Licenciatura em Engenharia de Sistemas Informáticos

Trabalho prático ETLs

Integração de Sistemas de Informação

Flávio Costa A20349 & João Pereira A20345
10-19-2024

Table of Contents

Introdução

No âmbito da Licenciatura em Engenharia de Sistemas Informáticos, da Unidade Curricular de Integração de Sistemas de Informação, lecionada no 1º semestre do 3º ano do curso em questão, no Instituto Politécnico do Cávado e Ave, foi proposto pelo docente Luís Ferreira a realização de um trabalho individual ou em grupo, grupo este composto por até 2 elementos, com os seguintes objetivos:

- Compreender a Integração de Sistemas de Informação com foco em dados;
- Identificar e descrever cenários de aplicação para processos Extract, Transform, Load (ETL);
- Explorar ferramentas que auxiliam na execução de processos ETL;
- Investigar tecnologias, frameworks e paradigmas emergentes na área;
- Aprimorar habilidades no desenvolvimento de software relacionado à Integração de Sistemas de Informação;
- Facilitar a compreensão e assimilação da unidade curricular.

ETL foi idealizado pois existe a necessidade de integrar dados de diferentes fontes, aplicar transformações relevantes a esses dados, e carregá-los em um formato apropriado para a criação de relatórios significativos. Isso envolve:

- Extract: Identificar as fontes de dados necessárias e extrair esses dados de várias origens, como bases de dados e ficheiros;
- Transform: Realizar transformações nos dados extraídos para garantir a consistência e qualidade, como limpeza, normalização e agregação.
- Load: Carregar os dados transformados em bases de dados ou ficheiros apropriados, garantindo que estejam disponíveis para a criação de relatórios.

O problema, portanto, está na eficiente implementação do ciclo ETL para atender às necessidades específicas, garantindo que os dados estejam corretos, atualizados e
prontos para análise e geração de insights significativos.

Problema

A gestão eficaz das informações em uma loja social é essencial para oferecer serviços de qualidade e atender adequadamente às necessidades da comunidade. No entanto, muitas dessas lojas enfrentam desafios significativos na coleta, integração e disponibilização de dados precisos e atualizados sobre seus beneficiários e produtos. Esses desafios incluem:

1.1. Fragmentação dos Sistemas de Informação

Muitas lojas sociais utilizam diferentes sistemas para gerenciar diversos aspectos, como o cadastro de beneficiários e o controle de estoque. Essa fragmentação resulta em dados dispersos, dificultando a obtenção de uma visão completa sobre as necessidades e o histórico dos beneficiários.

1.2. Informações Desatualizadas e Inconsistentes

A falta de sincronização entre os sistemas e a entrada manual de dados frequentemente levam a registros desatualizados e inconsistentes. Essa falta de precisão pode ocasionar atendimentos inadequados, dificultando a identificação das necessidades dos beneficiários e o acompanhamento dos produtos disponíveis.

1.3. Necessidade de Análises e Relatórios Eficazes

Além do atendimento direto aos beneficiários, as lojas sociais também precisam analisar dados para aprimorar processos internos, alocar recursos de maneira eficaz e garantir a conformidade com as diretrizes regulamentares. A ausência de um sistema de ETL robusto dificulta a obtenção de insights valiosos a partir dos dados coletados.

Diante desses desafios, é claro que a implementação de uma solução de ETL eficaz é fundamental para o funcionamento de uma loja social.

ETLs utilizados

O KNIME (Konstanz Information Miner) é uma ferramenta de código aberto amplamente utilizada para a integração e análise de dados. Faz parte de um ecossistema mais vasto de soluções em ciência de dados e oferece uma interface intuitiva que facilita a execução de processos de ETL (Extração, Transformação e Carga) de forma eficaz.

Algumas das principais características e funcionalidades do KNIME incluem:

- **Manipulação de grandes volumes de dados:** Capacidade de trabalhar com conjuntos de dados extensos provenientes de várias fontes.
- **Transformação de dados:** Permite aplicar diversas técnicas de transformação para preparar os dados para análise.
- **Limpeza de dados:** Ferramentas que ajudam a identificar e corrigir erros ou inconsistências nos dados.
- **Integração com outras ferramentas:** Facilita conexões com diversas plataformas de análise e visualização, permitindo a criação de fluxos de trabalho completos.

O KNIME é uma escolha popular entre organizações que desejam automatizar e otimizar os seus processos de ETL. A interface gráfica, baseada em nós, possibilita que os utilizadores construam fluxos de trabalho complexos de maneira visual, arrastando e soltando componentes. Isso torna o desenvolvimento de processos de integração e análise de dados acessível tanto a especialistas quanto a iniciantes na área da ciência de dados.

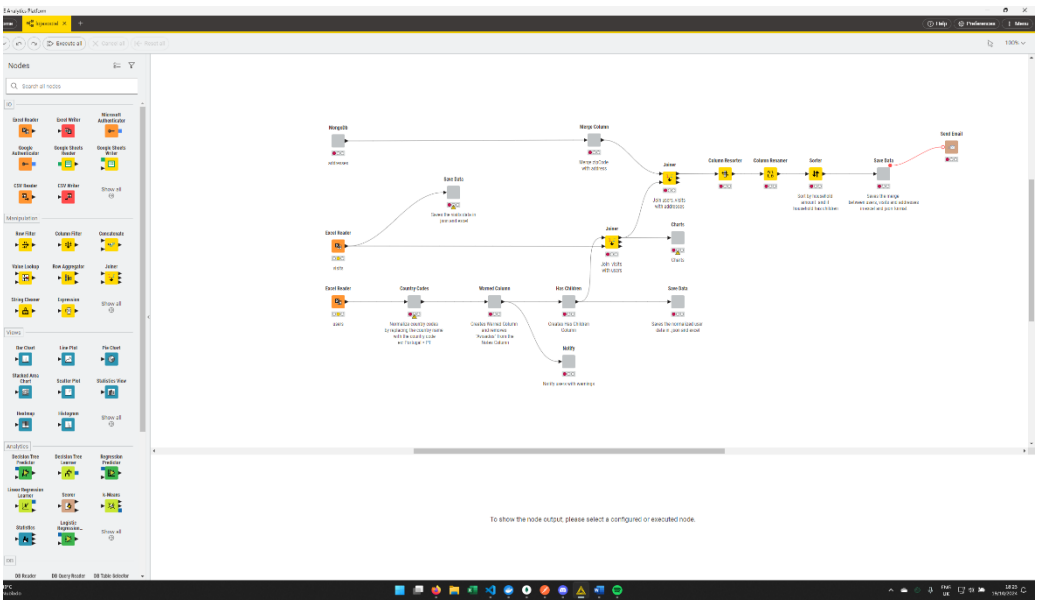


Figura 1 - Knime interface

Pentaho Data Integration (também conhecido como Kettle) é um software que pertence ao conjunto de ferramentas do Pentaho, este software é o responsável pelos processos de Extração, Transformação e Carregamento de dados – mais conhecidos como processos ETL.

O PDI não serve apenas como uma ferramenta ETL, mas também é usado para outros fins, como **migração de dados entre aplicações ou bases de dados**, **exportação de dados de bases de dados para ficheiros simples, limpeza de dados** e **Integração com outras ferramentas**, tal como o KNIME**.** O PDI possui um ambiente de design intuitivo, gráfico e de *drag and drop*, e seus recursos de ETL são poderosos.

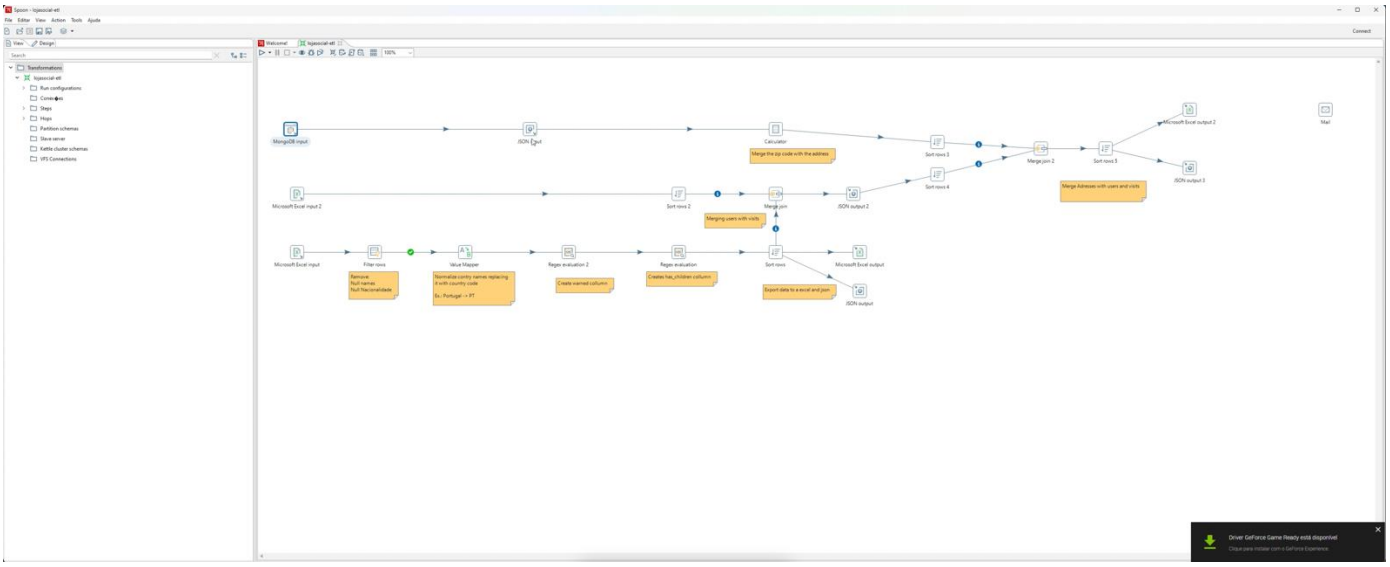


Figura 2 - Penthano Data Integration (Kettle)

Dataset

O dataset foi fornecido pela loja social em formato Excel, sendo posteriormente convertido para CSV. Após essa conversão, foram realizadas modificações significativas nos dados originais para assegurar a privacidade dos usuários. Para isso, utilizou-se a biblioteca Faker.js, que gerou dados falsos e realistas, substituindo as informações contidas no dataset.

Foi também gerado um novo dataset de moradas que associava uma morada a cada beneficiario presente no outro dataset.

Este processo garantiu que as identidades dos beneficiários permanecessem protegidas, permitindo que a análise dos dados fosse realizada de forma ética e responsável.

Original - 1," Santos","...88...","colega","3","Brasileira ",,"menina 11 anos"

Novo - 1,Gretchen Wolf,+15599156771,Mildred Morar,3,"Brasileira ",,menina 11 anos

Dados de utilizador e moradas falsas

A biblioteca Faker.js foi usada para gerar moradas falsas de forma automática. Esta biblioteca oferece a capacidade de criar diferentes tipos de dados fictícios, como nomes, países, números de telemóvel, e endereços completos. Através dela, foi gerada uma morada para cada utilizador presente no sistema, garantindo uma diversidade de dados realistas para fins de teste e simulação.

As moradas geradas foram então armazenadas numa base de dados MongoDB, que está configurada para rodar localmente através de um container Docker. O uso de Docker facilita a criação e o gerenciamento do ambiente de execução, permitindo que a base de dados seja executada de forma isolada e controlada, otimizando o processo e garantindo maior flexibilidade no desenvolvimento.

```
generateFakeAddresses: (size: number): FakerUserAddress[] => {
    const data: FakerUserAddress[] = [];

    for (let i = 1; i < size; i++) {
        const { zipCode, streetAddress } = faker.location;
        let address = streetAddress(true).replace(/^\d+\s*/, "");

        data.push({
            userId: i,
            address,
            zipCode: zipCode({
                format: "##### - ###",
            }),
        });
    }

    return data;
},
```

Figura 4 – Fakerjs para utilizadores

```
generateFakeUsers: (data: CsvFeilds[]): CsvFeilds[] => {
    const fakeUsers = data.map((user: any) => {
        return {
            ...user,
            ID: user.ID,
            Nome: faker.person.fullName(),
            "Telemovel": faker.phone.number({
                style: "international",
            }),
            "Referencia": faker.person.fullName(),
        };
    });

    return fakeUsers as CsvFeilds[];
}
```

Figura 3 – Fakerjs para moradas

Leitura dos dados

Um dos primeiros passos fundamentais na execução do processo ETL (Extract, Transform, Load) criado foi a leitura de um novo ficheiro contendo dados falsos em formato CSV. Este arquivo serviu como uma das fontes de dados iniciais para o processo, permitindo que as informações fossem extraídas e posteriormente tratadas.

Logo após essa etapa, foi realizada a leitura de outro ficheiro, este contendo informações relacionadas às datas e visitas de cada utilizador do sistema, dados essenciais para realizar análises e compreender o comportamento dos usuários ao longo do tempo.

Uma vez que os dados iniciais foram carregados, o próximo passo foi estabelecer uma ligação com a base de dados MongoDB. Através dessa conexão, o sistema foi capaz de acessar as informações armazenadas na coleção “addresses”, especificamente as moradas de cada utilizador.

A leitura destas três fontes – o ficheiro com dados falsos, o ficheiro de datas e visitas, e a base de dados MongoDB – formou a base para o restante das etapas do processo ETL.

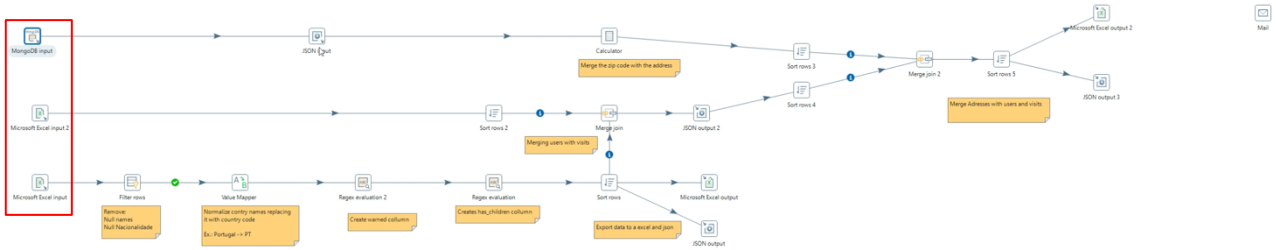


Figura 6 - Kettle nodes de leitura

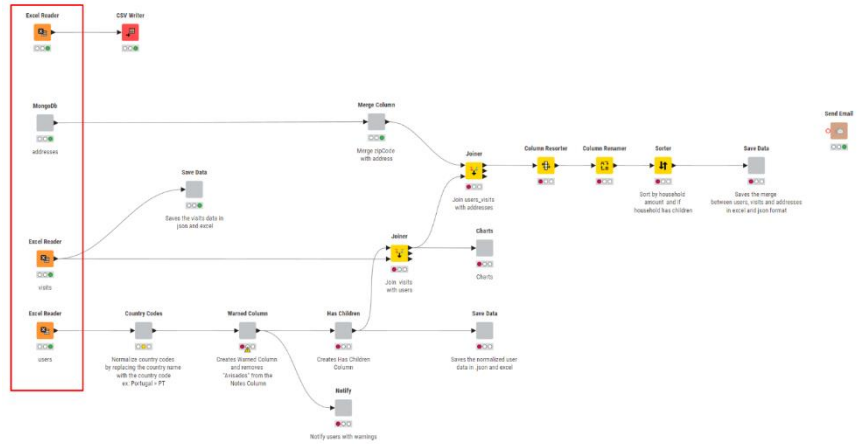


Figura 5 - Knime nodes de leitura

Nacionalidades

O dataset fornecido apresentava várias variações na forma de escrever o nome de um mesmo país, como "Portugugal", "português" ou "portuguêsa", o que causava inconsistências nos dados. Para solucionar esse problema, foi utilizado o Node String Manipulation com a função regexReplace. Essa abordagem permitiu identificar e substituir essas variações textuais de maneira eficiente.

Com a aplicação da função regexReplace, todas as nacionalidades foram normalizadas para o formato de código de país com duas letras.

Por exemplo, variações como "português" ou "portuguêsa" foram convertidas para "PT", criando assim uma padronização nos dados e facilitando o processamento. Devido a quantidade de nacionalidades diferentes, foram utilizados multiplos nodes para a nromalização do daset por inteiro.

Figura 7 - Sequencia de nodes (normalização da nacionalidade)

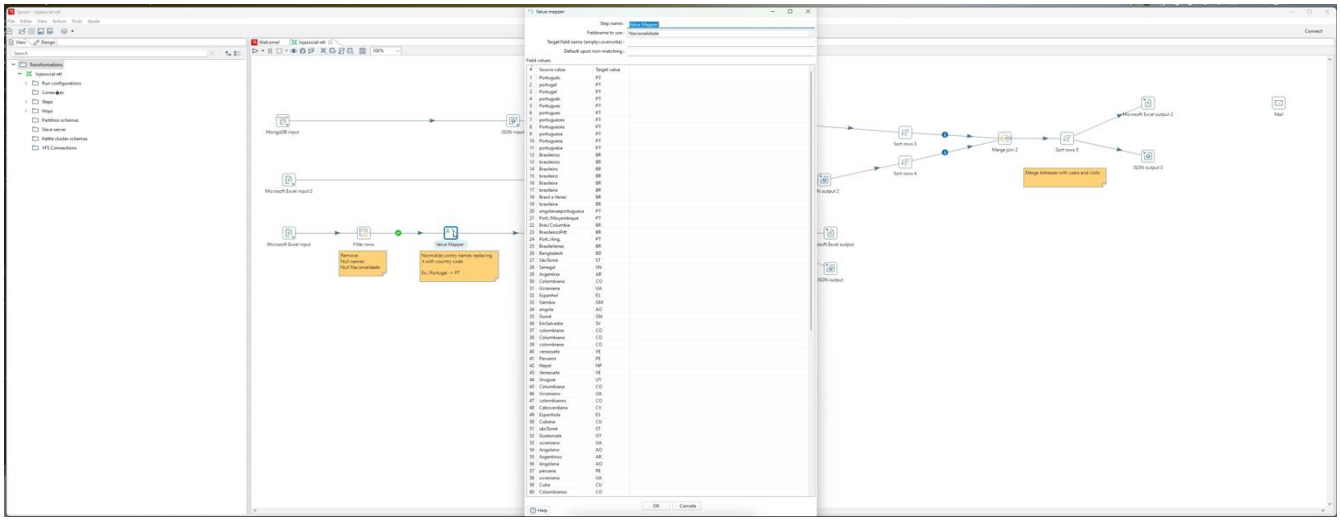


Figura 8 - Normalização de nacionalidade

Coluna Avisados

Este conjunto de dados apresenta uma coluna “notas”, onde, em alguns casos, o valor “Avisados” está presente, enquanto outros utilizadores possuem frases que não estão relacionadas. Para abordar esta questão, foi decidido implementar uma nova coluna chamada “hasWarning”, que armazena um valor booleano. Para a criação desta nova coluna, utilizei o node Rule Engine, que permitiu a adição da nova coluna, e o node String Manipulation, que foi utilizado para remover as ocorrências de “Avisados” da coluna notas. Esta abordagem assegura uma melhor organização e análise dos dados.

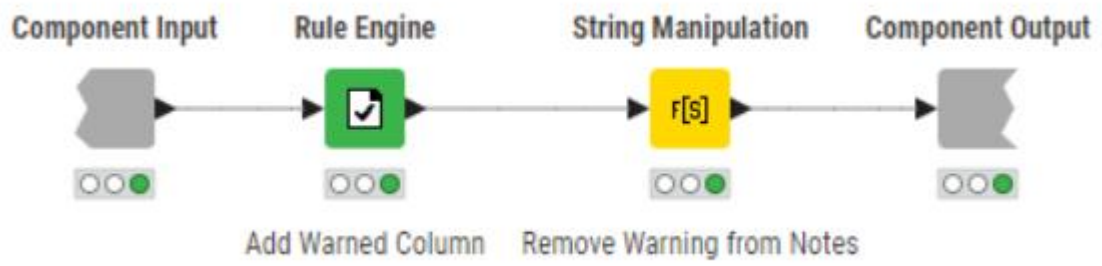


Figura 9 - Sequência de nodes (Coluna hasWarning)

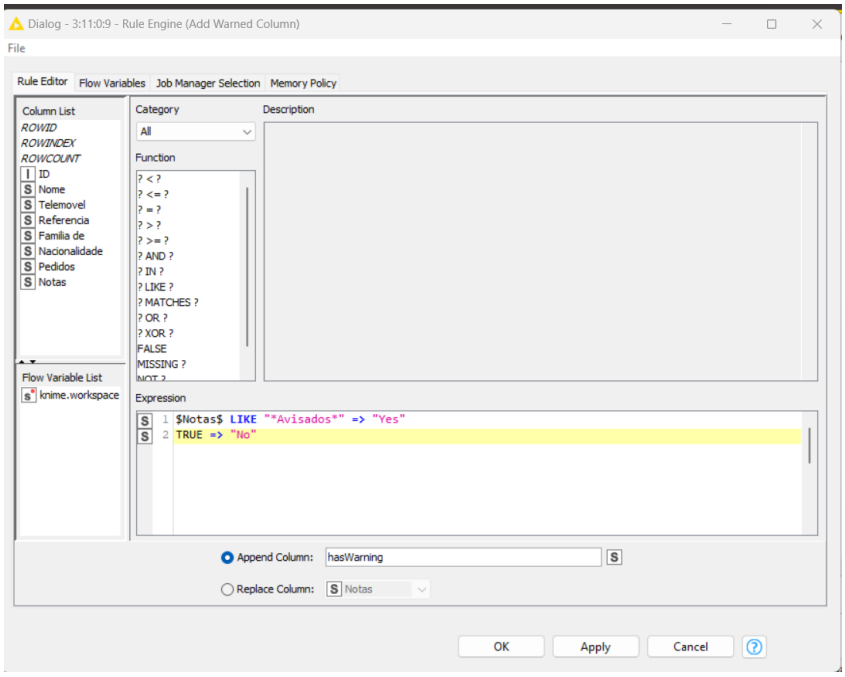


Figura 10 - Node Rule Engine

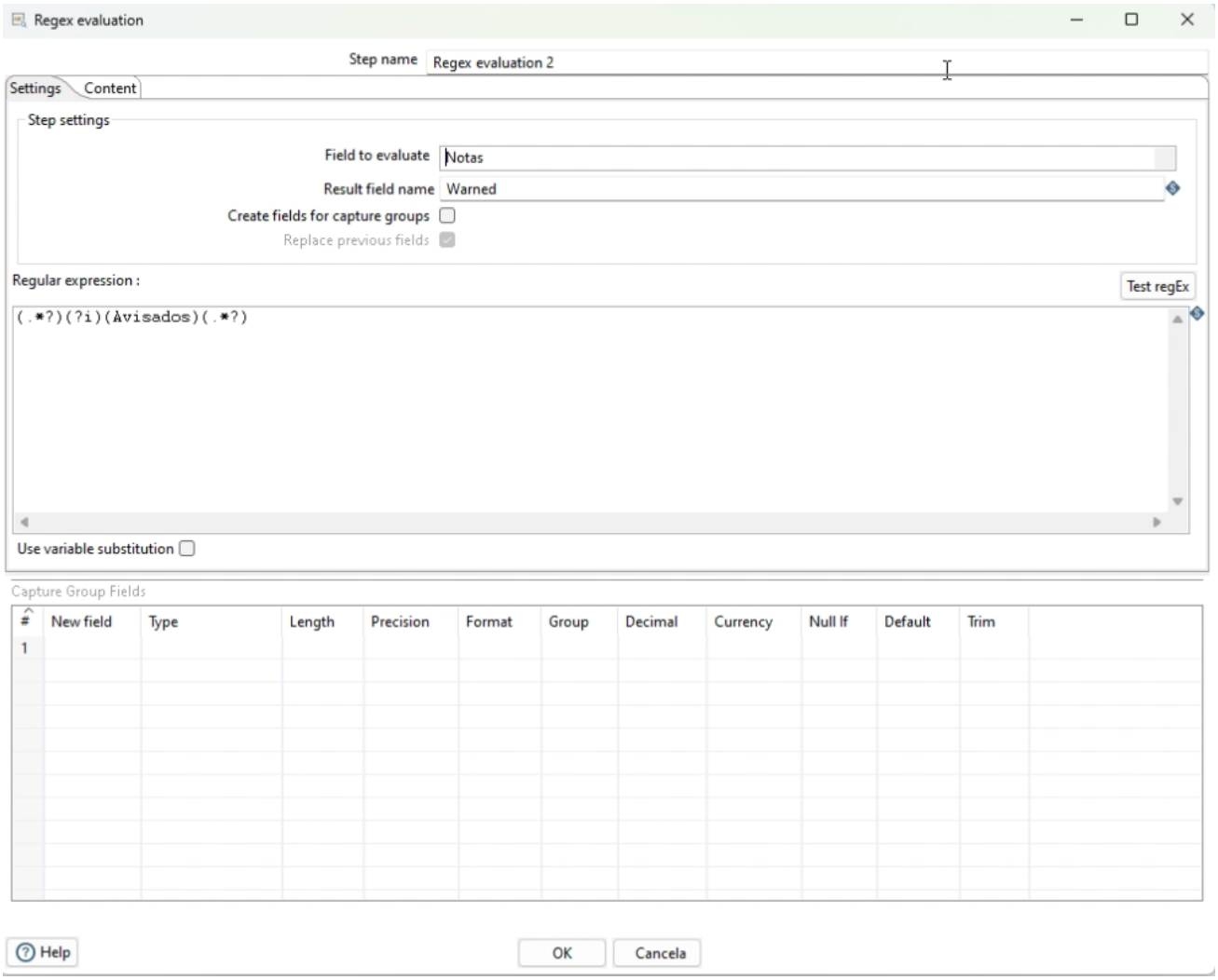


Figura 11 – Node rule engine

Notificar utilizadores com avisos

Após a criação da nova coluna, realizei um processo de filtragem para identificar todos os utilizadores que apresentam o valor verdadeiro. Para isso, utilizei o node denominado "Row Filter", que permitiu isolar os dados relevantes de maneira eficiente.

Com os dados filtrados, procedi à transformação dessas informações para o formato JSON, que é utilizado para comunicação entre aplicações. Essa conversão é ncessária, pois assegura que os dados sejam formatados de forma adequada para o envio.

Por fim, executei um pedido POST à minha API, anexando os dados no body do request. Através deste passo garantimos que as informações sejam processadas corretamente pela API.

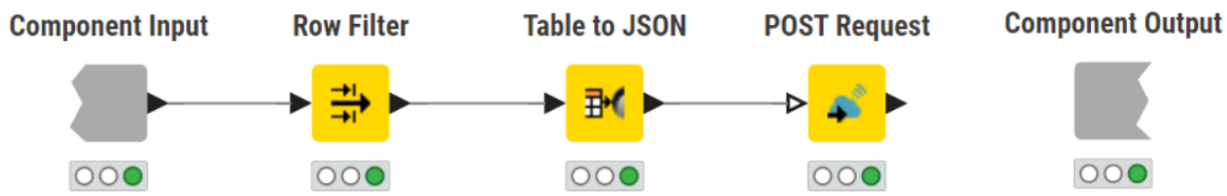


Figura 12 / Sequencia notificar utilizadores avisados

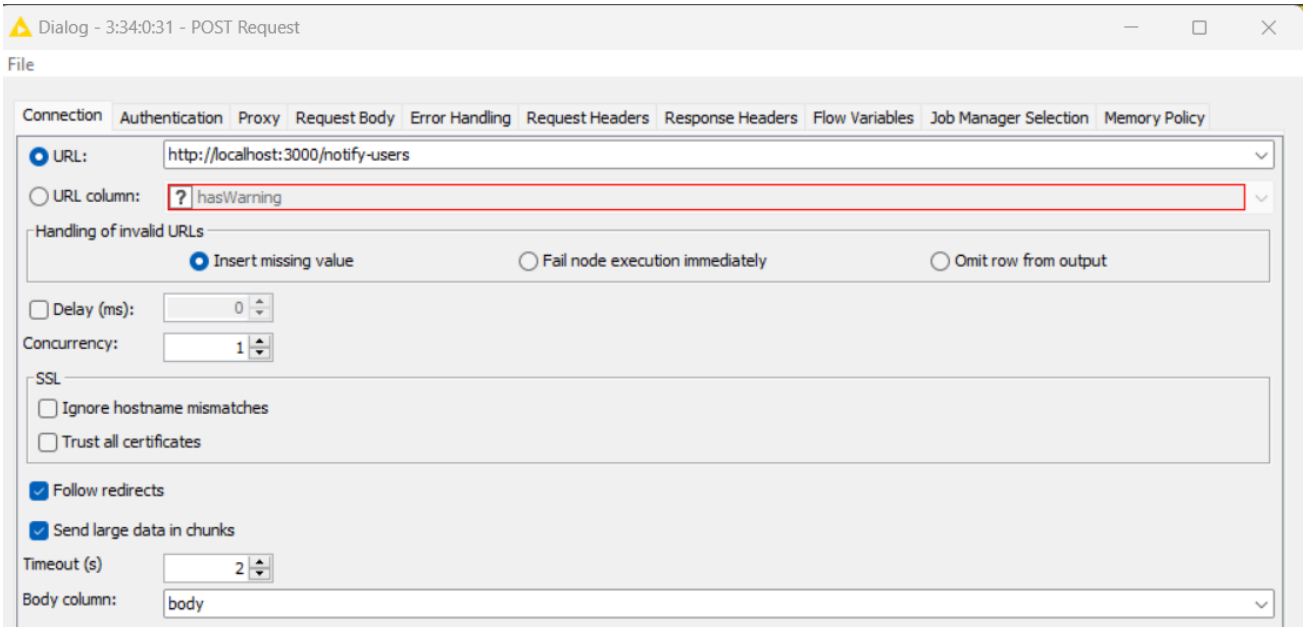


Figura 13 - Node Post request

Coluna com crianças

No conjunto de dados, nas colunas "Pedidos" e "Notas", encontramos palavras como "menino", "menina", "rapaz", "boy", "girl", entre outras variantes. Para identificar a presença dessas palavras, foi criada uma nova coluna chamada **hasChildren**. Nela, aplicamos uma expressão regular (regex) que verifica a ocorrência dos termos mencionados em ambas as colunas.

O regex utilizado foi: (.*?)(?i)(girls|boys|menino|menina...)(.*?)(?i)

Esta expressão funciona da seguinte maneira: o parâmetro (.*?) captura qualquer sequência de caracteres antes e depois das palavras-alvo, enquanto (?i) assegura que a busca seja insensível a maiúsculas e minúsculas.

Posteriormente esta coluna irá ser usada para dar prioridade as familias.

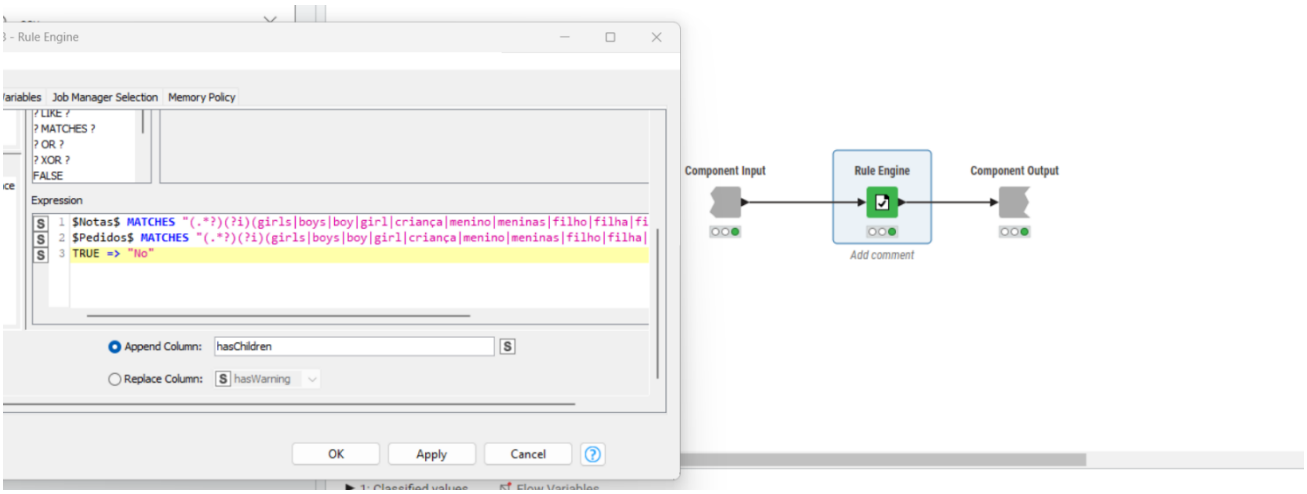


Figura 14 - Node Rule Engine, criação da coluna hasChildren

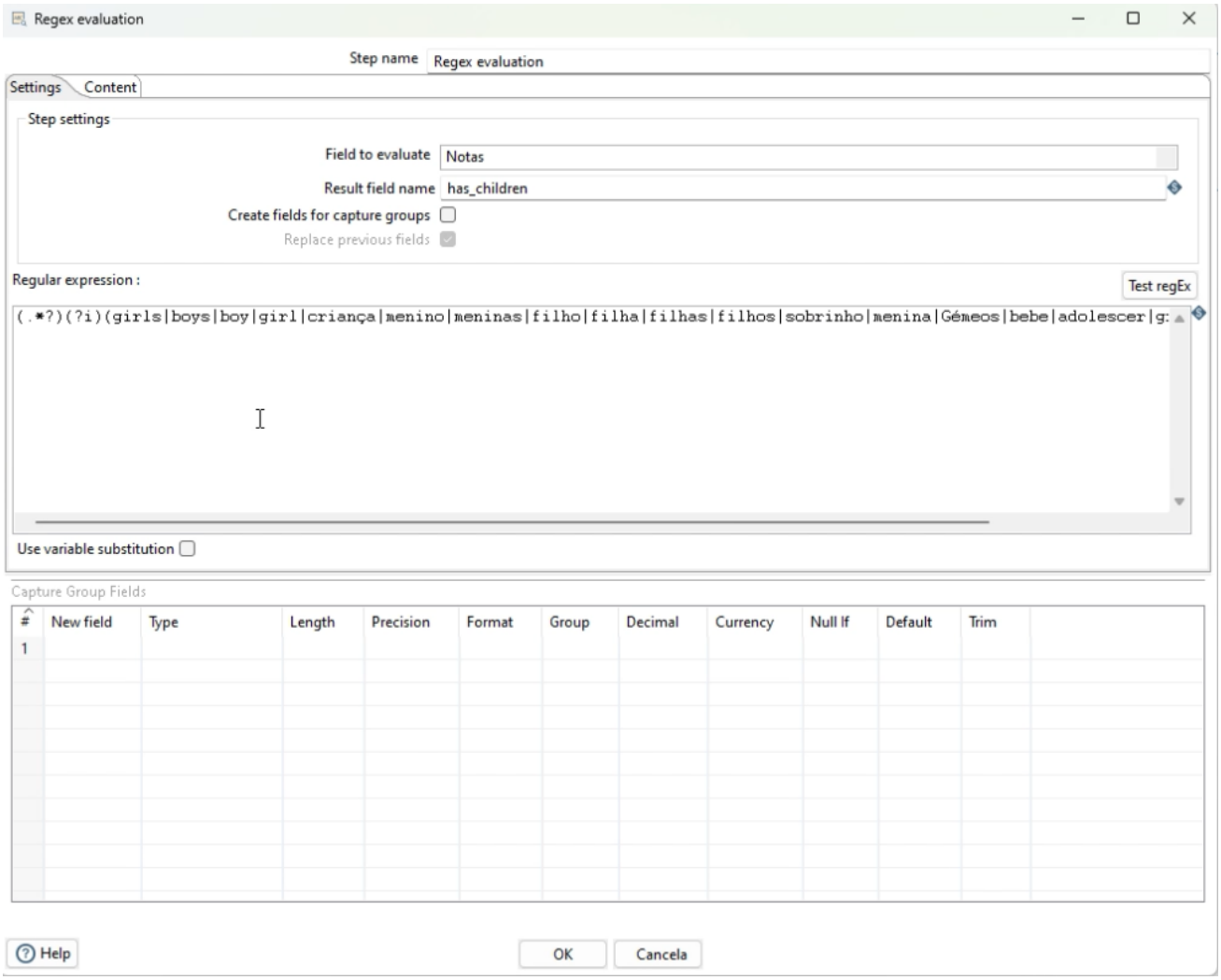


Figura 15 – Node rule engine, criação da coluna has_children

Junção de dados

Para facilitar a visualização das visitas de cada beneficiário, unimos a tabela de beneficiários à tabela de visitas por meio do node joiner, utilizando o campo ID como ponto de ligação. Essa integração permite uma análise mais eficiente da frequência de visitas, que foi utilizada para gerar relatórios sobre os dados disponíveis.

Além disso, a tabela de moradas também foi unida através do node joiner, possibilitando o acesso a todas as informações a partir de uma única tabela. Essa abordagem simplifica a gestão dos dados.

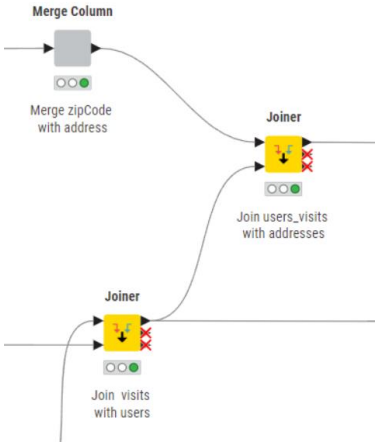


Figura 16 - Nodes joiners

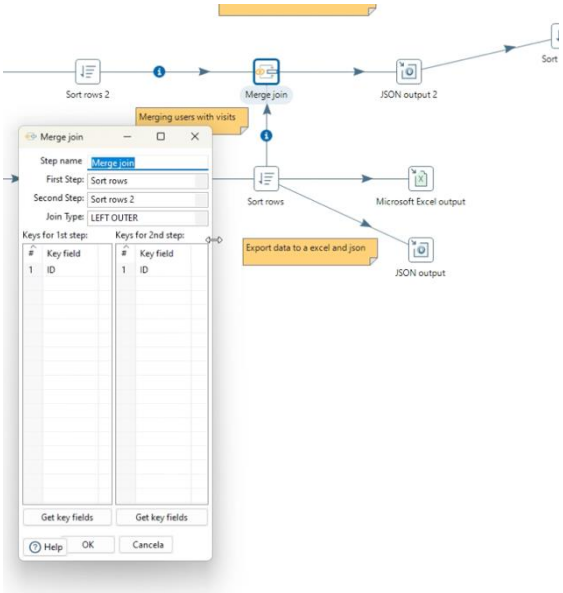


Figura 17 - Nodes joiners

Ordenação por prioridade

Com a adição da coluna "hasChildren" e o uso da coluna "household_amount", podemos priorizar as famílias mais numerosas que têm crianças.

A ordenação dos beneficiários foi realizada com base no número total de membros e na presença de crianças, ajudando a identificar aquelas que realmente precisam de suporte.

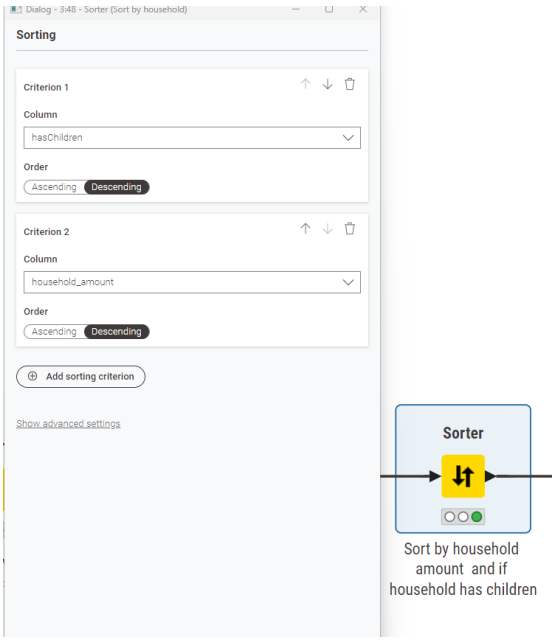


Figura 18 - ordenação por prioridade

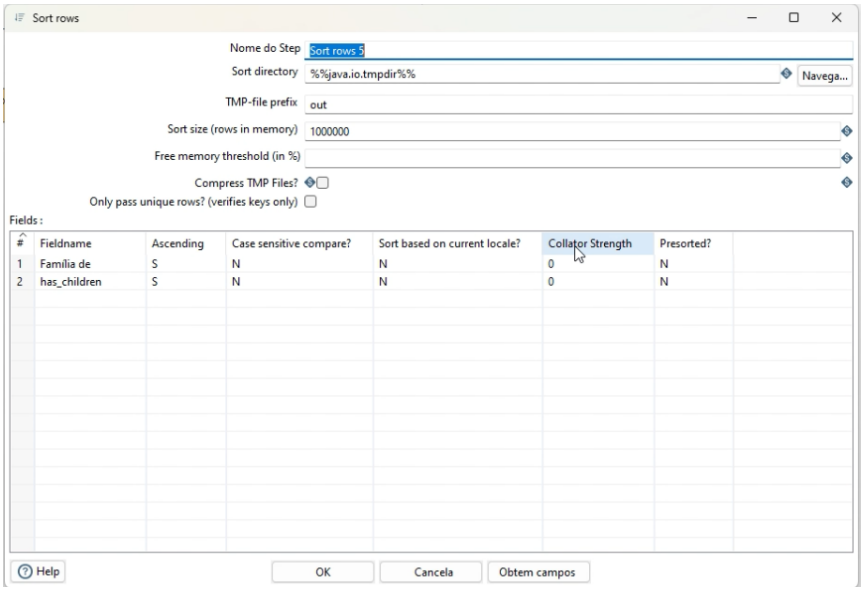


Figura 19 - Ordenação por prioridade

Dashboard

Hoje em dia, a extração de dados é importante para apoiar decisões informadas. Para esse fim, foi criado um dashboard com alguns gráficos relevantes. O primeiro gráfico apresenta o número de visitas por data, permitindo uma visão clara da evolução do fluxo de visitas ao longo do tempo. Depois, temos um gráfico circular que mostra a percentagem de famílias com diferentes números de pessoas, bem como a proporção de famílias com crianças. Finalmente, um gráfico exibe o tamanho das famílias por país.

Para a construção destes gráficos, foram utilizados os nodes disponíveis: o gráfico de barras para o número de visitas por data, o gráfico circular para a distribuição das famílias e o heatmap para ilustrar o tamanho das famílias por país. Estas visualizações permitem uma análise mais fácil dos dados.

No caso do Kettle não é possível fazer dashboards através da aplicação.

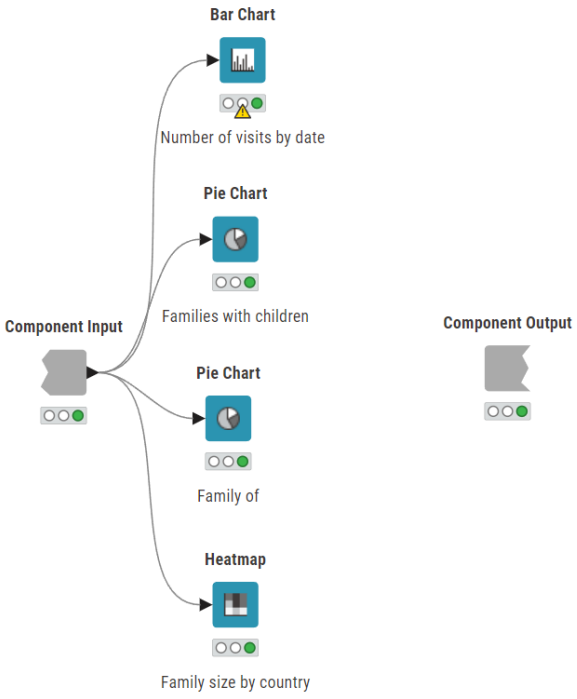


Figura 20 – Nodes gráficos

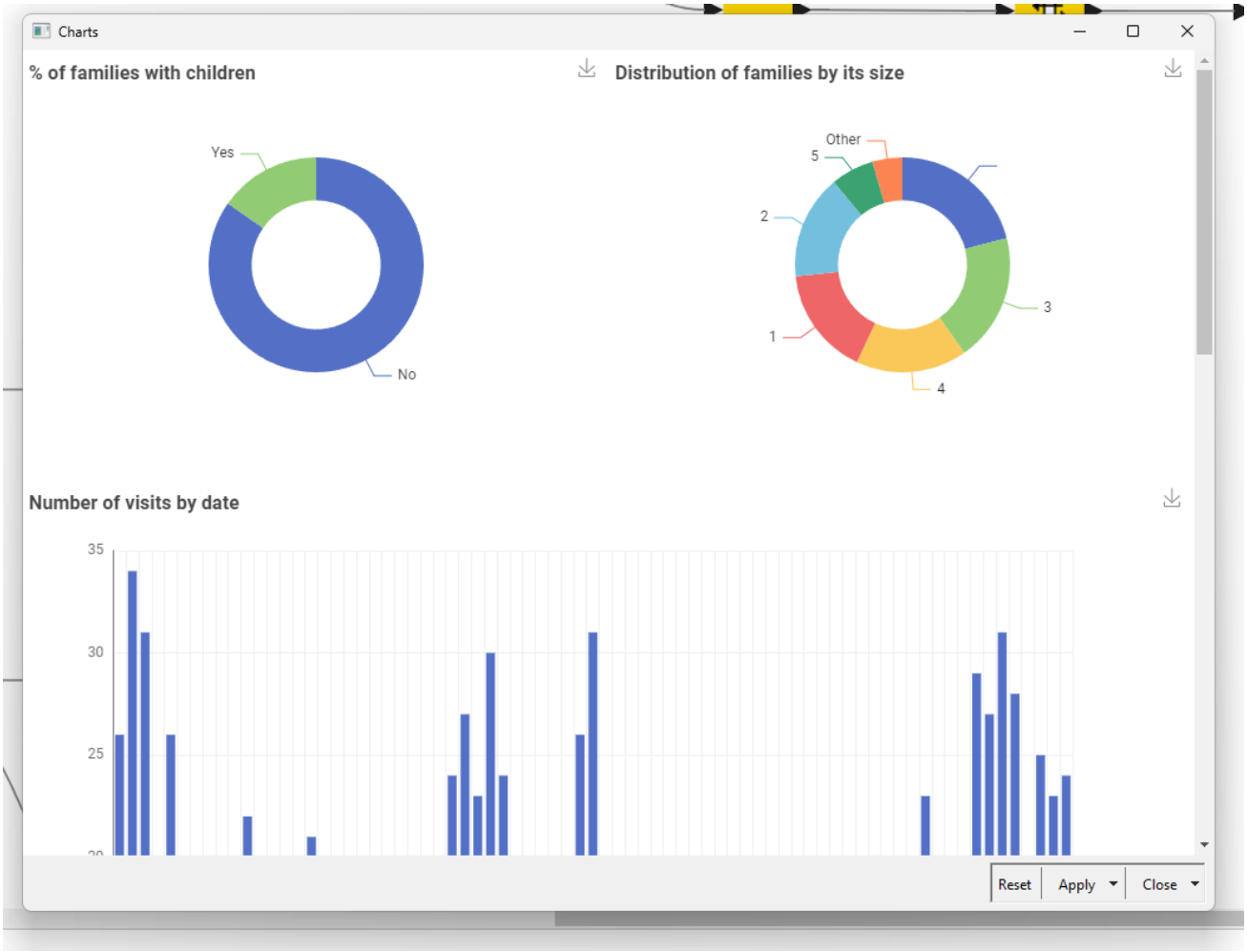
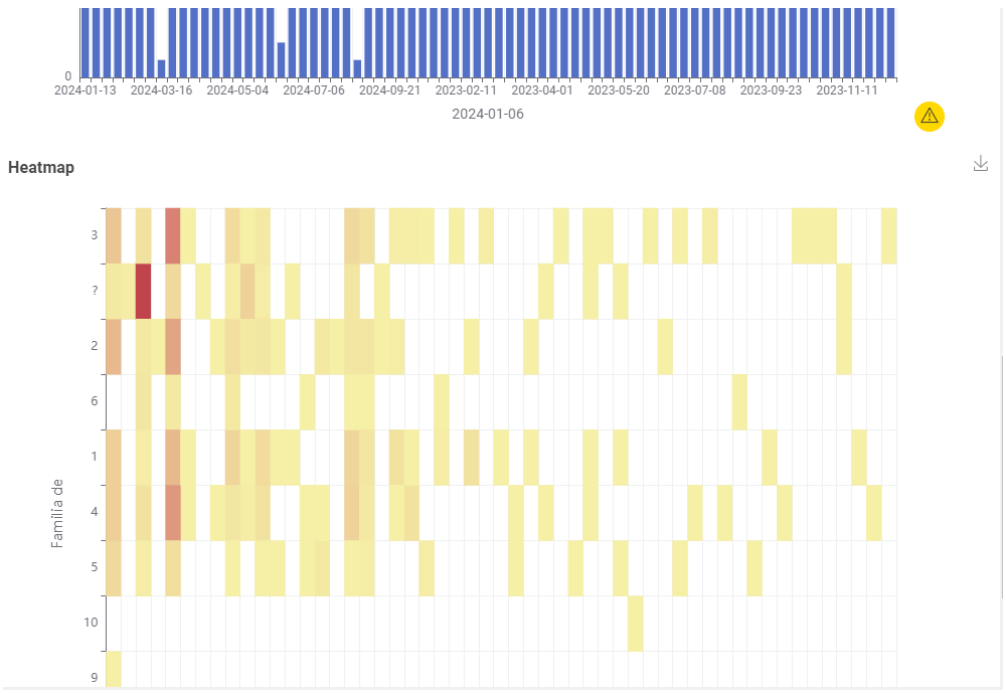


Figura 21 – Dashboard



QR codes with vídeos

Figura 22 - QRCODE video Kettle Penthano

Conclusão

A disciplina de Integração de Sistemas de Informação é essencial para entendermos os princípios fundamentais que regem o funcionamento dos processos de ETL. Neste trabalho, examinamos os conceitos e os elementos-chave envolvidos na gestão e transformação de dados, abrangendo desde a extração de diversas fontes até a organização e disponibilização dos dados.

Além disso, identificamos os desafios frequentes associados à implementação de ETL, como a qualidade e a manutenção dos dados. É importante destacar que o êxito na integração de sistemas de informação vai além da mera execução técnica; envolve também a definição de políticas e práticas que assegurem a confiabilidade e a segurança dos dados.

Em síntese, a integração de sistemas de informação através dos processos de ETL é um aspeto crucial na gestão de dados e na obtenção de insights valiosos. Este trabalho contribuiu para aprofundar entendimento sobre o tema, ressaltando a importância de enfrentar os desafios com a escolha adequada de abordagens e ferramentas. À medida que avançamos, é vital continuar a investigar novas tendências e tecnologias, aprimorando nossas competências para atender às crescentes demandas do setor.