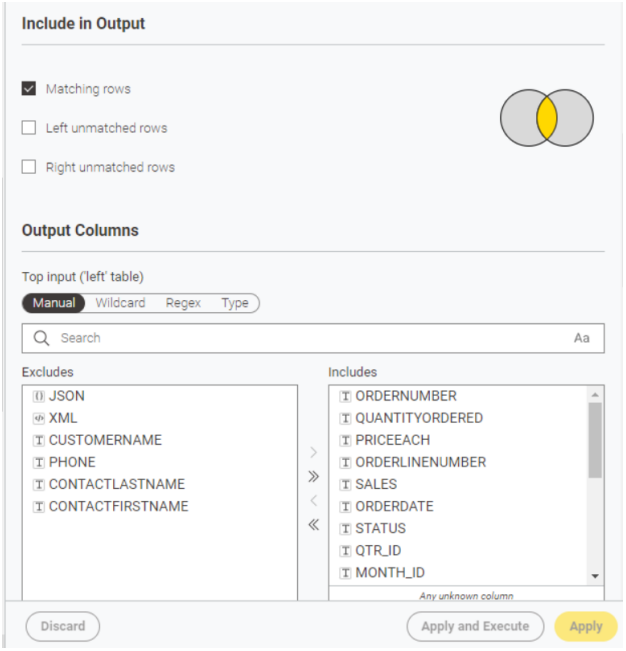

Figura 36 - Configuração do Joiner 2

Em seguida eliminei, caso exista, dados duplicados na tabela compra, desordenei os dados da tabela através do nó “Shuffle” e ordenei pelo identificador da tabela compra.

Com tudo normalizado, criei o próximo job ”Base de dados”

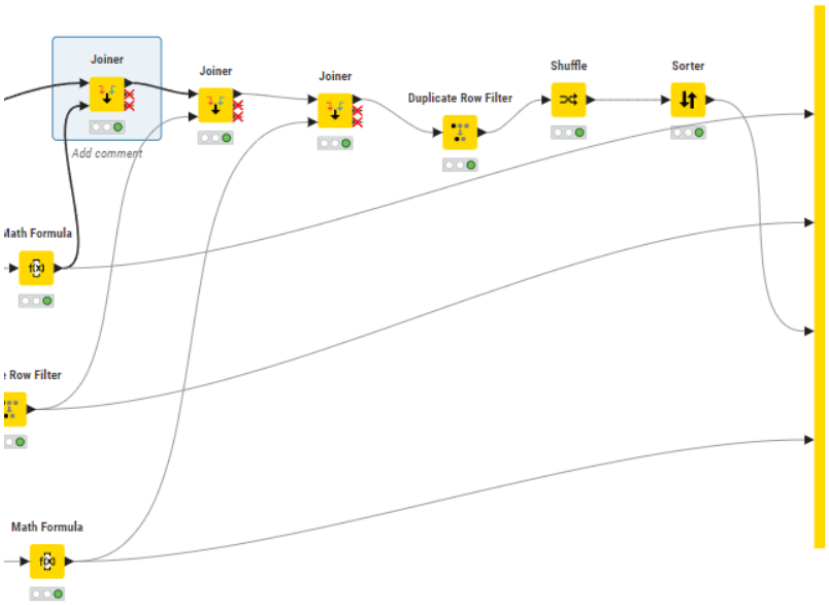

Figura 37 - Tratar tabela compra

3.2. Base de Dados

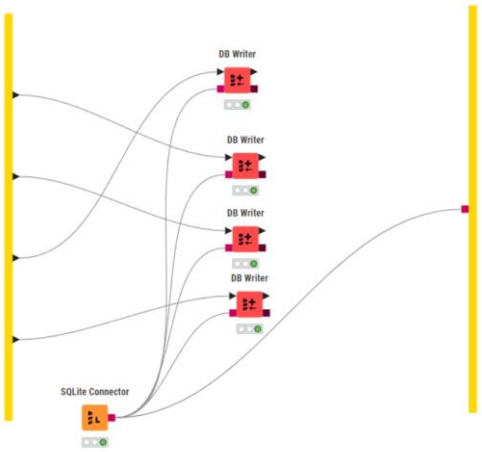

Figura 38 - Estrutura Base de dados

Com os dados preparados, introduzi nem uma base de dados. Primeiro criei a base de dados com o nó “SQLite Connector”, apenas foi necessário colocar o caminho onde vai gerar o ficheiro .sqlite

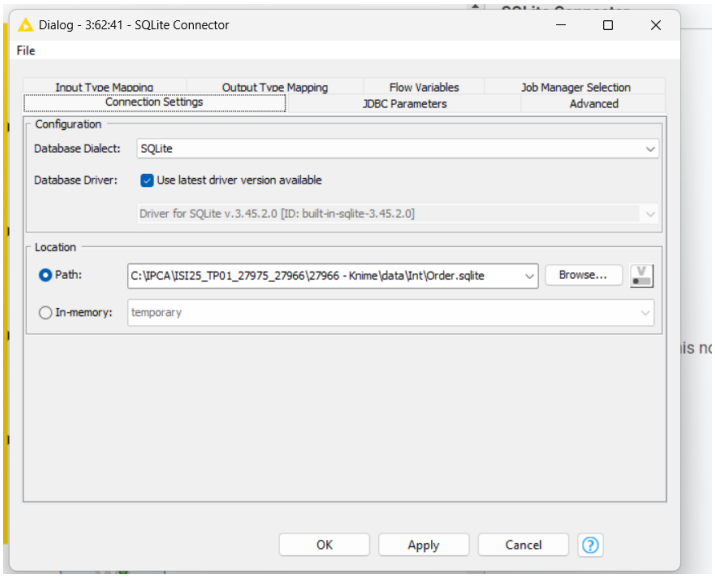

Figura 39 – SQLite

Em seguida, com a base de dados criada, inseri os dados com o nó “DB Writer” um para cada tabela, é necessário colocar o nome da coluna na base de dados e as respetivas colunas que vêm do job anterior

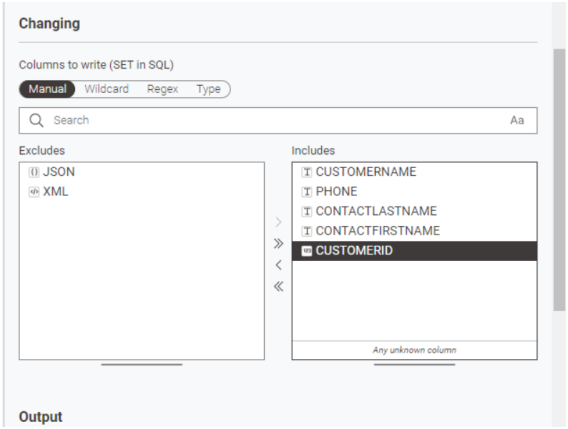

Figura 40 - Inserir na base de dados

Com a base de dados criada posso tratar os dados de forma mais simples com querys e com isso, criei o job Email e Graficos, onde vou ler os dados da base de dados e fazer ações.

3.3. Email

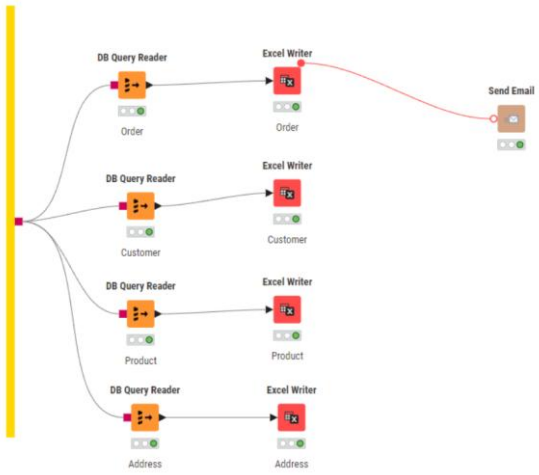

Figura 41 - Estrutura Base de Dados

Neste job utilizo os dados da base de dados para transformar em um ficheiro excel, através do "DB Query Reader"

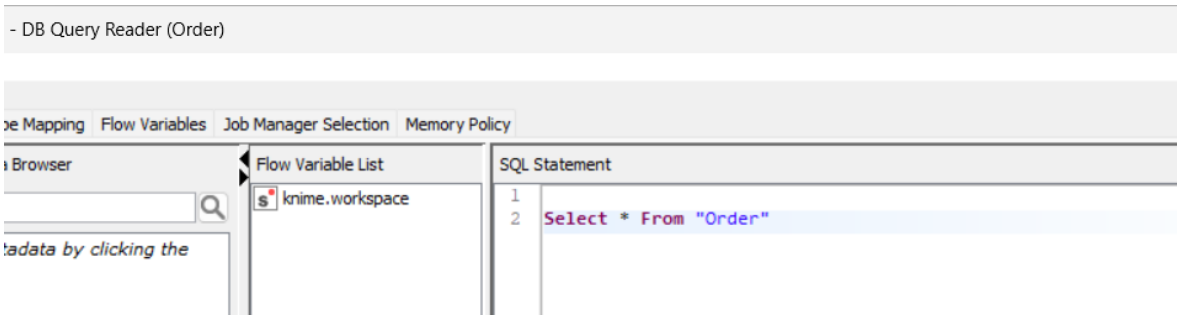

Figura 42 – Query

No nó "Excel Writer" apenas é necessário colocar o caminho onde vai ser gerado o ficheiro e marcar a opção "overwrite", caso o ficheiro já exista, para subscrever:

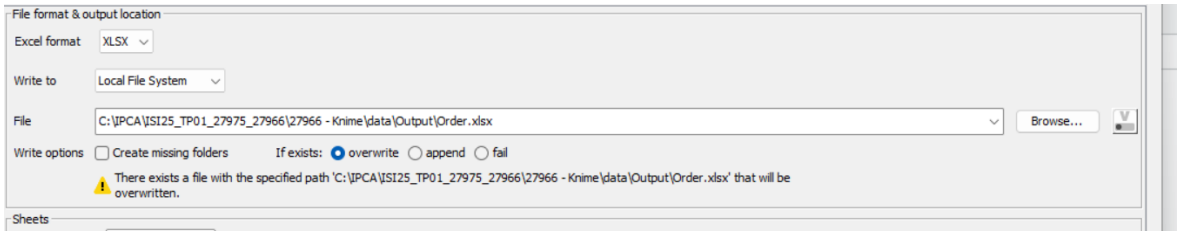

Figura 43 - Configuração excel

Depois de transformar em excel, envia por email os 4 ficheiros, na configuração preenchi os ficheiros que são para enviar, o meu email, criei uma palavra-passe de aplicação na gestão de contas no google e escrevi o email.

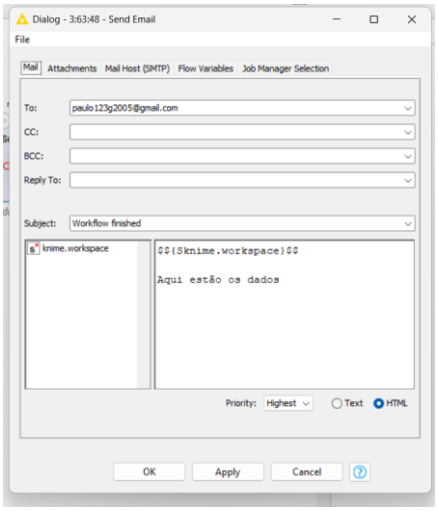

Figura 44 - Configuração email 1

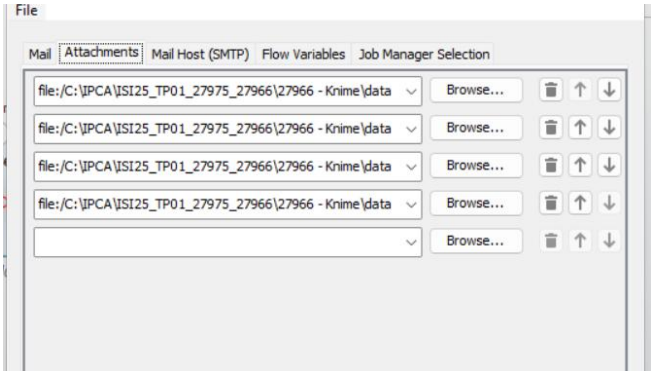

Figura 45 - Configuração email 2

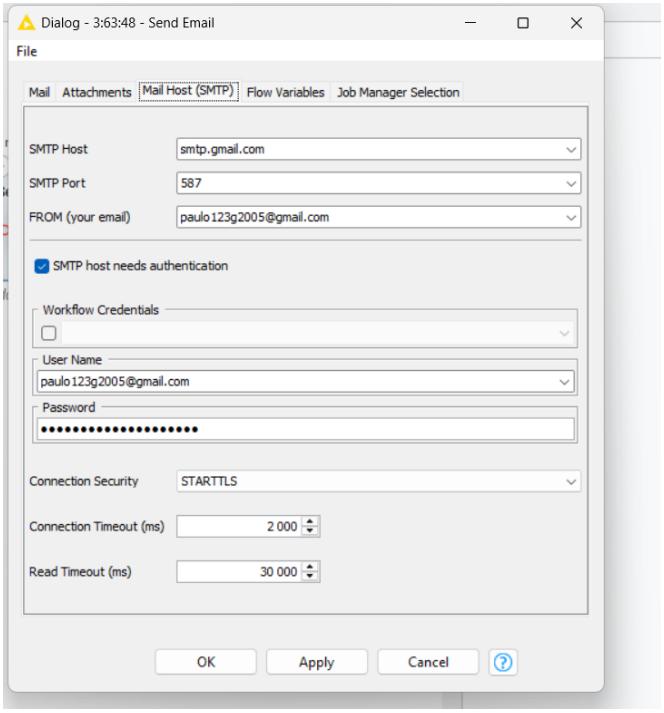

Figura 46 - Configuração email 3

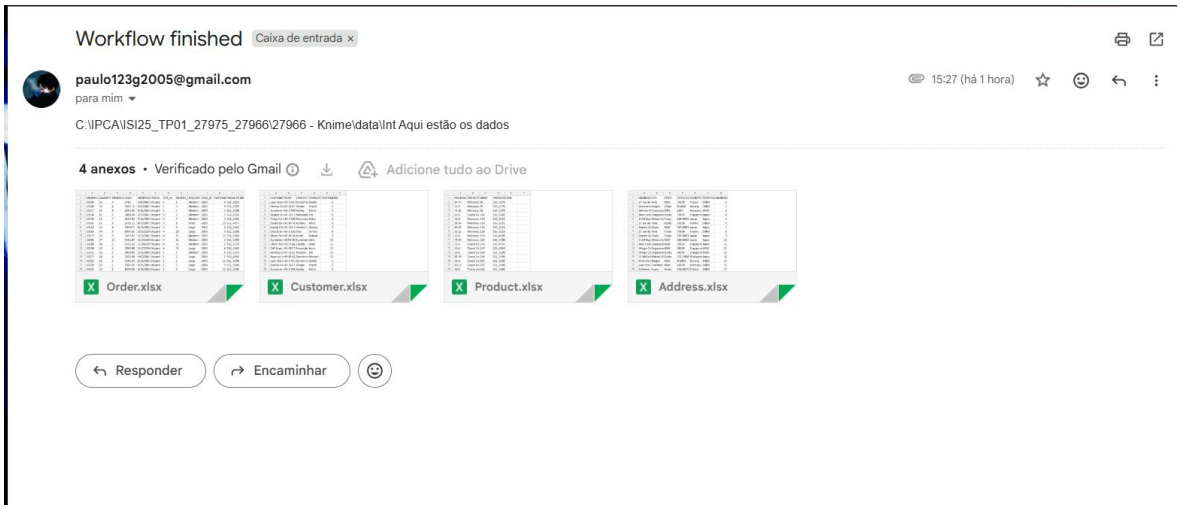

Figura 47 – Email

3.4. Gráficos

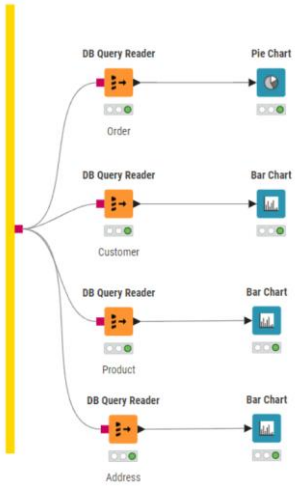

Figura 48 - Estrutura Graficos

Neste job utilizei o nó "DB Query Reader" como no job anterior para ler os dados da base de dados, e analisei os dados através dos nós "Pie Chart" e "Bar Chart"

Percentagens de encomendas em processo:

Figura 49 – Encomendas

Quantas pessoas utilizam cada nome

Figura 50 – Nomes

Linha do produto

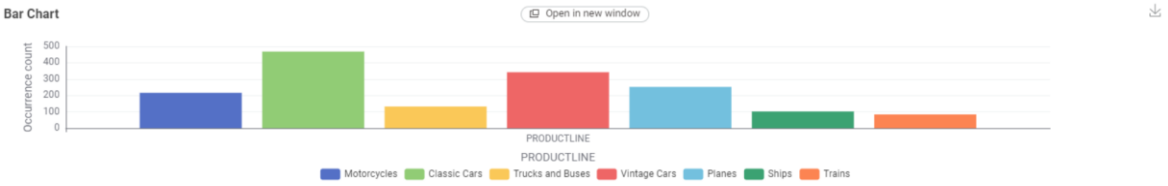

Figura 51 - Linha do produto

Clientes por estado

Figura 52 - Clientes por estado

3.5. API

Criei uma api com nodejs que, vai á base de dados sqlite “Order.sqlite”, estabelece uma conexão e cria um gráfico para mostrar o estado das encomendas e um para o tamanho das encomendas

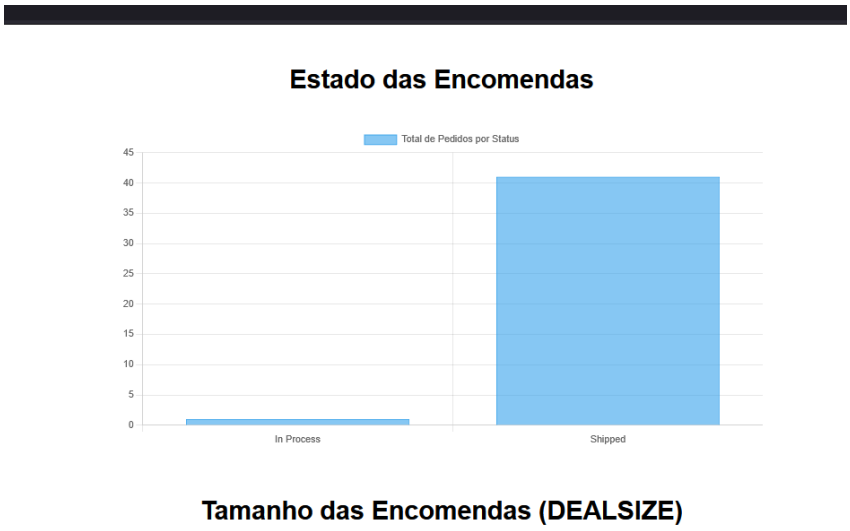

Figura 53 - Api estado

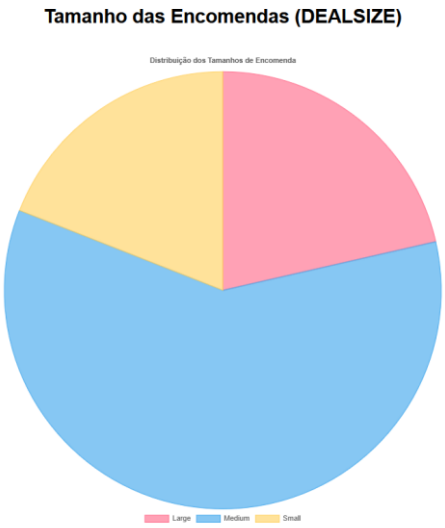

Figura 54 - Api tamanho

4. Vídeo com Demostração

4.1. Pentaho Kettle – Diogo Abreu

QR Code para o Vídeo

Link do Vídeo: Apresentação Pentaho.mp4

4.2. Knime – Paulo Gonçalves

QR Code para o Vídeo:

Link do Vídeo: Apresentação Knime.mkv

5. Conclusão

O presente projeto alcançou com sucesso o seu objetivo central: demonstrar um fluxo ETL completo e robusto, capaz de extrair dados de clientes de um formato tabular (CSV) e transformá-los em múltiplos ficheiros hierárquicos (XML).

A utilização de ferramentas de integração de dados, como o Pentaho Data Integration e Knime, permitiu concretizar as etapas cruciais de Data Quality:

- **Transformação e Divisão de Dados:** Os dados foram extraídos do ficheiro de origem e divididos em fluxos paralelos para criar "tabelas" dimensionais (Produto, Cliente, Data, Território) e uma tabela de factos (FactSales), todas exportadas para ficheiros XML.
- **Validação de Dados:** Foi implementada a validação de campos usando expressões regulares. Em particular, o campo PHONE foi validado para garantir que inicia com dígitos, e o campo ORDERDATE teve o seu formato validado para MM/DD/YYYY.
- **Preparação e Ordenação:** Demonstrou-se a capacidade de reformatar dados de data e de ordenar os registos nas novas tabelas XML criadas.

O sucesso da execução do Job de orquestração confirma a integração e o encadeamento lógico de todas as transformações. Este permitiu uma exploração aprofundada das capacidades de transformação do Pentaho e Knime. O projeto reforça a importância das práticas de ETL e Data Quality para a integração de dados em Sistemas de Informação.

6. Bibliografia

Documentação Oficial:

- Pentaho Documentation. (2025). Pentaho Data Integration User Guide

- Hitachi Vantara. (2025). Kettle Transformation Steps

Dataset: https://www.kaggle.com/datasets/kyanyoga/sample-sales-data

Ferramentas Utilizadas:

- Pentaho Data Integration 9.4

-KNIME Analytics Platform

- Git para controlo de versão