

Relatório do trabalho da disciplina de ISI

SALES

Susana Gomes – 18515

Engenharia de Sistemas Informáticos Pós-laboral

Outubro de 2024.

Índice

INTRODUÇÃO	9
OBJETIVOS DO PROJETO	10
PROGRAMAS UTILIZADOS NO PROJETO	12
INTRODUÇÃO GERAL AO DESENVOLVIMENTO DO PROJETO	13
CUSTOMER TRANSFORMATION	14
1. Extração dos Dados dos Clientes	15
2. Agregação dos Dados com Sorted Merge	18
3. Remoção de Duplicados com Unique Rows	18
4. Normalização dos Valores do Campo Country com Value Mapper	19
5. Ajustes no Campo City com Replace String	20
6. Fuzzy Matching para Corresponder ao Estado (State)	21
7. Conversão da Idade para Inteiro com Age to Integer	23
8. Filtragem dos Registos com Filter Rows	27
9. Cálculo do Valor Absoluto com Calculator	29
10. Atualização dos Campos e Seleção com Select Values 3	29
11. Agregação Final com Sorted Merge 2	30
12. Atualização da Tabela de Clientes no Banco de Dados com Lookup/Update	31
13. Atualização dos Dados dos Clientes com Step Update	32
Considerações Finais sobre a Transformação dos Clientes	34
PRODUCT TRANSFORMATION	36
1: Get Data from XML	36
2: JSON Input	38
3: Sort Rows	39

4: Select Values	40
Step 5: Sort Rows	41
Step 6: Sorted Merge	42
Step 7: Unique Rows	43
Step 8: Add Constants	43
Step 9: Dimension Lookup/Update	44
Considerações Finais sobre a Transformação dos Produtos	46
SALES TRANSFORMATION	47
Step 1: Table Input	47
Step 2: S3 CSV Input	48
Step 3: Select Values	49
Step 4: Sort Rows (Table Input)	50
Step 5: Sort Rows (S3 CSV Input)	51
Step 6: Append Streams	52
Step 7: Unique Rows	53
Step 8: Formula	53
Step 9: CSV File Input (Product Lookup)	54
Step 10: Stream Lookup	55
Step 11: Filter Rows	55
Step 12: Write to Log (False Condition)	56
Step 13: Select Values 3 (Remove Product Name)	57
Step 14: Calculator	58
Step 15: Filter Rows 2	59
Step 16: Microsoft Excel Output (Sales_Dates_Incorrect)	59
Step 17: Select Values 4	61
Step 18: Memory Group By	62
Considerações Finais sobre a Transformação das Vendas	63
JOB 65	

Step 1: Start	65
Step 2: Set Variables	66
Step 3: Customer Transformation	66
Step 4: Product Transformation	66
Step 5: Sales Transformation	67
Step 6: Check if Files Exist	67
Step 7: Abort Job (Caso o ficheiro não exista)	67
Step 8: Zip File (Caso o ficheiro exista)	67
Considerações finais	71
ESTRUTURA DAS BASES DE DADOS UTILIZADAS	72
1. Base de Dados AWS	72
2. Base de Dados SQL	72
VÍDEO DEMOSTRATIVO	74
CONCLUSÃO	75

Lista de Figuras

Figura 1: ESQUEMA PARA DESENVOLVIMENTO DO PROJETO SALES.	13
Figura 2: DIAGRAMA CUSTOMER TRANSFORMATION.	14
Figura 3: MANUAL INPUT.	15
Figura 4: TEXT FILE INPUT.	15
Figura 5: INPUT MULTIPLE FILES.	16
Figura 6: INPUTING EAST DATA FORM EXCEL.	16
Figura 7: INPUT FROM ZIP FILE.	17
Figura 8: SORT ROWS.	17
Figura 9: SORTED MERGE.	18
Figura 10: UNIQUE ROWS.	19
Figura 11: VALUE MAPPER.	20
Figura 12: REPLACE IN STRING.	20
Figura 13: FUZZY MATCH.	22
Figura 14: SELECT VALUES.	23
Figura 15: AGE TO INTEGER.	24
Figura 16: REPLACE IN STRING.	25
Figura 17: SELECT VALUES :META-DATA.	26
Figura 18: SELECT VALUES: SELECT&ALTER.	27
Figura 19: FILTER ROWS.	28
Figura 20: SORTED MERGE 2.	28
Figura 21: CALCULATOR.	29
Figura 22: SELECT VALUES 3: SELECT&ALTER.	30
Figura 23: SORTED MERGE2.	31
Figura 24: COMBINATION LOOKUP/UPDATE.	32
Figura 25: UPDATE SALESDBCONNECTION - CUSTOMER TABLE.	33
Figura 26: CUSTOMER TABLE IN POSTGRES.	35

Figura 27: PREVIEW DO UPDATE NO PDI.	35
Figura 28: PRODUCT TRANSFORMATION.	36
Figura 29: CONFIGURAÇÃO DA EXTRAÇÃO DE DADOS DO AQRUIVO XML.	36
Figura 30: GET DATA FROM XML: FILE.	37
Figura 31: GET DATA FROM XML: CONTENT.	37
Figura 32: GET DATA FROM XML: FIELDS.	38
Figura 33: CONFIGURAÇÃO DOS DADOS EXTRAÍDOS O FICHEIRO JSON.	38
Figura 34: JSON INPUT:FILE.	39
Figura 35: JSON INPUT: FIELDS.	39
Figura 36: SORT ROWS.	40
Figura 37: CAMPOS SELECT VALUES.	40
Figura 38: SELECT VALUES: SELECT&ALTER.	41
Figura 39: SORT ROWS.	42
Figura 40: SORTED MERGE.	42
Figura 41: UNIQUE ROWS.	43
Figura 42: ADD CONSTANTS.	44
Figura 43: DIMENSION LOOKUP/UPDATE: FIELDS.	44
Figura 44: DIMENSION LOOKUP/UPDATE: KEYS.	45
Figura 45: SALES TRANSFORMATION.	47
Figura 46: TABLE INPUT.	48
Figura 47: S3 CSV INPUT.	49
Figura 48: SELECT VALUES.	50
Figura 49: SORT ROWS (TABLE INPUT).	51
Figura 50: SORT ROWS (S3 CSV INPUT).	52
Figura 51: APPEND STREAMS.	52
Figura 52: UNIQUE ROWS.	53
Figura 53: FÓRMULA.	54
Figura 54: CSV FILE INPUT.	54

Figura 55: STREAM LOOKUP.	55
Figura 56: FILTER ROWS.	56
Figura 57: WRITE TO LOG.	57
Figura 58: SELECT VALUES 3.	58
Figura 59: CALCULATOR.	58
Figura 60: FILTER ROWS 2.	59
Figura 61: MICROSOFT EXCEL OUTPU: FILE.	60
Figura 62: MICROSOFT EXCEL OUTPUT: FIELDS.	61
Figura 63: SELECT VALUES 4.	62
Figura 64: MEMORY GROUP BY.	63
Figura 65: JOB.	65
Figura 66: SET VARIABLES.	66
Figura 67: ABORT JOB.	67
Figura 68: ZIP FILE.	68
Figura 69: MAIL: ADDRESSES.	69
Figura 70: MAIL:SERVER.	69
Figura 71: MAIL: EMAIL MESSAGE.	70
Figura 72: MAIL: ATTACHED FILES.	70
Figura 73: EMAIL RECEBIDO COM O ZIP FILE.	70
Figura 74: BICKET AWS.	72
Figura 75: TABELAS BASE DE DADOS POSTGRES SQL.	73
Figura 76: TABELA CUSTOMER.	73
Figura 77: TABELA PRODUCT.	73
Figura 78: TABELA SALES.	74

INTRODUÇÃO

No contexto atual de crescimento constante da competitividade empresarial, a gestão eficiente dos dados é um fator crucial para a tomada de decisões e otimização de processos. Este trabalho tem como foco a gestão dos dados de vendas de uma empresa, utilizando uma abordagem baseada em ferramentas de Business Intelligence (BI). O objetivo é proporcionar uma visão clara e concisa do desempenho da empresa, facilitando a compreensão dos padrões de consumo, a análise de vendas e o entendimento do comportamento dos clientes.

Para atingir este objetivo, foi utilizada a plataforma PENTAHO, um conjunto de ferramentas poderosas para o tratamento e integração de dados. No âmbito deste projeto, foram desenvolvidas três transformações principais: a primeira voltada para o tratamento dos dados dos clientes (Customers), a segunda para os dados dos produtos (Products), e a terceira para os dados de vendas (Sales). Cada transformação foi desenhada com o intuito de realizar a limpeza, normalização e integração dos dados, de forma a garantir a sua consistência e qualidade.

Além do uso do PENTAHO para a criação e gestão dessas transformações, este trabalho também integrou dados armazenados em diferentes fontes, incluindo a AWS (Amazon Web Services) e um banco de dados PostgreSQL. A combinação destas tecnologias permitiu a construção de um sistema robusto para o tratamento dos dados, garantindo a disponibilidade e a precisão necessárias para a análise detalhada dos indicadores de vendas.

A importância deste projeto reside na capacidade de consolidar informações dispersas em diferentes sistemas e fontes de dados, proporcionando à empresa uma base de dados coesa e ágil para consultas analíticas. Dessa forma, a empresa poderá melhorar o planeamento de estratégias de vendas, compreender melhor o perfil dos seus clientes e tomar decisões fundamentadas em dados concretos e confiáveis.

Objetivos do Projeto

O principal objetivo deste projeto é desenvolver uma solução eficiente para a gestão dos dados de vendas de uma empresa, utilizando a plataforma PENTAHO para processar, transformar e integrar dados provenientes de diferentes fontes. Esta solução visa consolidar informações dispersas, garantindo maior precisão, consistência e agilidade no acesso aos dados relevantes para a tomada de decisão.

Especificamente, os objetivos deste projeto incluem:

Extrair dados de múltiplas fontes: Realizar a extração dos dados provenientes de diversas fontes, incluindo arquivos Excel, CSV, XML, JSON, bases de dados SQL e sistemas de armazenamento na nuvem (AWS), consolidando-os no PENTAHO para tratamento posterior.

Integrar e agregar dados dos clientes, produtos e vendas: Agregar dados de diferentes origens em três fluxos distintos: Sales Data, Customer Data e Product Data. Cada fluxo foi tratado de maneira independente para garantir que todas as informações relevantes fossem consolidadas e disponibilizadas para análise.

Tratar e limpar os dados: Utilizar as transformações criadas no PENTAHO para realizar a limpeza e padronização dos dados dos clientes, produtos e vendas, assegurando a qualidade dos dados e reduzindo erros. Esse processo envolve o tratamento de colunas específicas, como IDs de clientes e produtos, além de informações adicionais, como datas de pedidos e detalhes financeiros.

Transformar, agregar e unificar dados para criar um Data Mart: Realizar transformações, agregações e uniões dos dados extraídos para facilitar análises. Essa etapa culmina no

carregamento dos dados tratados para um Data Mart, que possibilitará consultas rápidas e eficientes, suportando análises detalhadas e relatórios empresariais.

Facilitar a análise de dados e visualização: Proporcionar uma estrutura de dados organizada e otimizada que permita uma análise eficiente dos padrões de vendas, comportamentos dos clientes e performance dos produtos, contribuindo para insights relevantes para a empresa. A criação de relatórios e dashboards permitirá visualizar os dados de maneira intuitiva e em tempo hábil.

Automatizar processos de ETL (Extract, Transform, Load): Criar um processo automatizado para a extração, transformação e carga dos dados, garantindo que as informações estejam sempre atualizadas e disponíveis para consultas e relatórios, proporcionando maior eficiência ao fluxo de dados.

Suportar a tomada de decisões estratégicas: Disponibilizar relatórios e dashboards que forneçam informações relevantes de forma clara e precisa, auxiliando os gestores na formulação de estratégias e tomada de decisões baseadas em dados concretos e históricos de vendas.

Com esses objetivos, espera-se que a empresa possa ter um maior controle sobre suas operações de vendas, compreendendo melhor o comportamento dos clientes e otimizando suas estratégias de mercado com base em dados concretos e precisos.

Programas utilizados no projeto

- Pentaho Data Integracion;
- PgAdmin 4;
- AWS (Amazon Web Services).

Introdução geral ao desenvolvimento do projeto

Antes de iniciar o desenvolvimento do projeto, senti a necessidade de criar um esquema para compreender claramente o que precisaria ser desenvolvido ao longo do projeto. O objetivo era definir o que seria feito em cada uma das transformações, garantindo que a informação fosse tratada exatamente como planeado. Abaixo, apresento o esquema que criei.

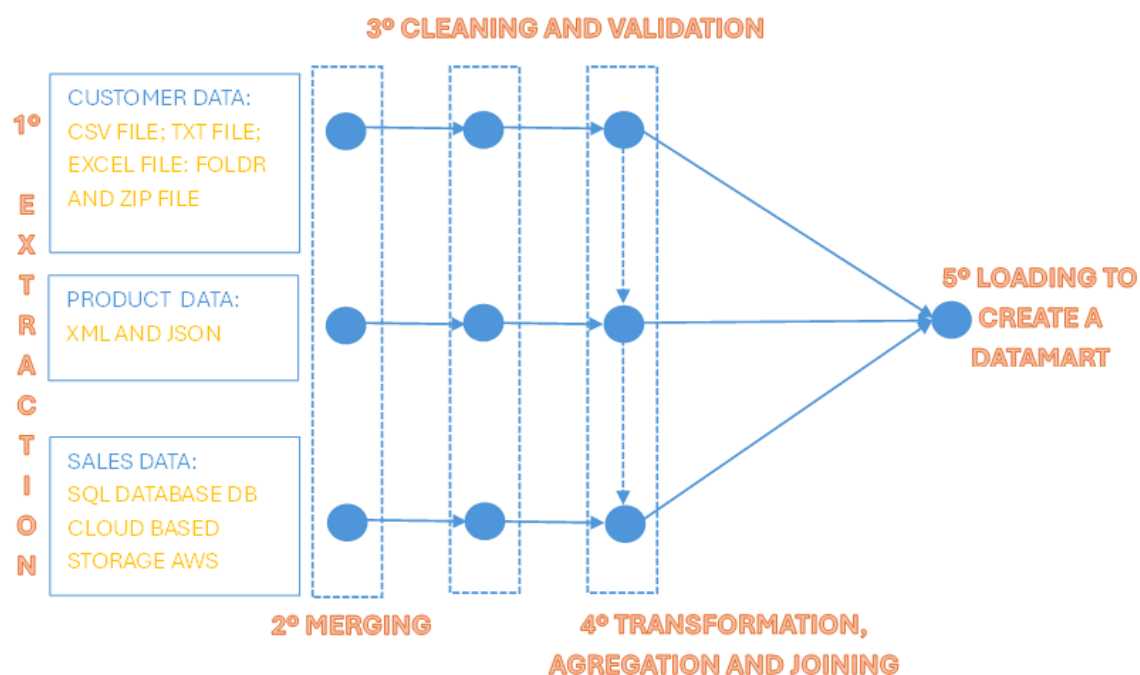


Figura 1: ESQUEMA PARA DESENVOLVIMENTO DO PROJETO SALES.

Customer Transformation

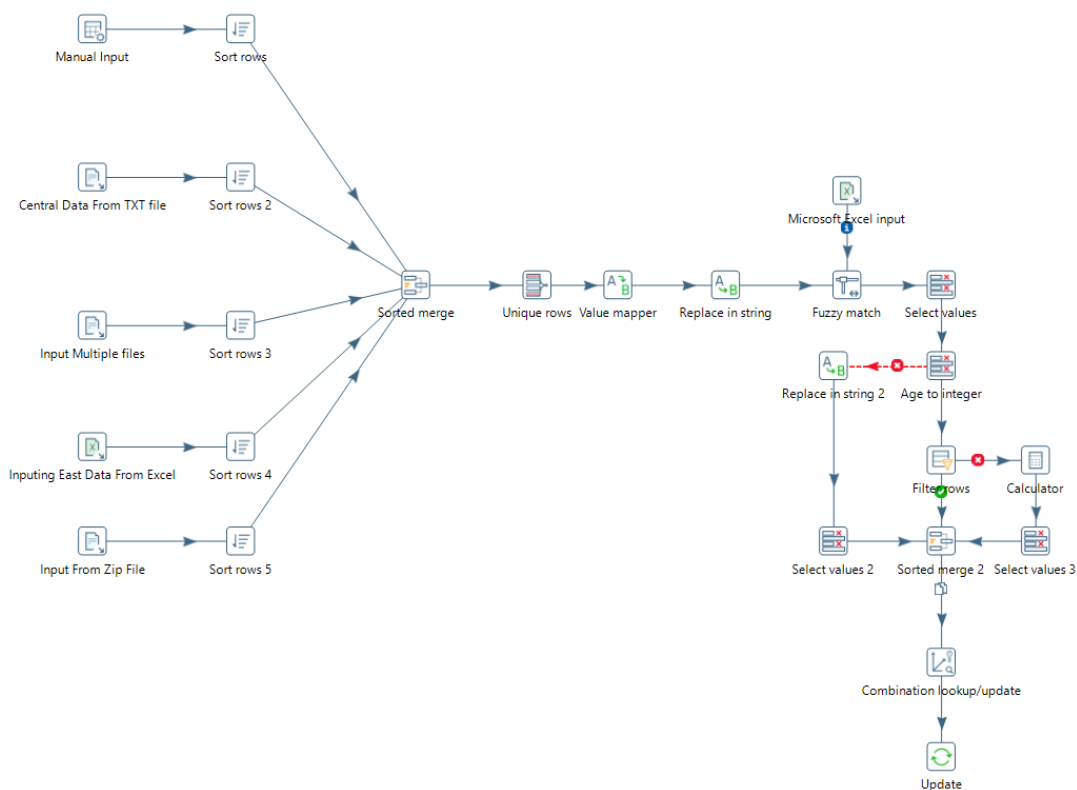


Figura 2: DIAGRAMA CUSTOMER TRANSFORMATION.

Nesta seção, iremos descrever detalhadamente a transformação dos dados dos clientes realizada através do Pentaho Data Integration (PDI). O objetivo desta transformação foi garantir a qualidade, consistência e integração dos dados dos clientes para possibilitar análises precisas e fundamentadas, fornecendo informações cruciais para a empresa. Cada etapa da transformação foi cuidadosamente elaborada para assegurar que os dados estivessem preparados para uso, sem inconsistências ou valores incorretos. Abaixo, encontra-se uma descrição detalhada de cada etapa, acompanhada de imagens de configuração sempre que relevante.

1. Extração dos Dados dos Clientes

A transformação iniciou-se com a extração dos dados de múltiplas fontes, como ficheiros Excel, CSV e bases de dados SQL. Cada um destes ficheiros foi carregado no PDI e foi realizado um passo de Sorted Rows para ordenar os dados com base no campo Customer ID. Esta ordenação foi essencial para que os dados pudessem ser corretamente mesclados na etapa seguinte.

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Null if	Set empty string?
1	Customer ID	String		10						N
2	Customer Name	String		50						N
3	Segment	String		20						N
4	Age	String		10						N
5	Country	String		25						N
6	City	String		50						N
7	State	String		50						N
8	Postal Code	String		10						N
9	Region	String		10						N

Figura 3: MANUAL INPUT.

#	File/Directory	Wildcard (RegExp)	Exclude wildcard	Required	Include subfolders
1	\$ROOT_PATH/Customer Data/CustomerData_Central.txt			N	N

Figura 4: TEXT FILE INPUT.

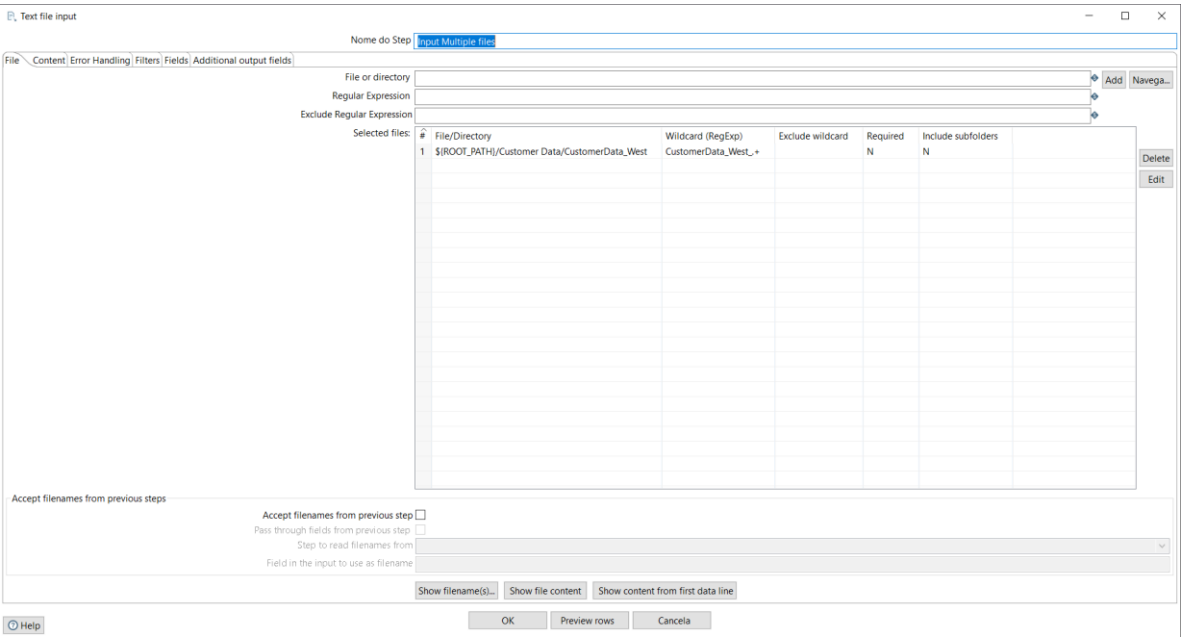


Figura 5: INPUT MULTIPLE FILES.

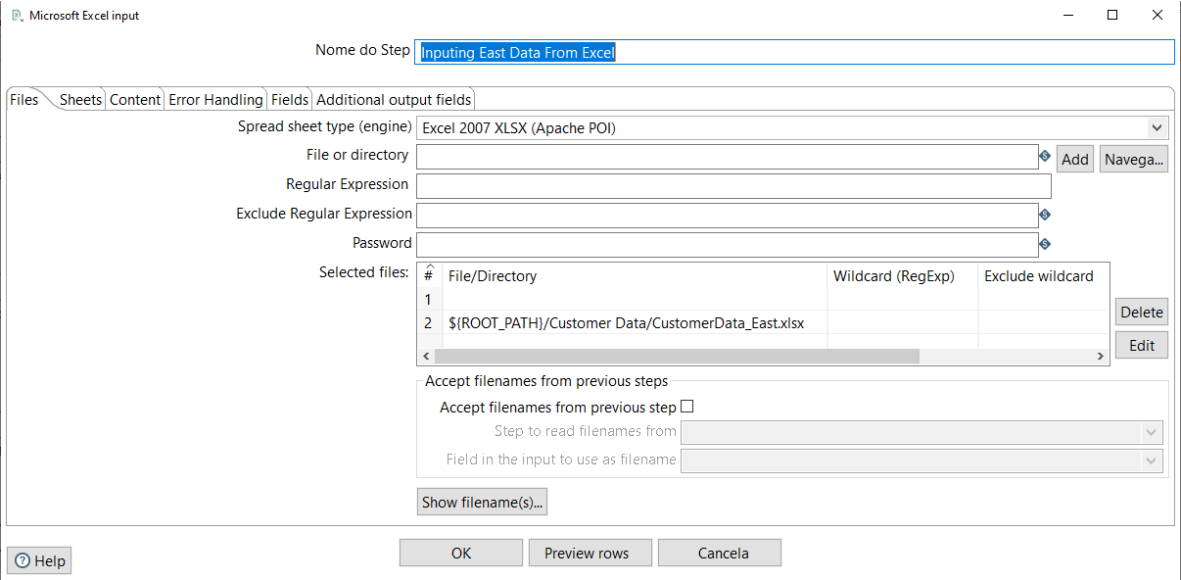


Figura 6: INPUTING EAST DATA FORM EXCEL.

Text file input

Nome do Step: **Input From Zip File**

File or directory:

Regular Expression:

Exclude Regular Expression:

Selected files:

#	File/Directory	Wildcard (RegExp)	Exclude wildcard	Required	Include subfolders
1	\$(ROOT_PATH)/Customer Data/CustomerData_South.zip			N	N

Accept filenames from previous steps: ☐

Accept filenames from previous step: ☐

Pass through fields from previous step: ☐

Step to read filenames from:

Field in the input to use as filename:

Show filename(s)... Show file content Show content from first data line

Help OK Preview rows Cancela

Figura 7: INPUT FROM ZIP FILE.

O step Sorte Rows que cada um desses inputs liga tem as mesmas configurações.

Sort rows

Nome do Step: **Sort rows 5**

Sort directory: Navega...

TMP-file prefix:

Sort size (rows in memory):

Free memory threshold (in %):

Compress TMP Files? ☐

Only pass unique rows? (verifies keys only) ☒

Fields:

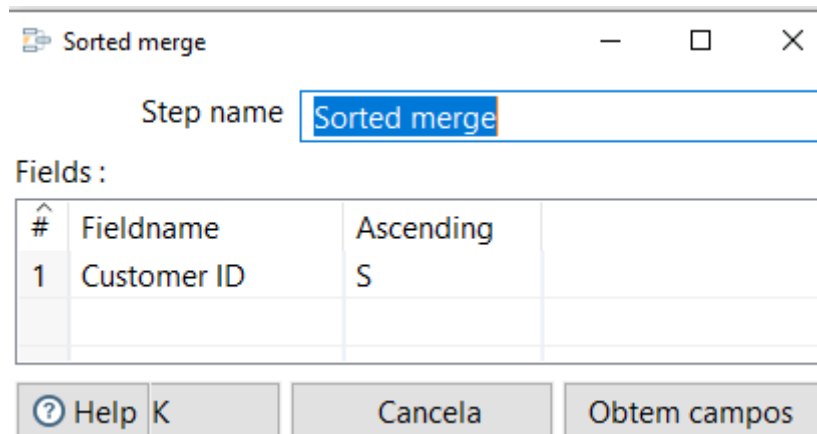
#	Fieldname	Ascending	Case sensitive compare?	Sort based on current locale?	Collator Strength	Presorted?
1	Customer ID	S	S	N	0	N

Help OK Cancela Obtem campos

Figura 8: SORT ROWS.

2. Agregação dos Dados com Sorted Merge

Após a ordenação dos dados de cada fonte, foi realizado um Sorted Merge utilizando o campo Customer ID para unir os diferentes fluxos de dados. O Sorted Merge garantiu que todas as informações dos clientes estivessem consolidadas em um único fluxo.



Sorted merge

Step name: Sorted merge

Fields :

#	Fieldname	Ascending
1	Customer ID	S

Buttons: ? Help K, Cancela, Obtem campos

Figura 9: SORTED MERGE.

3. Remoção de Duplicados com Unique Rows

Após a agregação dos dados, foi utilizada a etapa de Unique Rows para garantir que cada cliente tivesse um único registo, removendo entradas duplicadas com base no Customer ID.

Unique rows

Step name

Settings

Add counter to output? ☐ Counter field

Redirect duplicate row ☐ Error description

Fields to compare on (no entries means: compare complete row)

#	Fieldname	Ignore case
1	Customer ID	N

? Help OK Cancela Get

Figura 10: UNIQUE ROWS.

4. Normalização dos Valores do Campo Country com Value Mapper

Para uniformizar o campo Country, utilizámos o passo Value Mapper, que converteu os diferentes valores representando o mesmo país (ex.: USA, United States, United States of America) para um valor único (United States).

Value mapper

Step name: Value mapper

Fieldname to use: Country

Target field name (empty=overwrite):

Default upon non-matching:

Field values:

#	Source value	Target value
1	US	United States
2	USA	United States
3	United States of America	United States

Help OK Cancela

Figura 11: VALUE MAPPER.

5. Ajustes no Campo City com Replace String

Em seguida, foi utilizado o passo Replace String no campo City para padronizar e corrigir eventuais erros nos nomes das cidades. Esse passo foi utilizado para garantir que todas as cidades estivessem representadas de maneira consistente.

Replace in string

Step name: Replace in string

Fields string

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field	Whole Word	Case sensitive	Is Unicode
1	City		N	#		N		N	N	N

Help OK Get fields Cancela

Figura 12: REPLACE IN STRING.

6. Fuzzy Matching para Corresponder ao Estado (State)

Foi utilizado o passo Fuzzy Match para comparar os valores do campo State com uma lista de referência armazenada num ficheiro Excel chamado lookupforstates. Este ficheiro foi utilizado como lookup para garantir a consistência dos nomes dos estados. A configuração do Fuzzy Match envolveu o uso de um Microsoft Excel Input como lookup step, com o lookup field definido como state name e o main stream field como state.

O algoritmo escolhido foi o Levenshtein, que mede a similaridade entre strings ao calcular a distância de edição necessária para transformar uma palavra noutra. Foi definido um valor máximo de diferença de 2 (maximal value = 2) para garantir que apenas pequenas variações fossem aceites. A opção Get Closest Value foi seleccionada para encontrar o valor mais próximo.

Além disso, os campos foram configurados para que o resultado do match fosse armazenado no campo Match field e a medida de valor no campo Value field. Dessa forma, o campo state name foi padronizado de acordo com a lista de referência do ficheiro Excel, garantindo a consistência dos valores de estado na base de dados.

Fuzzy match — □ ×

Step name **Fuzzy match**

General Fields

Lookup stream (source)

Lookup step **Microsoft Excel input** ▼

Lookup field **State Name** ▼ ⚙

Main stream

Main stream field **State** ▼ ⚙

Settings

Algorithm **Levenshtein** ▼

Case sensitive ☐

Get closer value ☒

Minimal value **0** ⚙

Maximal value **2** ⚙

Values separator **,** ⚙

? Help OK Cancela

Figura 13: FUZZY MATCH.

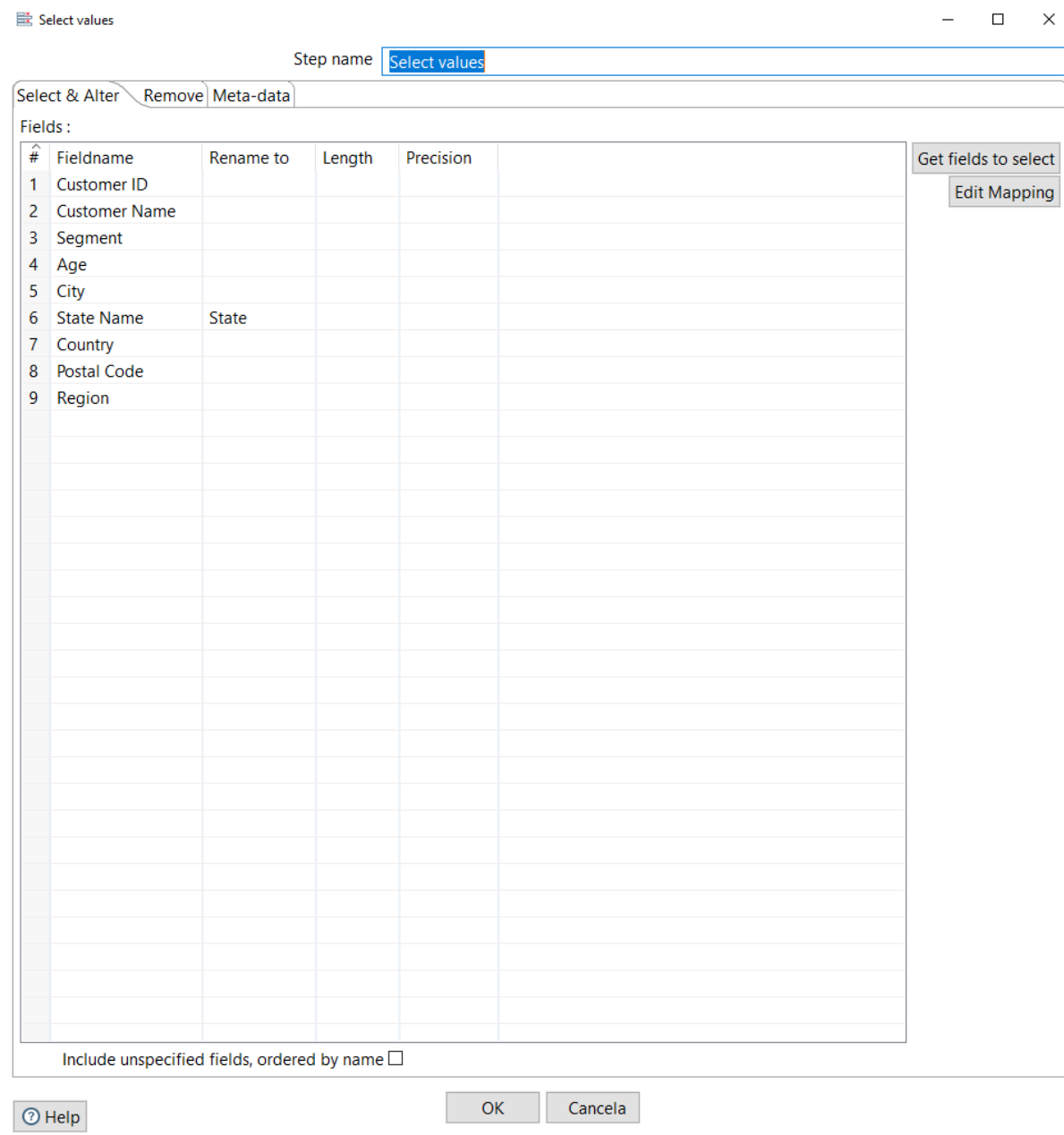


Figura 14: SELECT VALUES.

7. Conversão da Idade para Inteiro com Age to Integer

Após a realização do Fuzzy Matching para padronização dos estados, o passo Select Values foi utilizado para selecionar e renomear os campos relevantes, preparando o fluxo para a conversão da idade. Este passo foi seguido pelo Age to Integer, que

converteu o campo Age para o tipo Integer, garantindo que todos os valores de idade fossem numéricos e prontos para análises subsequentes.

Posteriormente, utilizou-se o passo Replace in String, onde foi configurado que, caso o campo Age contivesse o valor incorreto 'o', ele seria substituído por '0'. Esta etapa assegurou que todos os valores fossem válidos e consistentes para o restante do fluxo.

The screenshot shows a window titled "Select values". At the top right are standard window controls (minimize, maximize, close). Below the title bar is a text field labeled "Step name" containing the text "Age to integer". A horizontal tab bar contains three tabs: "Select & Alter", "Remove", and "Meta-data", with "Meta-data" being the active tab. Below the tabs is a label "Fields to alter the meta-data for :". Underneath is a large table with 8 columns: "#", "Fieldname", "Rename to", "Type", "Length", "Precision", "Binary to Normal?", and "Format". The first row of the table contains the following data: "#", "1", "Age", "", "Integer", "", "", "N", and "#". The rest of the table rows are empty. To the right of the table, there is a button labeled "Get fields to change...". At the bottom left is a "Help" button with a question mark icon. At the bottom center are two buttons: "OK" and "Cancela".

Figura 15: AGE TO INTEGER.

Replace in string

Step name: Replace in string 2

Fields string

#	In stream field	Out stream field	use RegEx	Search	Replace with	Set empty string?	Replace with field	Whole Word	Case sensitive	Is Unicode
1	Age		N	o	0	N		N	N	N

Help OK Get fields Cancela

Figura 16: REPLACE IN STRING.

O passo Select Values foi utilizado para seleccionar e renomear os campos relevantes, preparando o fluxo para a conversão da idade.

[illegible]

Figura 17: SELECT VALUES :META-DATA.

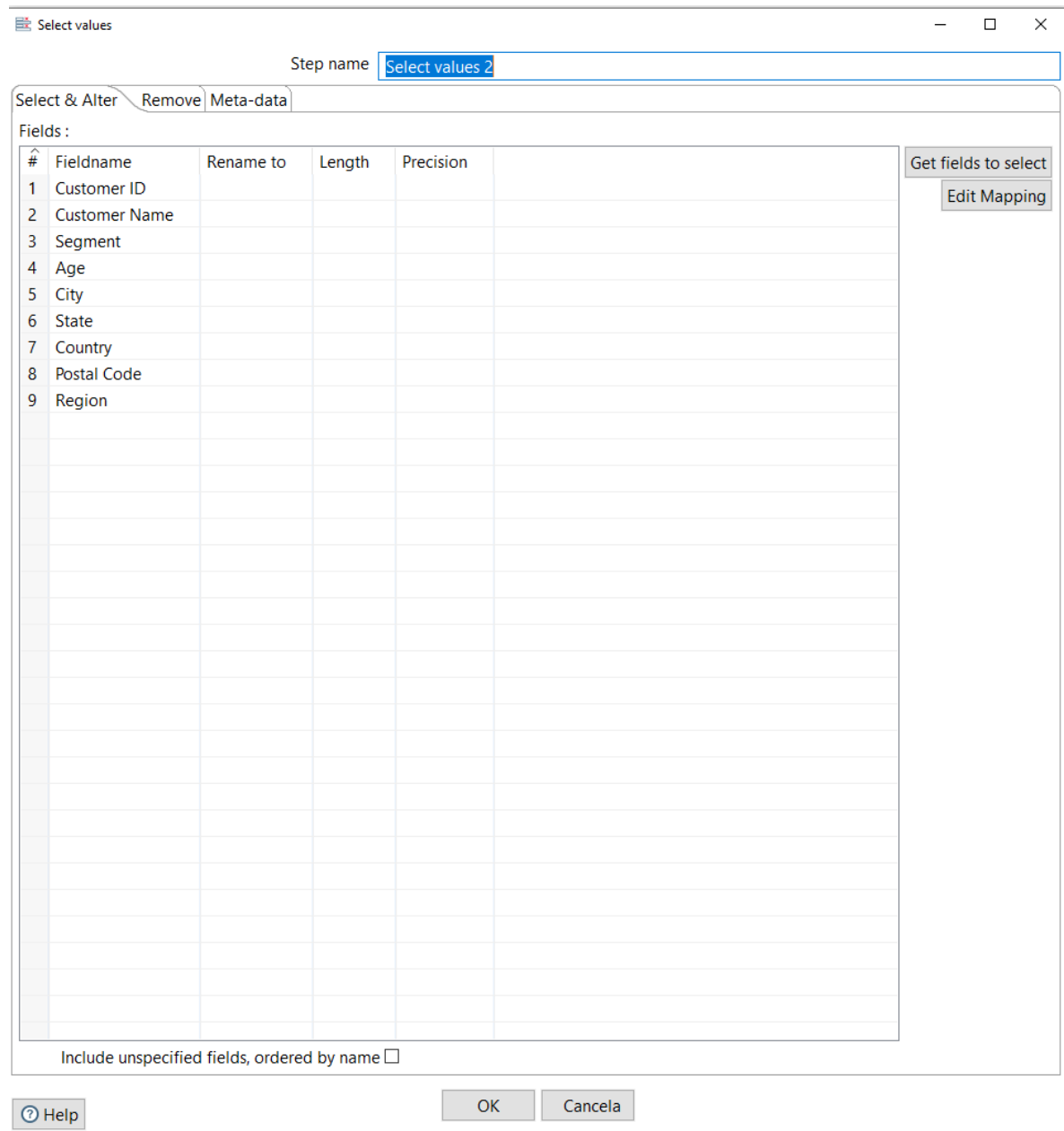


Figura 18: SELECT VALUES: SELECT&ALTER.

8. Filtragem dos Registos com Filter Rows

O passo Filter Rows foi utilizado para segmentar os registos de acordo com a condição do campo Age. A condição especificada foi Age > 0, de forma a garantir que apenas

registos com idades válidas (maiores que zero) fossem mantidos no fluxo principal de dados.

Os registos que atendiam a essa condição eram enviados para o passo Sorted Merge 2, enquanto os registos que não atendiam à condição eram encaminhados para o passo Calculator para posterior tratamento. Esta etapa foi fundamental para garantir que apenas dados válidos e consistentes fossem considerados nas etapas seguintes.

Filter rows

Step name:

Send 'true' data to step:

Send 'false' data to step:

The condition:

(Integer)

Figura 19: FILTER ROWS.

Sorted merge

Step name:

Fields :

#	Fieldname	Ascending
1	Customer ID	S

Figura 20: SORTED MERGE 2.

9. Cálculo do Valor Absoluto com Calculator

No passo Calculator, foi configurado o cálculo do valor absoluto para o campo Age, garantindo que não houvesse idades negativas. Este novo campo foi chamado Age1, e posteriormente renomeado para Age.

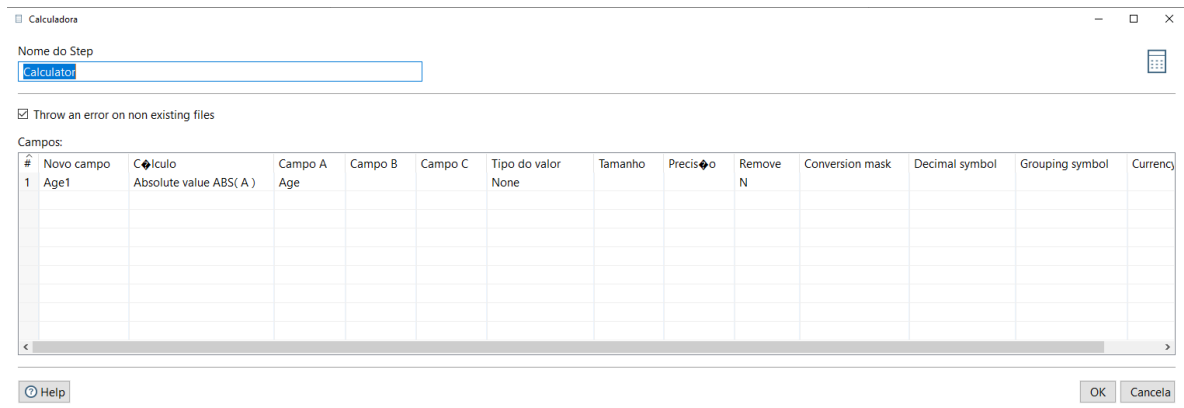


Figura 21: CALCULATOR.

10. Atualização dos Campos e Seleção com Select Values 3

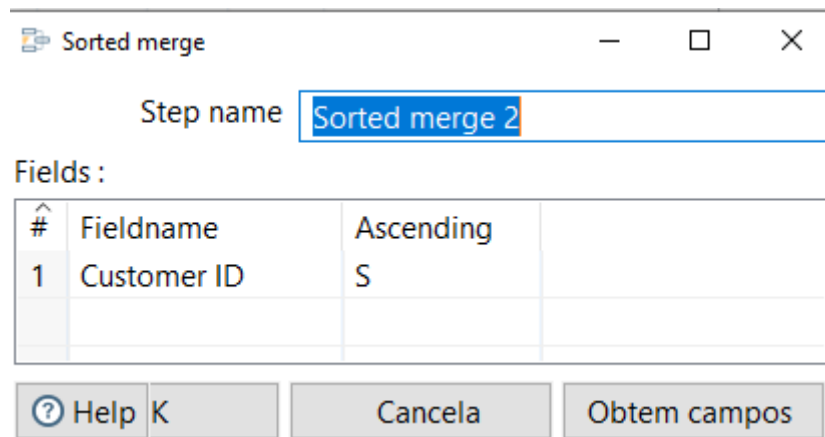
No passo Select Values 3, o campo Age1 foi renomeado para Age, e os outros campos foram selecionados para continuar no fluxo de dados. Isso garantiu a consistência dos nomes e a preparação dos dados para as próximas etapas.

[illegible]

Figura 22: SELECT VALUES 3: SELECT&ALTER.

11. Agregação Final com Sorted Merge 2

Os fluxos de dados resultantes de Select Values 2, Filter Rows, e Select Values 3 foram unidos com o passo Sorted Merge 2, utilizando o campo Customer ID em ordem ascendente.



Sorted merge

Step name

Fields :

#	Fieldname	Ascending
1	Customer ID	S

Figura 23: SORTED MERGE2.

12. Atualização da Tabela de Clientes no Banco de Dados com Lookup/Update

Foi então utilizado o passo Lookup/Update para verificar se os clientes já existiam na tabela customer da base de dados (salesDBconnection). Caso não existissem, era inserido um novo registo, gerando um SURR_ID com base na maior chave existente na tabela.

Combinação lookup / update

Nome do Step: Combination lookup/update

Connection: salesDBconnection

Target schema: public

Tabela de destino: customer

Confirma tamanho: 1

Tamanho do Cache: 9999

Pre-load the cache? ☐

Campos Chave (para verificar linha na tabela):

#	Campo Dimensão	Campo no fluxo	
1	customer_id	Customer ID	

Campo chave técnica: SURR_ID

Criação de chave técnica

☒ Usa tabela máxima + 1

☐ Usa sequência

☐ Usa o campo de auto incremento

Remove campos lookup? ☐

Usa hashcode? ☐

Campo Hashcode na tabela


Date of last update field (optional)

Help OK Cancela Obtem Campos SQL

Figura 24: COMBINATION LOOKUP/UPDATE.

13. Atualização dos Dados dos Clientes com Step Update

Por fim, foi realizado um Update nos registos existentes, atualizando os campos relevantes, como customer_name, segment, age, city, state_name, country, postal_code, e region, com os valores do fluxo de dados.

 Update

Step name

Connection

Target schema

Target table

Commit size

Use batch updates? ☐

Skip lookup ☐

Ignore lookup failure? ☐ Flag field (key found)

The key(s) to look up the value(s):

#	Table field	Comparator	Stream field1	Stream field2
1	customer_id	=	Customer ID	

Update fields:

#	Table field	Stream field
1	customer_name	Customer Name
2	segment	Segment
3	age	Age
4	city	City
5	state_name	State
6	country	Country
7	postal_code	Postal Code
8	region	Region

Figura 25: UPDATE SALESDBCONNECTION - CUSTOMER TABLE.

Considerações Finais sobre a Transformação dos Clientes

Esta transformação foi essencial para garantir que todos os dados dos clientes estivessem consistentes, precisos e prontos para serem utilizados em análises futuras. Durante o processo, foram realizadas diversas etapas que incluíram a extração de dados de múltiplas fontes, a padronização de campos importantes como estado e país, e a eliminação de registos duplicados. Cada uma dessas etapas foi projetada para assegurar a qualidade dos dados e evitar problemas que pudessem comprometer a análise.

O uso de técnicas como o Fuzzy Matching garantiu a consistência dos nomes dos estados, corrigindo variações e assegurando uma uniformização eficaz. Além disso, a conversão da idade para o tipo Integer e o tratamento de valores incorretos no campo Age foram fundamentais para assegurar que as análises numéricas pudessem ser realizadas sem erros ou incongruências. A filtragem adicional aplicada com o Filter Rows também garantiu que apenas dados válidos fossem considerados, reforçando a qualidade do dataset final.

Outro ponto de destaque foi a agregação de dados provenientes de múltiplas fontes, criando uma visão integrada de cada cliente, o que permite uma análise mais aprofundada e detalhada do perfil dos consumidores da empresa. Esta integração foi possível graças ao uso de passos como Sorted Merge, que consolidaram as informações em um único fluxo robusto.

Por fim, a atualização dos dados no banco de dados, utilizando os passos Lookup/Update e Step Update, assegurou que a base de dados de clientes estivesse sempre sincronizada e refletisse as informações mais atuais, evitando discrepâncias e garantindo a confiabilidade das informações utilizadas pelos gestores da empresa.

Como resultado, esta transformação criou uma base de dados coesa, limpa e padronizada, facilitando a utilização dos dados em relatórios analíticos e contribuindo para uma melhor tomada de decisão por parte da empresa. A utilização cuidadosa de cada técnica de transformação foi fundamental para garantir a integridade dos dados, a minimização de erros e o sucesso da integração de informações de clientes em um único ambiente analítico.

postgres/postgres@PostgreSQL 17

Data Output Messages Notifications

	surr_id [PK] integer	customer_id character varying	customer_name character varying	segment character varying	age integer	city character varying	state_name character varying	country character varying	postal_code character varying	region character varying
1	102	BP-11185	Ben Peterman	Corporate	48	Arvada	Colorado	United States	80004	West
2	2	AA-10375	Allen Arnold	Consumer	22	Mesa	Arizona	United States	85204	West
3	4	AA-10645	Anna Andreadi	Consumer	32	Chester	Pennsylvania	United States	19013	East
4	6	AB-10060	Adam Bellavance	Home Office	25	New York City	New York	United States	10009	East
5	7	AB-10105	Adrian Barton	Consumer	63	Phoenix	Arizona	United States	85023	West
6	26	AG-10765	Anthony Garverick	Home Office	40	Philadelphia	Pennsylvania	United States	19120	East
7	27	AG-10900	Arthur Gainer	Consumer	56	Tucson	Arizona	United States	85705	West
8	28	AH-10030	Aaron Hawkins	Corporate	60	Philadelphia	Pennsylvania	United States	19134	East
9	29	AH-10075	Adam Hart	Corporate	21	New York City	New York	United States	10011	East
10	30	AH-10120	Adrian Hane	Home Office	27	Tucson	Arizona	United States	85705	West
11	31	AH-10195	Alan Haines	Corporate	67	Tamarac	Florida	United States	33319	South
12	32	AH-10210	Alan Hwang	Consumer	58	Brentwood	California	United States	94513	West
13	33	AH-10465	Amy Hunt	Consumer	24	New York City	New York	United States	10035	East
14	34	AH-10585	Angele Hood	Consumer	34	Chicago	Illinois	United States	60623	Central
15	35	AH-10690	Anna Haberlin	Corporate	39	New York City	New York	United States	10024	East
16	36	AI-10855	Arianne Irving	Consumer	35	Philadelphia	Pennsylvania	United States	19120	East
17	37	AJ-10780	Anthony Jacobs	Corporate	47	Springfield	Virginia	United States	22153	South
18	38	AJ-10795	Anthony Johnson	Corporate	27	Saint Petersburg	Florida	United States	33710	South
19	39	AJ-10945	Ashley Jarboe	Consumer	62	Wilmington	North Carolina	United States	28403	South
20	40	AJ-10960	Astrea Jones	Consumer	30	Rochester	New York	United States	14609	East
21	41	AM-10360	Alice McCarthy	Corporate	45	Grand Prairie	Texas	United States	75051	Central
22	42	AM-10705	Anne McFarland	Consumer	28	Auburn	Alabama	United States	36830	South

Figura 26: CUSTOMER TABLE IN POSTGRES.

Examine preview data

Rows of step: Update (793 rows)

#	Customer ID	Customer Name	Segment	Age	City	State	Country	Postal Code	Region	SURR_ID
1	AA-10315	Alex Avila	Consumer	66	Minneapolis	Minnesota	United States	55407	Central	1
2	AA-10375	Allen Arnold	Consumer	22	Mesa	Arizona	United States	85204	West	2
3	AA-10480	Andrew Allen	Consumer	50	Concord	North Carolina	United States	28027	South	3
4	AA-10645	Anna Andreadi	Consumer	32	Chester	Pennsylvania	United States	19013	East	4
5	AB-10015	Aaron Bergman	Consumer	66	Seattle	Washington	United States	98103	West	5
6	AB-10060	Adam Bellavance	Home Office	25	New York City	New York	United States	10009	East	6
7	AB-10105	Adrian Barton	Consumer	63	Phoenix	Arizona	United States	85023	West	7
8	AB-10150	Aimee Bixby	Consumer	65	Long Beach	New York	United States	11561	East	8
9	AB-10165	Alan Barnes	Consumer	22	Los Angeles	California	United States	90036	West	9
1.	AB-10255	Alejandro Ballentine	Home Office	34	Lorain	Ohio	United States	44052	East	10
1.	AB-10600	Ann Blume	Corporate	34	Tucson	Arizona	United States	85705	West	11
1.	AC-10420	Alyssa Crouse	Corporate	69	San Francisco	California	United States	94122	West	12
1.	AC-10450	Amy Cox	Consumer	46	Minneapolis	Minnesota	United States	55407	Central	13
1.	AC-10615	Ann Chong	Corporate	61	New York City	New York	United States	10009	East	14
1.	AC-10660	Anna Chung	Consumer	30	Huntsville	Texas	United States	77340	Central	15
1.	AD-10180	Alan Dominguez	Home Office	52	Houston	Texas	United States	77041	Central	16
1.	AF-10870	Art Ferguson	Consumer	63	College Station	Texas	United States	77840	Central	17
1.	AF-10885	Art Foster	Consumer	40	Louisville	Kentucky	United States	40214	South	18
1.	AG-10270	Alejandro Grove	Consumer	18	West Jordan	Utah	United States	84084	West	19
2.	AG-10300	Aleksandra Gannaway	Corporate	68	Los Angeles	California	United States	90049	West	20
2.	AG-10330	Alex Grayson	Consumer	51	Stockton	California	United States	95207	West	21
2.	AG-10390	Allen Goldenen	Consumer	47	Cincinnati	Ohio	United States	45231	East	22
2.	AG-10495	Andrew Gjertsen	Corporate	24	Philadelphia	Pennsylvania	United States	19140	East	23
2.	AG-10525	Andy Gerbode	Corporate	69	Saint Petersburg	Florida	United States	33710	South	24
2.	AG-10675	Anna Gayman	Consumer	69	Houston	Texas	United States	77036	Central	25
2.	AG-10765	Anthony Garverick	Home Office	40	Philadelphia	Pennsylvania	United States	19120	East	26
2.	AG-10900	Arthur Gainer	Consumer	56	Tucson	Arizona	United States	85705	West	27
2.	AH-10030	Aaron Hawkins	Corporate	60	Philadelphia	Pennsylvania	United States	19134	East	28
2.	AH-10075	Adam Hart	Corporate	21	New York City	New York	United States	10011	East	29
3.	AH-10120	Adrian Hane	Home Office	27	Tucson	Arizona	United States	85705	West	30
3.	AH-10195	Alan Haines	Corporate	67	Tamarac	Florida	United States	33319	South	31
3.	AH-10210	Alan Hwang	Consumer	58	Brentwood	California	United States	94513	West	32
3.	AH-10465	Amy Hunt	Consumer	24	New York City	New York	United States	10035	East	33
3.	AH-10585	Angele Hood	Consumer	34	Chicago	Illinois	United States	60623	Central	34
3.	AH-10690	Anna Haberlin	Corporate	39	New York City	New York	United States	10024	East	35
3.	AI-10855	Arianne Irving	Consumer	35	Philadelphia	Pennsylvania	United States	19120	East	36

Figura 27: PREVIEW DO UPDATE NO PDI.

Product Transformation

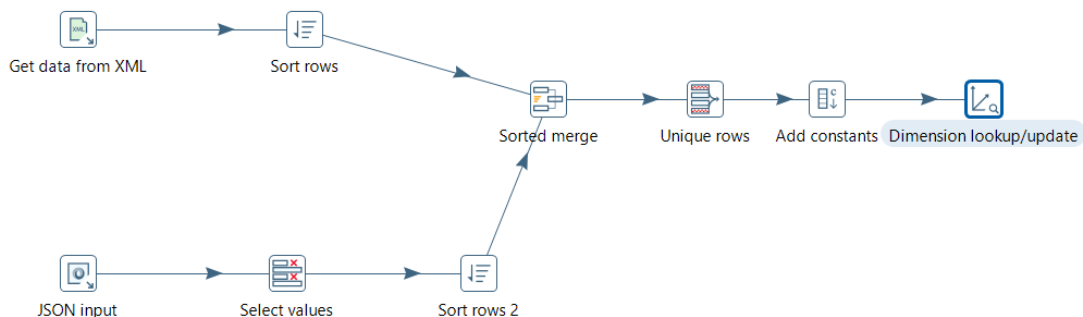


Figura 28: PRODUCT TRANSFORMATION.

A transformação, designada Product Transformation, utiliza dados em formatos XML e JSON e executa vários passos para integrar e ordenar as informações de produtos. Abaixo, encontram-se os detalhes da configuração de cada step e o respetivo papel no processo de transformação.

1: Get Data from XML

O primeiro step, Get Data from XML, foi configurado para extrair dados de um ficheiro XML que contém informações sobre produtos. No campo Look XPath, foi especificado /Rows/Row para identificar cada linha de dados dentro do ficheiro XML. Os campos extraídos do ficheiro XML são configurados da seguinte forma:

Name	XPath	Element	Result type	Type	Length
Product_ID	Product_ID	Node	Value of	String	20
Category	Category	Node	Value of	String	50
Sub_Category	Sub_Category	Node	Value of	String	50
Product_Name	Product_Name	Node	Value of	String	200

Figura 29: CONFIGURAÇÃO DA EXTRAÇÃO DE DADOS DO AQRUIVO XML.

Get data from XML

Nome do Step: **Get data from XML**

File | Content | Fields | Additional output fields

XML source from field

XML source is defined in a field? ☐

XML source is a filename? ☐

Read source as Uri ☐

get XML source from a field:

File or directory: **Add** **Browse**

Regular Expression:

Exclude Regular Expression:

Selected files:

#	File/Directory	Wildcard (RegExp)	Exclude wildcard
1	\$(ROOT_PATH=)/Product Data/ProductDataAsXML.xml.xml		

Delete **Edit**

Show filename(s)...

Help **OK** **Preview rows** **Cancela**

Figura 30: GET DATA FROM XML: FILE.

Get data from XML

Nome do Step: **Get data from XML**

File | Content | Fields | Additional output fields

Settings

Loop XPath: **Get XPath nodes**

Encoding:

Namespace aware? ☐

Ignore comments? ☐

Validate XML? ☐

Use token? ☐

Ignore empty file? ☐

Do not raise an error if no files ☒

Limit:

Prune path to handle large files:

Additional fields

Include filename in output? ☐ Filename fieldname:

Rownum in output? ☐ Rownum fieldname:

Add to result filename

Add files to result filename ☐

Help **OK** **Preview rows** **Cancela**

Figura 31: GET DATA FROM XML: CONTENT.

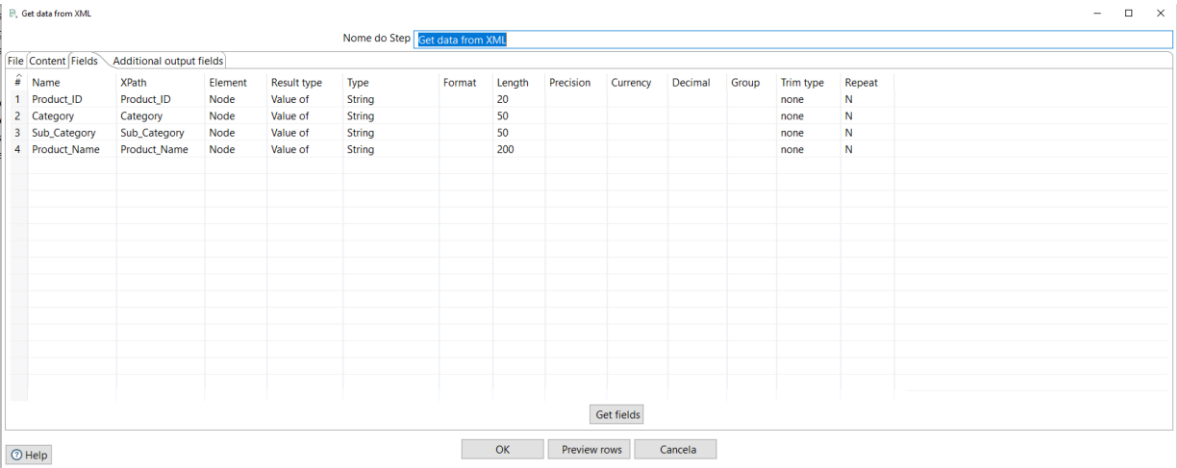


Figura 32: GET DATA FROM XML: FIELDS.

2: JSON Input

O step JSON Input, configura-se a extração de dados a partir de um ficheiro JSON. Tal como no passo anterior, este step permite obter informações sobre produtos, complementando os dados XML. Os campos extraídos a partir do ficheiro JSON são definidos conforme a tabela abaixo:

Name	Path	Type	Length
Sub_Category	\$..[*].Sub_Category	String	20
Category	\$..[*].Category	String	50
Product_ID	\$..[*].Product_ID	String	20
Product_Name	\$..[*].Product_Name	String	200

Figura 33: CONFIGURAÇÃO DOS DADOS EXTRAÍDOS O FICHEIRO JSON.

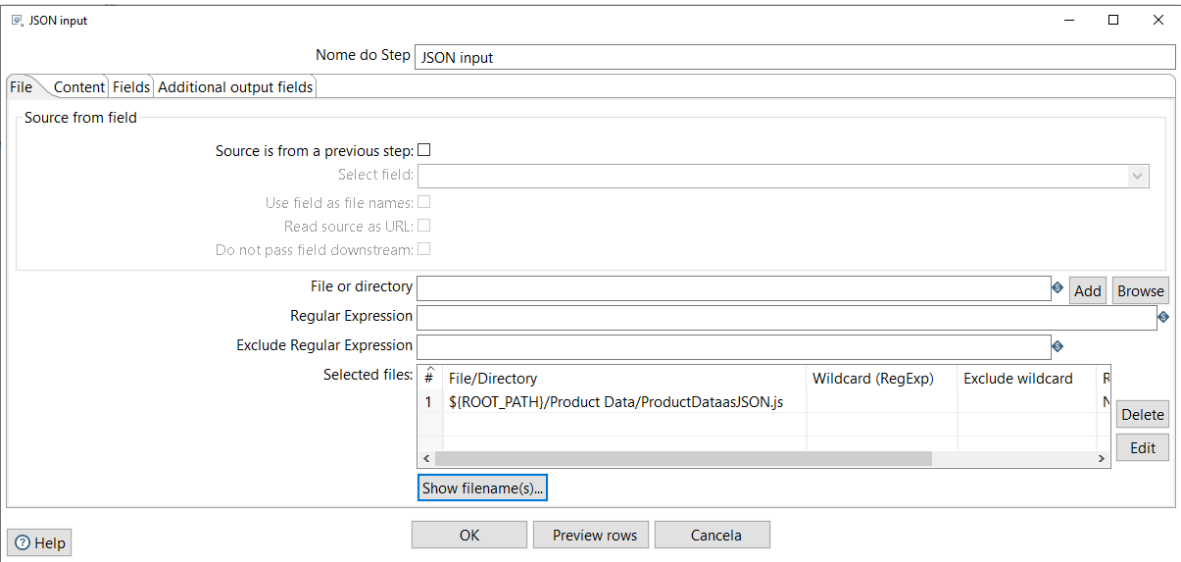


Figura 34: JSON INPUT:FILE.

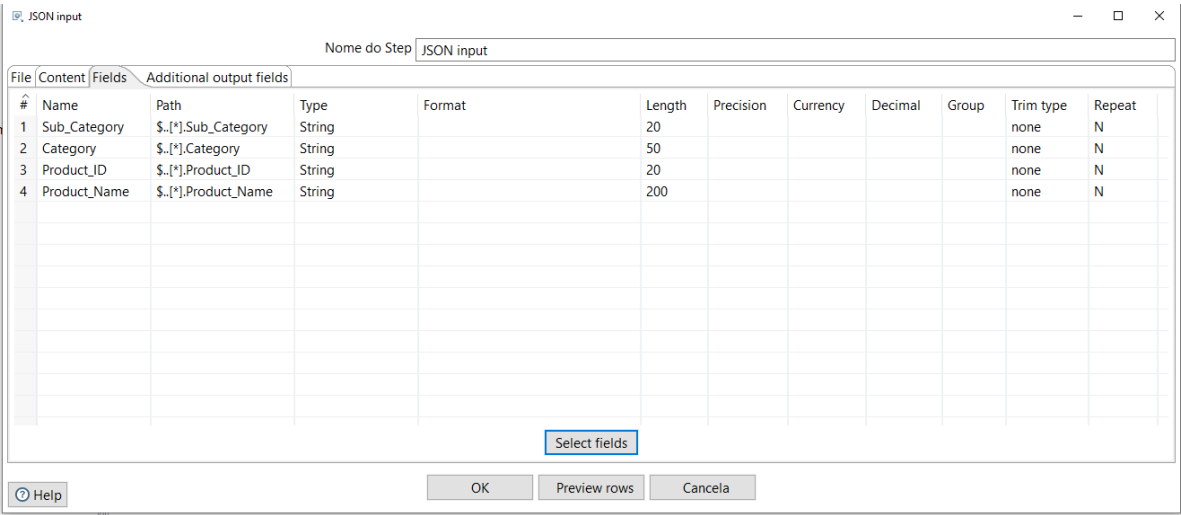


Figura 35: JSON INPUT: FIELDS.

3: Sort Rows

Após a extração dos dados dos ficheiros XML e JSON, é necessário ordenar as linhas para facilitar a integração dos dados. No step Sort Rows, os dados provenientes do XML são organizados de acordo com o campo Product_ID. Estes dados serão ordenado pelo campo Product_ID.

[illegible]

Figura 36: SORT ROWS.

4: Select Values

O step Select Values permite que os dados extraídos do JSON avancem com os campos necessários para a integração. A configuração garante que apenas os campos desejados sigam para o próximo passo, sem renomeações ou ajustes adicionais.

Select and Alter Fieldname
Product_ID
Category
Sub_Category
Product_Name

Figura 37: CAMPOS SELECT VALUES.

[illegible]

Figura 38: SELECT VALUES: SELECT&ALTER.

Step 5: Sort Rows

Após o step Select Values, os dados do JSON passam por um step adicional de ordenação, Sort Rows, para garantir que estejam alinhados e preparados para integração com os dados XML. Este step realiza a ordenação com base no campo Product_ID.

[illegible]

Figura 39: SORT ROWS.

Step 6: Sorted Merge

Após ambos os passos de ordenação dos dados extraídos dos ficheiros XML e JSON, estes são integrados no step Sorted Merge, configurado para realizar a união dos dados com base no campo Product_ID utilizado como chave para a integração dos dados. Este passo garante a integração correta dos registos provenientes das duas fontes.

Sorted merge

Step name

Fields :

#	Fieldname	Ascending
1	Product_ID	S

K

Figura 40: SORTED MERGE.

Step 7: Unique Rows

Após a junção dos dados, o step Unique Rows é utilizado para garantir que apenas registos únicos avancem no fluxo, evitando duplicação. O Campo Product_ID é o campo pelo qual a unicidade será avaliada.

Unique rows

Step name

Settings

Add counter to output? ☐ Counter field

Redirect duplicate row ☐ Error description

Fields to compare on (no entries means: compare complete row)

#	Fieldname	Ignore case	
1	Product_ID	N	

Figura 41: UNIQUE ROWS.

Step 8: Add Constants

O step Add Constants é utilizado para adicionar um campo com uma data de efetividade constante para todos os registos processados. Este campo será importante para definir a validade temporal dos dados quando estes forem carregados na tabela de dimensão.

[illegible]

Figura 42: ADD CONSTANTS.

Step 9: Dimension Lookup/Update

O step Dimension Lookup/Update conecta a transformação à base de dados onde a tabela de produtos será atualizada, inserindo novos registos ou atualizando registos existentes conforme necessário. A Keys define o campo de correspondência entre o fluxo de dados e a tabela de destino. A Technical Key Field `suur_id` é o campo na tabela de dimensão utilizado como chave técnica para identificação única de registos. A Stream Date Field `effective date` é o campo no fluxo que contém a data de efetividade para a atualização de registos na dimensão.

Fields configura a ação a ser realizada para cada campo da dimensão:

Dimension Field	Stream Field to Compare With	Type of Dimension Update
category	Category	Insert
sub_category	Sub_Category	Insert
product_name	Product_Name	Insert
current	Last version (sem fonte)	N/A
lastupdate	effective date	Update

Figura 43: DIMENSION LOOKUP/UPDATE: FIELDS.

Dimension lookup/update

Step name:

Update the dimension? ☒

Connection:

Target schema:

Target table:

Commit size:

Enable the cache? ☒

Pre-load the cache? ☐

Cache size in rows (0 = cache all):

Keys Fields

Key fields (to look up row in dimension):

#	Dimension field	Field in stream
1	product_id	Product_ID

Technical key field:

Creation of technical key

☒ Use table maximum + 1

☐ Use sequence

☐ Use auto increment field

Version field:

Stream Datefield:

Date range start field: Min. year:

Use an alternative start date? ☐

Table date range end: Max. year:

Figura 44: DIMENSION LOOKUP/UPDATE: KEYS.

Considerações Finais sobre a Transformação dos Produtos

A **Product Transformation** foi desenvolvida para integrar dados de produtos a partir de múltiplas fontes (XML e JSON), convertendo-os numa estrutura única e consistente para inserção numa tabela de dimensão de produtos, localizada na base de dados de vendas. Este processo automatiza a integração de dados de diferentes formatos, garantindo que os registos sejam organizados, ordenados e únicos antes do carregamento.

Os dados são extraídos dos ficheiros XML e JSON, com os campos relevantes definidos para que informações como **Product_ID**, **Category**, **Sub_Category** e **Product_Name** sejam uniformemente capturadas. Com o uso de steps de ordenação para cada fluxo (um após o **Get Data from XML** e outro após o **Select Values** dos dados JSON), os dados são preparados para integração. No step **Sorted Merge**, estes dados são finalmente unificados com base em **Product_ID**, mantendo a consistência e precisão dos registos.

Para assegurar a qualidade dos dados, o step **Unique Rows** remove duplicados, e um campo de **effective date** é acrescentado a todos os registos. Este campo de data é fundamental para rastrear a validade temporal de cada entrada na dimensão de produtos.

Por fim, no **Dimension Lookup/Update**, os dados são carregados para a tabela de dimensão de produtos na base de dados. Este step está configurado para adicionar novos registos ou atualizar registos existentes, preservando um histórico das mudanças através dos campos de data de início e fim de validade. Esta configuração é crítica para garantir que a tabela de dimensão esteja atualizada e que todas as alterações possam ser rastreadas historicamente, o que é essencial para relatórios precisos e uma visão completa dos produtos ao longo do tempo.

A **Product Transformation** alcança, assim, uma integração robusta e automatizada, proporcionando uma base de dados fiável para consultas e análises avançadas.

Sales Transformation

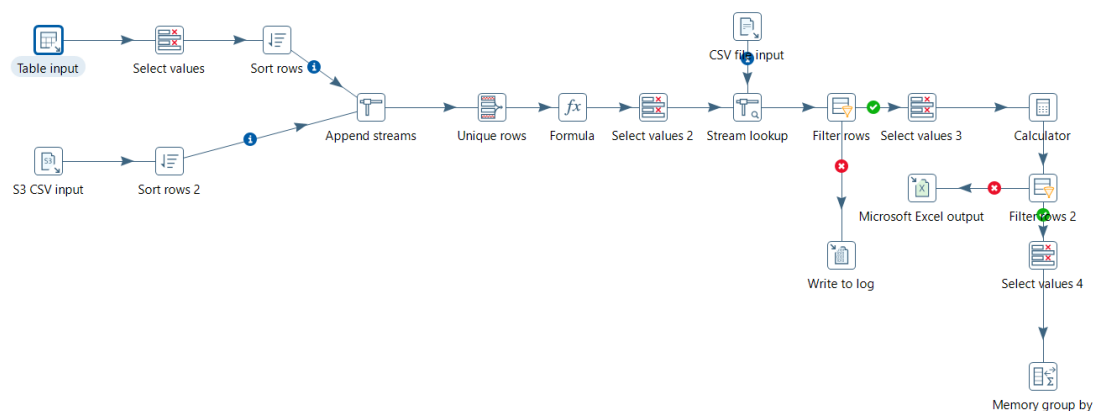


Figura 45: SALES TRANSFORMATION.

A **Sales Transformation** foi criada para integrar dados de vendas provenientes de uma base de dados relacional e de um ficheiro CSV armazenado no Amazon S3. Esta transformação visa consolidar dados de vendas, garantindo que os registos sejam uniformes em termos de estrutura e ordenados antes de serem unificados.

Step 1: Table Input

O primeiro step, **Table Input**, realiza a extração de dados diretamente da base de dados configurada como **salesDBconnection**. Este step executa uma consulta SQL para capturar todos os dados da tabela de vendas, com a seguinte instrução: `select * from sales;`

Table input

Step name

Connection

SQL

```
SELECT * FROM sales
```

Line 1 Column 0

Store column info in step meta data ☐

Enable lazy conversion ☐

Replace variables in script? ☐

Insert data from step

Execute for each row? ☐

Limit size

Figura 46: TABLE INPUT.

Step 2: S3 CSV Input

O step **S3 CSV Input** extrai dados de vendas de um ficheiro CSV armazenado num bucket do Amazon S3. Este ficheiro contém vendas históricas armazenadas na cloud e é essencial para complementar os dados da base de dados. Tem no bucket onde o ficheiro está armazenado e o nome do mesmo. Para este step funcionar tive de configurar os acessos colocando os seguintes dados:

- ACCESS KEY: AKIA3FRRI2Z35P5SK4WC;

- SECRET KEY: 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;
- arn:aws:s3:::oldsalesdataaws.

Step name: S3 CSV input

S3 Bucket: oldsalesdataaws Select bucket

Filename: SalesforAWSandHadoop.csv Navega...

Delimiter: , Insert TAB

Enclosure: "

Max line size: 5000

Lazy conversion? ☐

Header row present? ☒

The row number field name (optional):

Running in parallel? ☐

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	Order_Line	Integer	#	15	0	EUR	.	,	none
2	Order_ID	String		14		EUR	.	,	none
3	Order_Date	Date	MM/dd/yyyy			EUR	.	,	none
4	Ship_Date	Date	MM/dd/yyyy			EUR	.	,	none
5	Ship_Mode	String		14		EUR	.	,	none
6	Customer_ID	String		8		EUR	.	,	none
7	Product_ID	String		15		EUR	.	,	none
8	Sales	Number	##	8	2	EUR	.	,	none
9	Quantity	Integer	#	15	0	EUR	.	,	none
1..	Discount	Number	##	4	2	EUR	.	,	none
1..	Profit	Number	##0.###	8	2	EUR	.	,	none

Help OK Cancela Preview Obtem campos

Figura 47: S3 CSV INPUT.

Step 3: Select Values

Após a extração dos dados da base de dados, o step Select Values é utilizado para garantir que os campos estejam corretamente configurados e correspondam à estrutura de dados esperada, especialmente para que os dados possam ser facilmente integrados com os do CSV mais tarde.

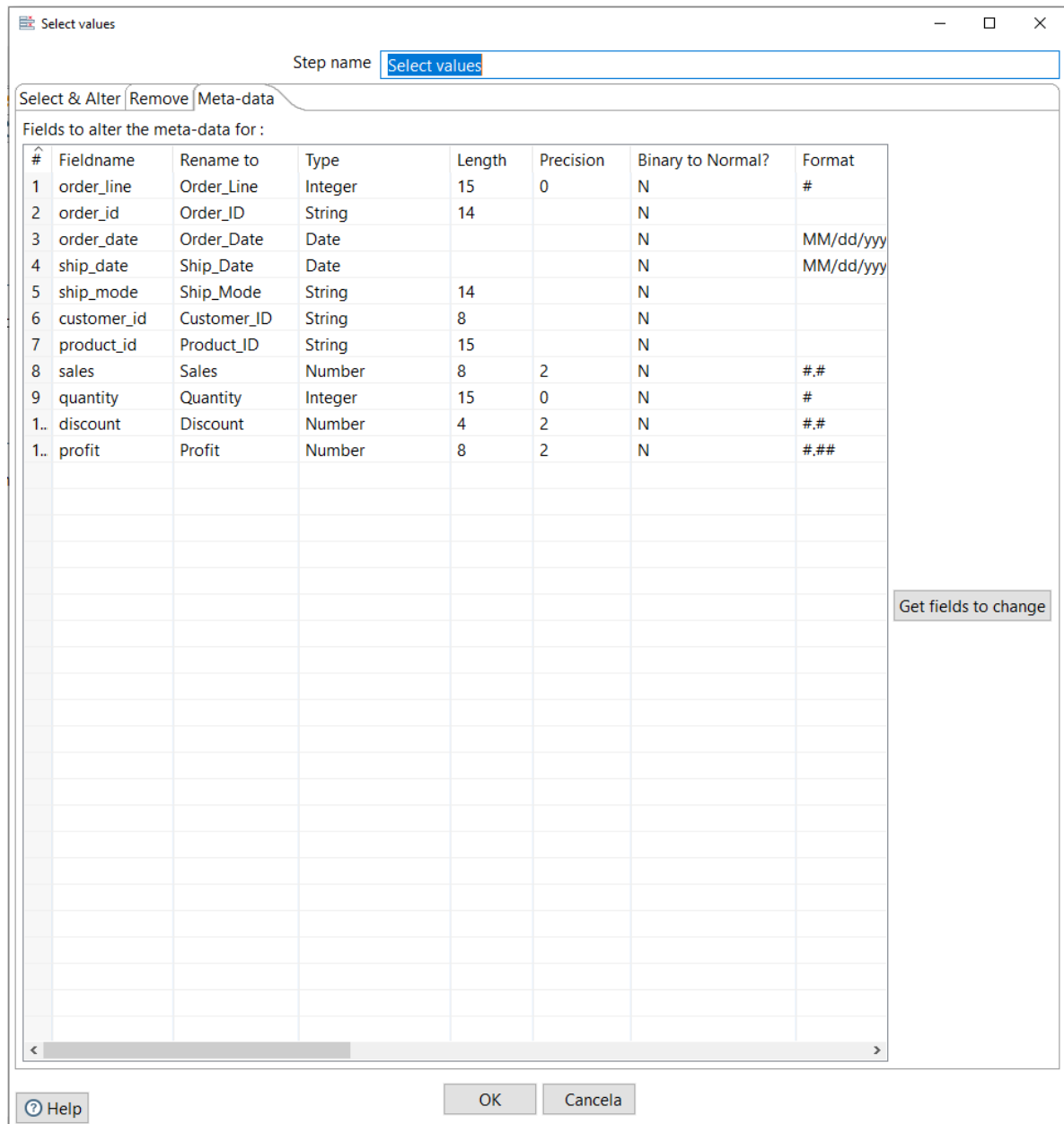


Figura 48: SELECT VALUES.

Step 4: Sort Rows (Table Input)

Após o **Select Values**, os dados do **Table Input** passam por um step **Sort Rows** para organizar os registos de acordo com o campo **Order_Line**, facilitando a integração posterior com os dados CSV.

[illegible]

Figura 49: SORT ROWS (TABLE INPUT).

Step 5: Sort Rows (S3 CSV Input)

Os dados do **S3 CSV Input** também passam por um step **Sort Rows** com a mesma configuração, ordenando-os pelo campo **Order_Line**. Este passo garante que a ordenação dos dados do CSV coincida com a dos dados do **Table Input**, essencial para uma integração correta.

Sort rows

Nome do Step:

Sort directory: Navega...

TMP-file prefix:

Sort size (rows in memory):

Free memory threshold (in %):

Compress TMP Files? ☐

Only pass unique rows? (verifies keys only) ☒

Fields :

#	Fieldname	Ascending	Case sensitive compare?	Sort based on current locale?	Collator Strength	Presorted?
1	Order_Line	S	N	N	0	N

? Help OK Cancela Obtem campos

Figura 50: SORT ROWS (S3 CSV INPUT).

Step 6: Append Streams

Com os dados ordenados, o step **Append Streams** é utilizado para unir os dados dos dois fluxos de forma sequencial, permitindo que os registros da base de dados e do CSV estejam no mesmo fluxo de dados para processamento adicional. O fluxo do S3 CSV INPUT será colocado no início da sequência e o fluxo da Table Input será colocado no final da sequência.

Step name: Append streams

Head hop: Sort rows 2

Tail hop: Sort rows

Help OK Cancel

Figura 51: APPEND STREAMS.

Step 7: Unique Rows

Após a união dos fluxos, o step **Unique Rows** é configurado para assegurar que apenas registos únicos avancem, com base no campo **Order_Line**, prevenindo duplicação nos dados de vendas integrados.

Step name: Unique rows

Settings

Add counter to output? ☐ Counter field:

Redirect duplicate row ☐ Error description:

Fields to compare on (no entries means: compare complete row)

#	Fieldname	Ignore case
1	Order_Line	N

Buttons: ? Help, OK, Cancela, Get

Figura 52: UNIQUE ROWS.

Step 8: Formula

No step **Formula**, é aplicado um cálculo ao campo **Discount** para converter os valores de desconto em percentagem. Este cálculo facilita o tratamento posterior dos dados financeiros. O resultado será armazenado no campo Discount e este substituirá o valor que estava armazenado nesse campo.

fx Formula

Nome do Step Formula

Fields:

#	New field	Formula	Value type	Length	Precision	Replace value
1	Discount	[Discount]*100	Integer			Discount

Help OK Cancela

Figura 53: FÓRMULA.

Step 9: CSV File Input (Product Lookup)

Este step permite a importação de um ficheiro CSV separado que contém informações adicionais de produtos, como o **Product ID** e **Product Name**. Estes dados serão utilizados para verificar e complementar os registos de vendas através do **Stream Lookup**.

CSV file input

Step name CSV file input

Filename \${ROOT_PATH}/Product Data/ProductLookupForSales.csv Navega...

Delimiter , Insert TAB

Enclosure "

NIO buffer size 50000

Lazy conversion? ☒

Header row present? ☒

Add filename to result ☐

The row number field name (optional)

Running in parallel? ☐

New line possible in fields? ☐

Format mixed

File encoding

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	Product ID	String		15		EUR	.	,	none
2	Product Name	String		98		EUR	.	,	none

Help OK Obtem campos Preview Cancela

Figura 54: CSV FILE INPUT.

Step 10: Stream Lookup

O step **Stream Lookup** permite verificar os dados do fluxo de vendas com base nas informações de produto carregadas do ficheiro CSV. Este step procura valores adicionais, como o **Product Name**, com base no **Product ID**, e insere-os nos dados de vendas. O CSV 's onde são obtidos os dados de lookup, a chave de pesquisa é o Product_ID e o campo de correspondência no CSV é o Product ID. O Product Name é o campo adicional a ser recuperado e inserido nos dados de vendas.

Stream lookup

Step name:

Lookup step:

The key(s) to look up the value(s):

#	Field	LookupField
1	Product_ID	Product ID

Specify the fields to retrieve :

#	Field	New name	Default	Type
1	Product Name			None

☒ Preserve memory (costs CPU)
☐ Key and value are exactly one integer field
☐ Use sorted list (i.s.o. hashtable)

Figura 55: STREAM LOOKUP.

Step 11: Filter Rows

O **Filter Rows** é utilizado para separar os registos com base na condição de existência do **Product Name**. Caso o **Product Name** não esteja presente, o registo será enviado para um step de logging para referência de inconsistências. A condição é se o Product Name não está vazio, se for verdade o registo é enviado para o próximo passo se for falso envia para o log a falta de correspondência.

Filter rows

Step name: Filter rows

Send 'true' data to step: Select values 3

Send 'false' data to step: Write to log

The condition:

+

Product Name <> <field> <value> (String)

Help OK Cancela

Figura 56: FILTER ROWS.

Step 12: Write to Log (False Condition)

Para registos onde o **Product Name** não é encontrado, o step **Write to Log** é utilizado para registar uma mensagem de erro, permitindo identificar eventuais problemas de correspondência de dados no lookup de produtos. Todos os campos de vendas são listados para facilitar a revisão de dados incompletos.

Write to log

Step name

Log level

Print header ☒

Limit rows? ☐

Nr of rows to print

Write to log

Fields

#	Field	
1	Order_Line	
2	Order_ID	
3	Order_Date	
4	Ship_Date	
5	Ship_Mode	
6	Customer_ID	
7	Product_ID	
8	Sales	
9	Quantity	
1..	Discount	
1..	Profit	
1..	Product Name	

? Help OK Obtem campos Cancela

Figura 57: WRITE TO LOG.

Step 13: Select Values 3 (Remove Product Name)

Para registos que passaram na verificação do **Product Name**, o **Select Values 3** remove este campo dos dados, uma vez que já foi verificado e a presença de informações adicionais não é mais necessária para os passos seguintes.

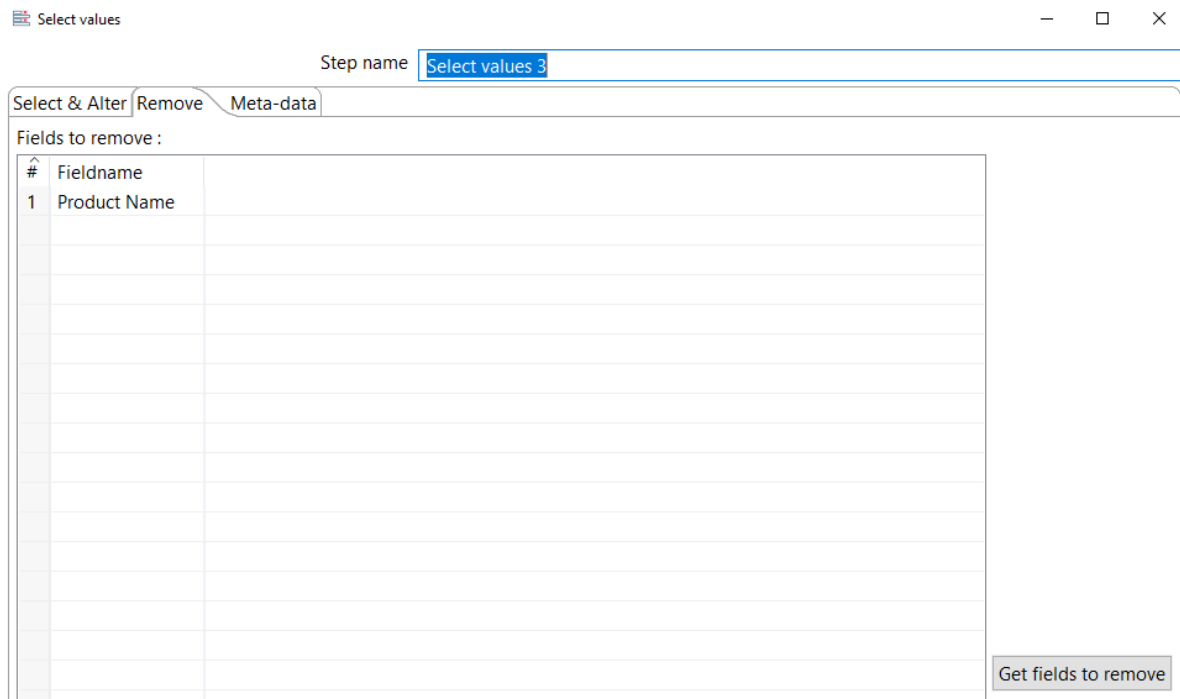


Figura 58: SELECT VALUES 3.

Step 14: Calculator

No step **Calculator**, é calculado o número de dias entre a data de encomenda (**Order_Date**) e a data de envio (**Ship_Date**). Este cálculo é importante para verificar se as datas estão corretas e, consequentemente, identificar possíveis problemas de integridade nos dados. É criado um campo number of days onde armazena o cálculo efetuado, fazendo a diferença entre a Ship_Date e a Order_Date.

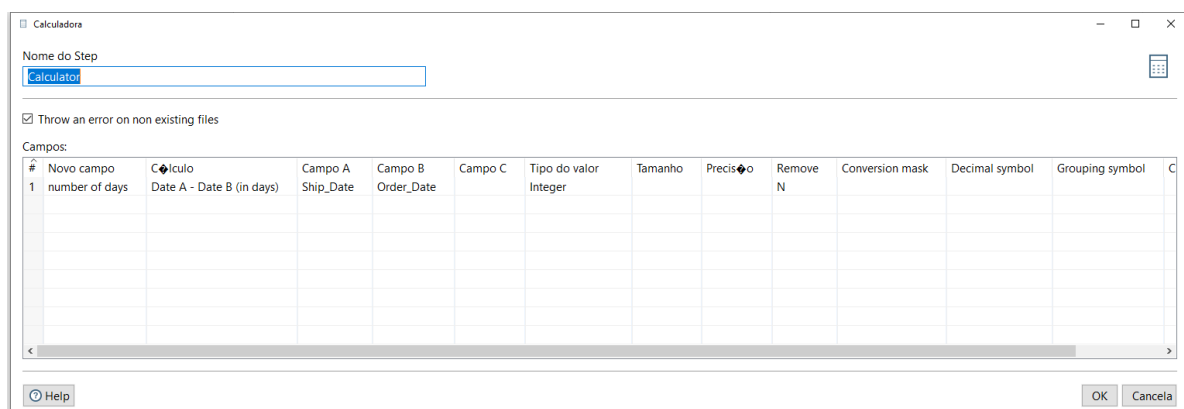


Figura 59: CALCULATOR.

Step 15: Filter Rows 2

O **Filter Rows 2** utiliza o campo calculado **number of days** para validar os dados de data. Se o valor for negativo, o registo é considerado incorreto e é enviado para um ficheiro Excel para posterior revisão. Os registos válidos seguem para o próximo passo. A condição foi de que o valor do number of days tem de ser ≥ 0 se for verdade passa para o próximo step se for falso é enviado para um ficheiro excel.

Filter rows

Step name:

Send 'true' data to step:

Send 'false' data to step:

The condition:

(Integer)

Figura 60: FILTER ROWS 2.

Step 16: Microsoft Excel Output (Sales_Dates_Incorrect)

Para registos onde a condição de data é falsa, o step **Microsoft Excel Output** cria um ficheiro XLS para armazenar esses dados incorretos. Esse ficheiro permite identificar e corrigir inconsistências de datas nos dados de vendas.

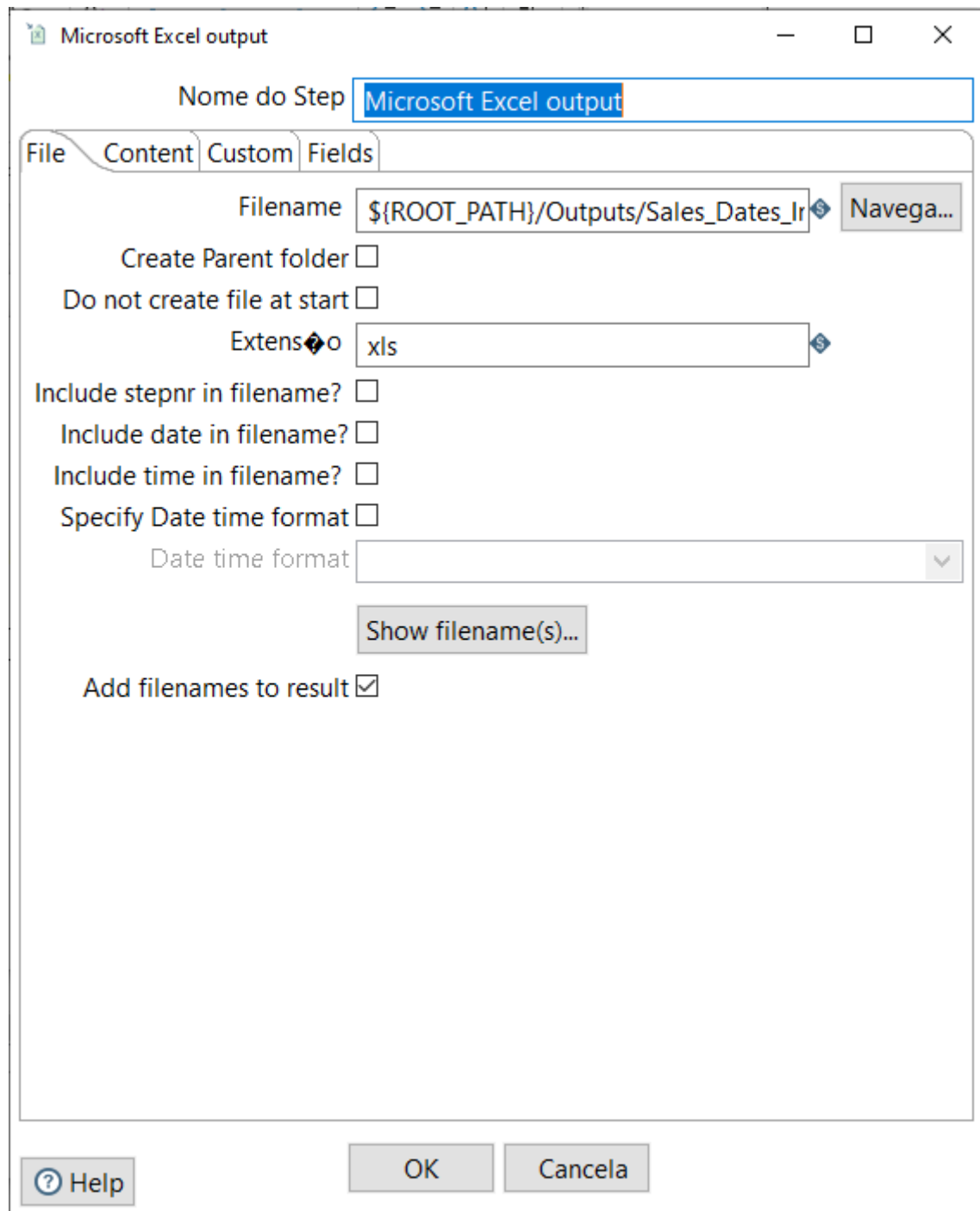


Figura 61: MICROSOFT EXCEL OUTPU: FILE.

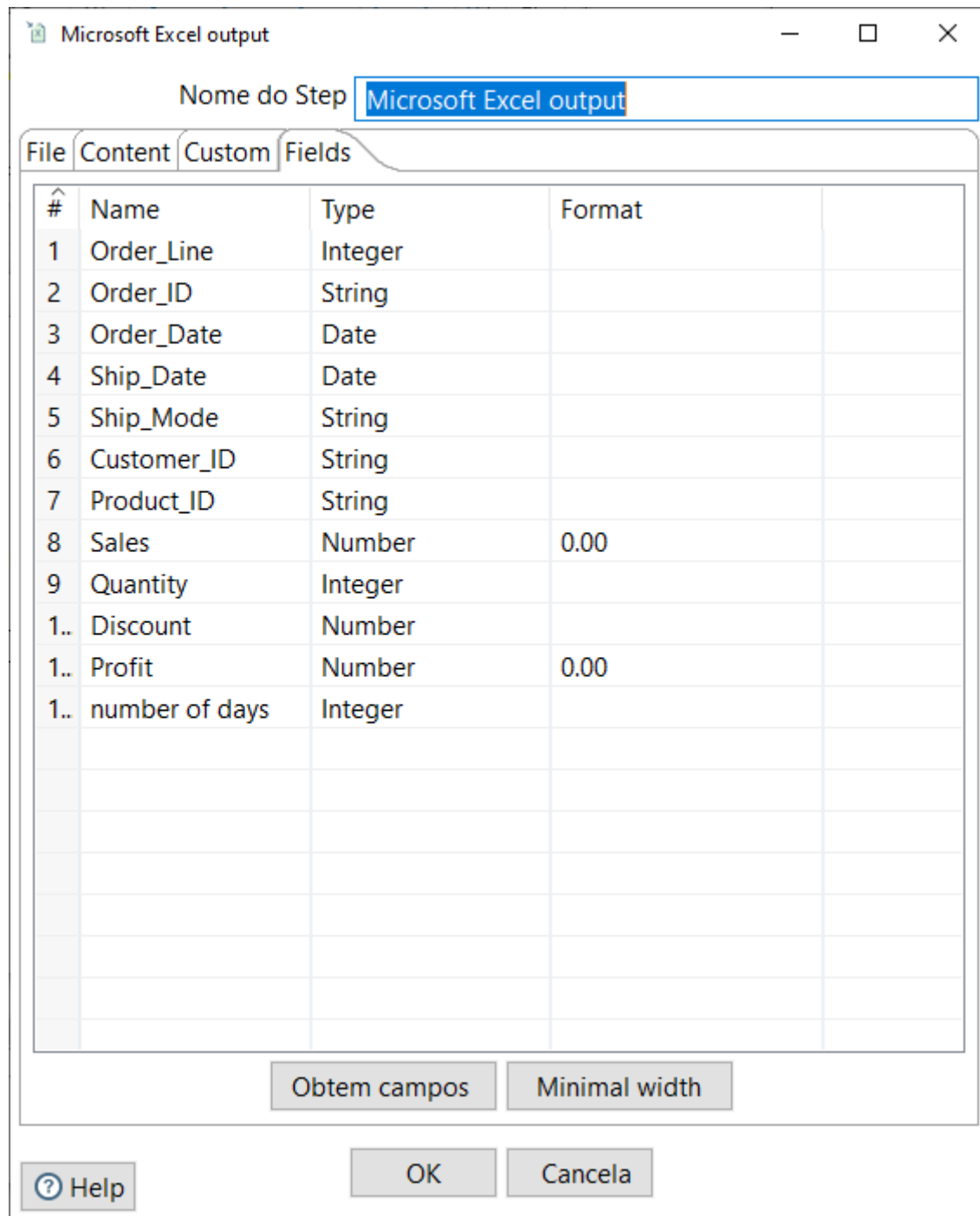


Figura 62: MICROSOFT EXCEL OUTPUT: FIELDS.

Step 17: Select Values 4

No step **Select Values 4**, o campo **number of days**, utilizado para validação, é removido dos registos que avançaram no fluxo, pois não será necessário para os passos finais.

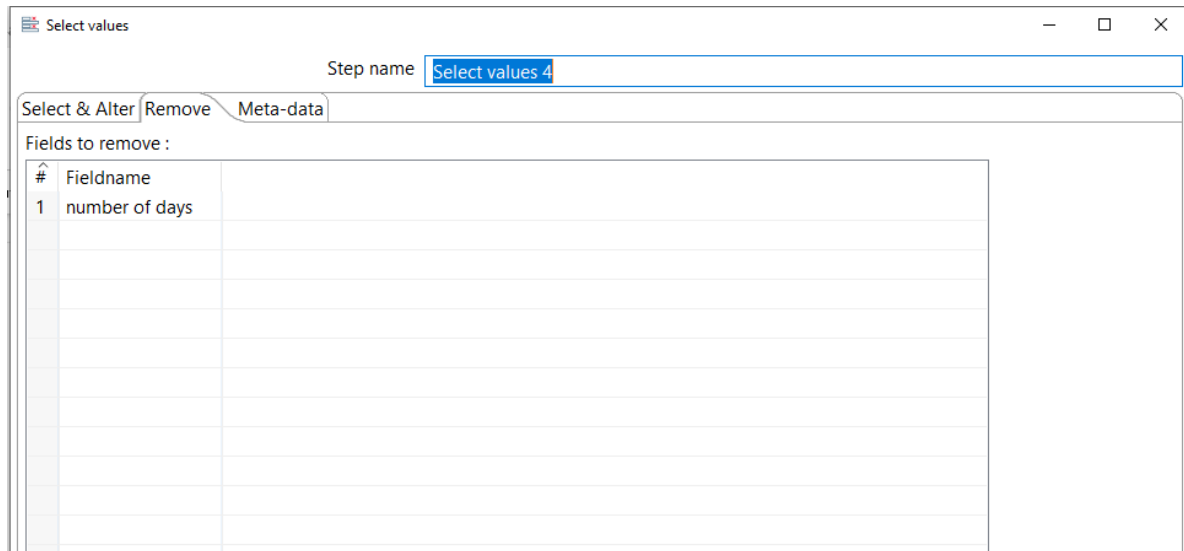


Figura 63: SELECT VALUES 4.

Step 18: Memory Group By

O último step, **Memory Group By**, agrupa os dados pelo **Product_ID** e calcula a soma total das vendas para cada produto. Esta agregação permite obter uma visão consolidada das vendas de cada produto.

Memory group by

Step name

Always give back a result row ☐

The fields that make up the group:

#	Group field	
1	Product_ID	

Get Fields

Aggregates :

#	Name	Subject	Type	Value
1	Sum of sales	Sales	Sum	

Get lookup fields

Help OK Cancela

Figura 64: MEMORY GROUP BY.

Considerações Finais sobre a Transformação das Vendas

A **Sales Transformation** integra dados de vendas provenientes de uma base de dados relacional e de um ficheiro CSV no Amazon S3, criando um único fluxo de dados consolidado e validado. Esta transformação utiliza diversos steps no Pentaho para uniformizar, ordenar e verificar os dados de vendas antes de serem analisados ou exportados.

Inicialmente, são extraídos dados de ambas as fontes e padronizados para garantir que os campos e formatos correspondem, facilitando a ordenação e integração subsequente. Através de um processo de validação, é assegurado que os registos duplicados são eliminados e que valores financeiros, como descontos, estão formatados corretamente.

Durante o fluxo, é feito um lookup dos dados de produtos para associar informações adicionais, como o nome dos produtos, a cada transação. Caso os produtos não sejam encontrados, esses registos são registados num log, permitindo uma análise e correção posterior. Outro passo de validação é aplicado nas datas de encomenda e envio, com os registos incorretos sendo exportados para um ficheiro Excel para posterior análise.

O passo final da transformação agrega as vendas por **Product_ID**, permitindo calcular o total de vendas para cada produto. Esta agregação fornece uma visão geral das vendas por produto e facilita análises de desempenho de vendas para suportar decisões estratégicas.

Em resumo, a **Sales Transformation** automatiza a consolidação e validação de dados de vendas, assegurando a integridade e qualidade dos dados antes de serem utilizados em relatórios ou análises de vendas.

Job

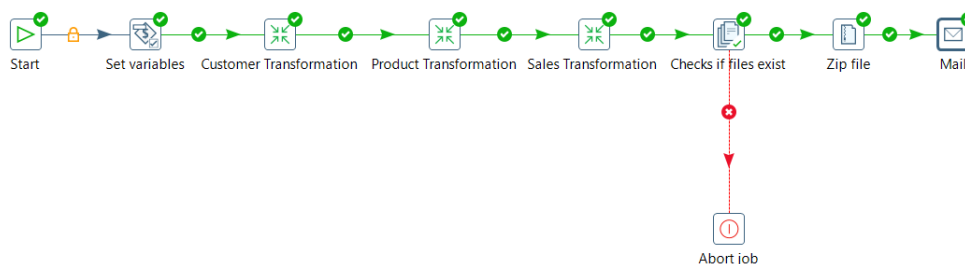


Figura 65: JOB.

Para garantir que todas as transformações necessárias para o processamento dos dados de produtos e vendas sejam executadas de forma automatizada e sequencial, foi desenvolvido um **Job** no Pentaho Data Integration. Este job orquestra a execução das transformações **Product Transformation** e **Sales Transformation**, mas também aplica verificações para confirmar a criação de ficheiros, zipa e envia ficheiros por e-mail se necessário, centralizando todo o fluxo de dados numa operação coordenada, de forma a garantir que os dados de vendas e produtos são extraídos, transformados e carregados com integridade e consistência.

O **Job** foi configurado para iniciar cada transformação de acordo com uma sequência específica, aplicando validações e verificações conforme necessário. Desta forma, o processo completo é otimizado, reduzindo o risco de erros e assegurando que as dependências entre as transformações são respeitadas. Abaixo, detalham-se os steps e configurações implementados no job para alcançar uma execução eficiente e precisa.

Step 1: Start

O job inicia com o step **Start**, que define o início da execução dos passos subsequentes.

Step 2: Set Variables

No step **Set Variables**, foram definidas variáveis que configuram informações importantes para a execução das transformações e para o envio de e-mails. Estas variáveis garantem que o job tem acesso a diretórios, configurações de servidor de e-mail e outros detalhes necessários para uma execução bem-sucedida.

Job entry name:

Properties file

Name of properties file:

Variable scope:

Settings

Variable substitution? ☒

Variables :

#	Variable name	Value	Variable scope type
1	server	smtp.gmail.com	Valid in the Java Virtual Machine
2	port	465	Valid in the Java Virtual Machine
3	username	susanapvl@gmail.com	Valid in the Java Virtual Machine
4	\${ROOT_PATH}	D:\LESI_3ANO\1_SEMESTRE\ISI\TRABALHO_PRATICO_ISI_1_SALES\Data/	Valid in the root job
5	\$(FILES_TO_SEND)	D:\LESI_3ANO\1_SEMESTRE\ISI\TRABALHO_PRATICO_ISI_1_SALES\Temp\FilesToSendEmail	Valid in the root job
6	\$(ZIP_OUTPUT_PATH)	D:\LESI_3ANO\1_SEMESTRE\ISI\TRABALHO_PRATICO_ISI_1_SALES\Data/Outputs	Valid in the Java Virtual Machine

Help OK Cancela

Figura 66: SET VARIABLES.

Step 3: Customer Transformation

Após a definição das variáveis, o job executa a transformação **Customer Transformation**, que processa e prepara os dados de clientes para integração com os dados de produtos e vendas.

Step 4: Product Transformation

Em sequência, o job inicia a transformação **Product Transformation**, responsável por consolidar e estruturar os dados de produtos, essencial para a análise conjunta com os dados de vendas.

Step 5: Sales Transformation

A transformação **Sales Transformation** é então executada, integrando os dados de vendas, completando assim o processo de ETL dos dados de clientes, produtos e vendas.

Step 6: Check if Files Exist

Após a execução das transformações, o step **Check if Files Exist** verifica se o ficheiro Sales_Dates_Incorrect.xls foi gerado e está localizado no diretório esperado. Esta verificação assegura que o ficheiro de dados incorretos foi criado, sinalizando potenciais problemas de integridade nos dados de datas. Se o ficheiro existir, o fluxo avança para o próximo passo de zipping; caso contrário, o fluxo segue para o step **Abort Job**.

Step 7: Abort Job (Caso o ficheiro não exista)

Caso o ficheiro não seja encontrado, o step **Abort Job** é acionado, interrompendo o job e registando uma mensagem no log para indicar o motivo da falha que diz o seguinte : "The JOB failed because the file was not in location."

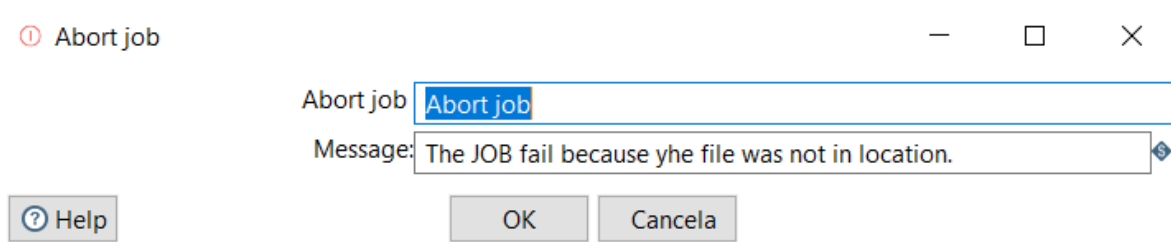


Figura 67: ABORT JOB.

Step 8: Zip File (Caso o ficheiro exista)

Se o ficheiro for encontrado, o step **Zip File** é utilizado para comprimir o ficheiro Sales_Dates_Incorrect.xls e movê-lo para o diretório designado para envio por e-mail.

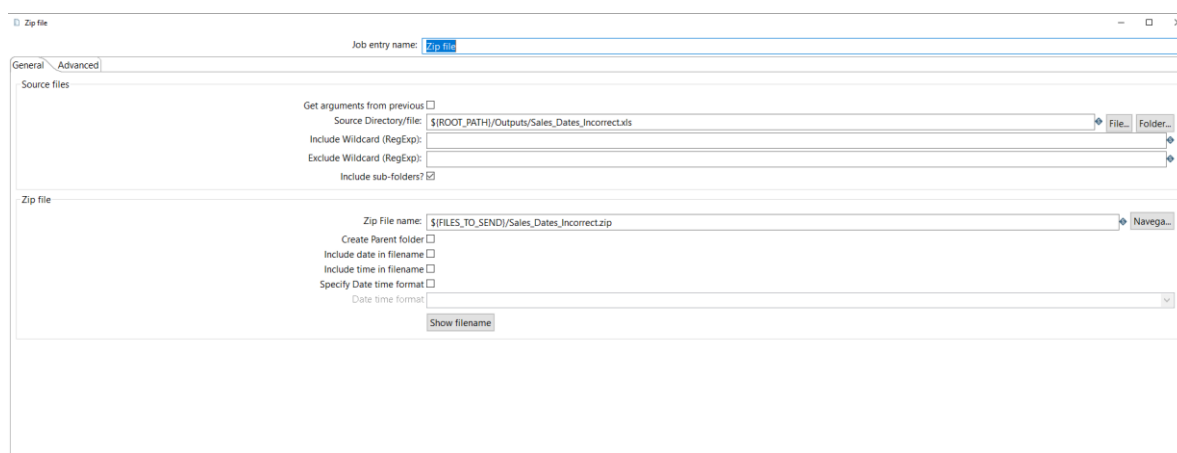


Figura 68: ZIP FILE.

Step 9: Mail (Envio do Ficheiro Zipado por E-mail)

Após o ficheiro ser zipado, o step **Mail** envia o ficheiro comprimido por e-mail, permitindo que os dados incorretos sejam partilhados com os responsáveis para revisão e correção.

- **SMTP Server:** \${server} - Configuração do servidor de e-mail.
- **Port:** \${port} - Porta de comunicação para o servidor de e-mail.
- **Username:** \${username} - Conta de e-mail utilizada para envio.
- **Attachment:** Ficheiro zip localizado no diretório \${FILES_TO_SEND}.

Mail

Name of mail job entry: Mail

Addresses | Server | EMail Message | Attached Files

Destination

Destination address: susanapvl@hotmail.com

Cc:

Bcc:

Sender

Sender name: Pentaho

Sender address: susanapvl@gmail.com

Reply to: susanapvl@gmail.com

Contact person: Pentaho admin

Contact phone:

Help OK Cancela

Figura 69: MAIL: ADDRESSES.

Mail

Name of mail job entry: Mail

Addresses | Server | EMail Message | Attached Files

SMTP Server

SMTP Server: \${server}

Port: \${port}

Authentication

Use authentication? ☒

Authentication user: \${username}

Authentication password:

Use secure authentication? ☒

Secure connection type: SSL

Help OK Cancela

Figura 70: MAIL:SERVER.

Mail

Name of mail job entry: Mail

AddressesServerE-Mail MessageAttached Files

Message Settings

Include date in message?☒

Only send comment in mail body?☒

Use HTML format in mail body?☐

EncodingUTF-8

Manage priority☐

PriorityNormal

ImportanceNormal

SensitivityNormal

Message

Subject:Error file for sales.

Comment:Error in Sales data, please check and rectify.

Help

OK

Cancela

Figura 71: MAIL: EMAIL MESSAGE.

Mail

Name of mail job entry: Mail

AddressesServerE-Mail MessageAttached Files

Files added in result filename

Attach file(s) to message?☒

Select file type:GeneralLogError lineErrorWarning

Zip files to single archive?☒

Name of zip archive:\${FILES_TO_SEND}/Sales_Dates_Incorrect.zip

Embedded images

Filename

Content ID

Add

Browse files ...

Embedded images

#	Image	Content ID
1		

Delete

Edit

Help

OK

Cancela

Figura 72: MAIL: ATTACHED FILES.

P

Pentaho<susanapvl@gmail.com>

Para: Você

Responder

Responder a todos

Encaminhar

Dom, 27/10/2024 13:

Sales_Dates_Incorrect.zip

46 KB

Iniciar a responder com:

Fixed!

I fixed it.

All fixed.

Error in Sales data, please check and rectify.

Message date: 2024/10/27 13:32:22.901

Responder

Encaminhar

Figura 73: EMAIL RECEBIDO COM O ZIP FILE.

Considerações finais

Este **Job** no Pentaho Data Integration orquestra todo o processo de ETL, desde a execução sequencial das transformações de dados até à validação da criação de ficheiros e ao envio de dados incorretos por e-mail. A definição de variáveis no início do job facilita a parametrização e o controlo do processo, assegurando que configurações de caminho e e-mail são acessíveis em toda a execução.

O processo completo inclui uma verificação de existência do ficheiro de erros, `Sales_Dates_Incorrect.xls`, que garante a integridade das operações e permite identificar falhas nos dados em fases críticas. A compressão e envio por e-mail deste ficheiro automatizam a comunicação de erros, permitindo que estes sejam rapidamente identificados e resolvidos.

Em resumo, este **Job** centraliza e automatiza o fluxo de trabalho de processamento de dados, integrando transformações e validações com verificações de ficheiros e notificações, assegurando um processo de ETL robusto e eficiente para a análise de dados de vendas.

Estrutura das Bases de Dados Utilizadas

Esta secção apresenta a estrutura das bases de dados utilizadas no processo de ETL, detalhando os campos e a organização das tabelas essenciais para as transformações. As bases de dados AWS e SQL contêm dados de produtos, clientes e vendas, que são processados e integrados nas transformações descritas.

1. Base de Dados AWS

A base de dados AWS armazena os dados históricos de vendas, acessíveis através de ficheiros CSV e integrados no processo ETL. Esta estrutura permite a recuperação de informações de vendas passadas, essenciais para complementar os dados atuais.

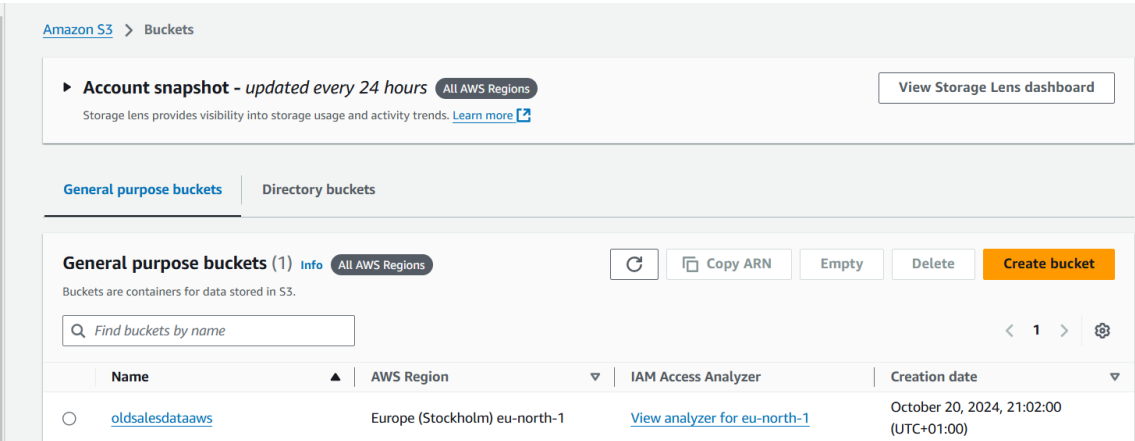


Figura 74: BICKET AWS.

2. Base de Dados SQL

A base de dados SQL armazena os dados atuais de vendas e produtos e é utilizada para consultas e transformações adicionais. O acesso a esta base de dados permite a obtenção de dados em tempo real, que são depois combinados com os dados históricos da AWS.

- **Estrutura das Tabelas:** Os prints abaixo ilustram a tabela **sales** e a tabela **products**, com campos importantes como **Product_ID**, **Product_Name**, **Category**, e **Sales**, entre outros, que são processados na transformação.

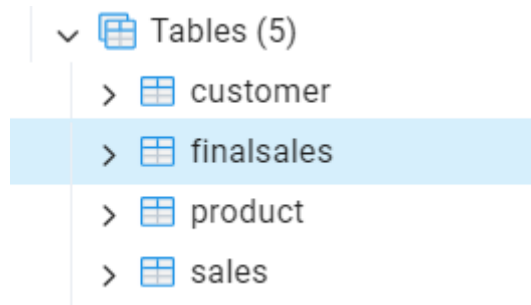


Figura 75: TABELAS BASE DE DADOS POSTGRES SQL.

Data Output Messages Notifications										
	surr_id [PK] integer	customer_id character varying	customer_name character varying	segment character varying	age integer	city character varying	state_name character varying	country character varying	postal_code character varying	region character varying
1	102	BP-11185	Ben Peterman	Corporate	48	Arvada	Colorado	United States	80004	West
2	2	AA-10375	Allen Arnold	Consumer	22	Mesa	Arizona	United States	85204	West
3	4	AA-10645	Anna Andreadi	Consumer	32	Chester	Pennsylvania	United States	19013	East
4	6	AB-10060	Adam Bellavance	Home Office	25	New York City	New York	United States	10009	East
5	7	AB-10105	Adrian Barton	Consumer	63	Phoenix	Arizona	United States	85023	West
6	26	AG-10765	Anthony Garverick	Home Office	40	Philadelphia	Pennsylvania	United States	19120	East
7	27	AG-10900	Arthur Gainer	Consumer	56	Tucson	Arizona	United States	85705	West
8	28	AH-10030	Aaron Hawkins	Corporate	60	Philadelphia	Pennsylvania	United States	19134	East
9	29	AH-10075	Adam Hart	Corporate	21	New York City	New York	United States	10011	East
10	30	AH-10120	Adrian Hane	Home Office	27	Tucson	Arizona	United States	85705	West
11	31	AH-10195	Alan Haines	Corporate	67	Tamarac	Florida	United States	33319	South
12	32	AH-10210	Alan Hwang	Consumer	58	Brentwood	California	United States	94513	West
13	33	AH-10465	Amy Hunt	Consumer	24	New York City	New York	United States	10035	East
14	34	AH-10585	Angele Hood	Consumer	34	Chicago	Illinois	United States	60623	Central
15	35	AH-10690	Anna Haberlin	Corporate	39	New York City	New York	United States	10024	East
16	36	AI-10855	Arianne Irving	Consumer	35	Philadelphia	Pennsylvania	United States	19120	East
17	37	AJ-10780	Anthony Jacobs	Corporate	47	Springfield	Virginia	United States	22153	South
18	38	AJ-10795	Anthony Johnson	Corporate	27	Saint Petersburg	Florida	United States	33710	South
19	39	AJ-10945	Ashley Jarboe	Consumer	62	Wilmington	North Carolina	United States	28403	South
20	40	AJ-10960	Astrea Jones	Consumer	30	Rochester	New York	United States	14609	East
21	41	AM-10360	Alice McCarthy	Corporate	45	Grand Prairie	Texas	United States	75051	Central
22	42	AM-10705	Anne McFarland	Consumer	28	Auburn	Alabama	United States	36830	South
Total rows: 793 of 793 Query complete 00:00:00.211 Ln 1, Col 24										

Figura 76: TABELA CUSTOMER.

Data Output

Messages

Notifications

Figura 77: TABELA PRODUCT.

	order_line [PK] integer	order_id character varying	order_date date	ship_date date	ship_mode character varying	customer_id character varying	product_id character varying	sales numeric	quantity integer	discount numeric	profit numeric
1	7973	CA-2017-166142	2020-01-01	2020-01-05	Standard Class	MM-17260	OFF-BI-10004094	26.55	3	0	13.01
2	7974	CA-2017-166142	2020-01-01	2020-01-05	Standard Class	MM-17260	FUR-TA-10004607	310.44	3	0.3	-48.78
3	7975	CA-2017-164378	2020-01-01	2020-01-04	Second Class	MM-18055	OFF-AR-10001177	6.56	2	0	1.9
4	7976	CA-2017-164378	2020-01-01	2020-01-04	Second Class	MM-18055	OFF-LA-10000634	7.83	3	0	3.6
5	7977	CA-2017-164378	2020-01-01	2020-01-04	Second Class	MM-18055	TEC-AC-10004708	41.9	2	0	8.8
6	7978	CA-2017-164378	2020-01-01	2020-01-04	Second Class	MM-18055	FUR-CH-10002084	664.15	6	0.1	88.55
7	7979	CA-2017-164378	2020-01-01	2020-01-04	Second Class	MM-18055	OFF-PA-10004519	8.96	2	0	4.39
8	7980	US-2017-128951	2020-01-01	2020-01-03	First Class	RS-19420	OFF-AP-10002191	179.94	3	0	50.38
9	7981	US-2017-128951	2020-01-01	2020-01-03	First Class	RS-19420	FUR-TA-10004575	872.94	3	0	157.13
10	7982	US-2017-128951	2020-01-01	2020-01-03	First Class	RS-19420	OFF-PA-10003177	12.96	2	0	6.22
11	7983	US-2017-156909	2020-01-02	2020-01-04	Second Class	SF-20065	FUR-CH-10002774	71.37	2	0.3	-1.02
12	7984	CA-2017-109778	2020-01-02	2020-01-07	Standard Class	VM-21685	OFF-AR-10003759	2.91	2	0.2	0.91
13	7985	US-2017-152842	2020-01-02	2020-01-09	Standard Class	NF-18385	FUR-CH-10004218	242.35	3	0.2	15.15
14	7986	CA-2017-139948	2020-01-03	2020-01-08	Standard Class	SW-20455	FUR-FU-10002597	7.9	2	0.2	2.17
15	7987	US-2017-105046	2020-01-03	2020-01-09	Standard Class	BE-11335	TEC-PH-10004536	269.98	2	0	67.5
16	7988	US-2017-105046	2020-01-03	2020-01-09	Standard Class	BE-11335	OFF-PA-10004353	99.9	5	0	47.95
17	7989	US-2017-105046	2020-01-03	2020-01-09	Standard Class	BE-11335	FUR-FU-10004848	39.08	4	0	14.46
18	7990	CA-2017-126662	2020-01-03	2020-01-07	Standard Class	AB-10255	TEC-CO-10004202	479.98	2	0.2	90
19	7991	CA-2017-142342	2020-01-03	2020-01-05	Second Class	AJ-10795	OFF-PA-10004609	32.4	5	0	15.55
20	7992	CA-2017-142342	2020-01-03	2020-01-05	Second Class	AJ-10795	OFF-EN-10002592	57.9	5	0	28.95
21	7993	CA-2017-142342	2020-01-03	2020-01-05	Second Class	AJ-10795	OFF-ST-10002957	10.56	2	0	0
22	7994	CA-2017-142342	2020-01-03	2020-01-05	Second Class	AJ-10795	FUR-BO-10002613	1194.17	5	0.15	210.74

Figura 78: TABELA SALES.

Vídeo demonstrativo

[InShot 20241027 194934942.mp4](#)

CONCLUSÃO

O desenvolvimento deste projeto de ETL com recurso ao Pentaho Data Integration foi essencial para estruturar, transformar e consolidar dados provenientes de diferentes fontes, como bases de dados SQL e AWS, e ficheiros CSV. A execução sequencial das transformações Customer Transformation, Product Transformation e Sales Transformation permitiu criar um fluxo de dados robusto e integrado, proporcionando uma visão global dos dados de vendas e produtos ao longo do tempo.

Cada transformação foi cuidadosamente configurada para extrair, organizar e validar dados críticos para o negócio. A utilização de steps de validação e verificação de integridade garantiu a consistência e qualidade dos dados. Adicionalmente, o Job criado para orquestrar todas as transformações trouxe um nível adicional de automação, assegurando que as tarefas são realizadas numa sequência lógica e eficiente. A inclusão de variáveis no job centralizou configurações importantes, como caminhos de diretório e credenciais, aumentando a flexibilidade e reutilização do processo.

Um dos pontos-chave deste projeto foi a criação de um processo que, além de transformar os dados, também verifica erros críticos, como inconsistências em datas de envio, e alerta os responsáveis por meio de um sistema de notificação automático. O envio de relatórios de erro por e-mail, com anexos comprimidos dos dados incorretos, permite uma resposta rápida para a resolução de problemas, aumentando a fiabilidade dos dados para as análises.

Em síntese, o trabalho desenvolvido demonstra a importância de um processo de ETL bem estruturado para a integração e validação de dados. Este sistema oferece uma base sólida e consistente para a análise de vendas e suporte à tomada de decisões estratégicas, com a capacidade de lidar com grandes volumes de dados de diferentes fontes e assegurar a qualidade dos mesmos. A abordagem utilizada fornece uma fundação escalável e automatizada que pode ser expandida para suportar novos requisitos e integrações de dados no futuro, tornando-se um ativo valioso para a gestão da informação na organização.