UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

Edson Fagner da Silva Cristovam

Processo de ETL Voltado para Jurimetria

Recife

2021

Edson Fagner da Silva Cristovam

Processo de ETL Voltado para Jurimetria

Artigo apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Estatística e Informática

Curso de Bacharelado em Sistemas de Informação

Orientador: Silvana Bocanegra

Recife

2021

À todos que, de alguma maneira, participaram desta caminhada.

Agradecimentos

Agradeço aos meus colegas da universidade e professores por todo o aprendizado que obtive nesses anos de curso e por todos os momentos memoráveis. Foram dias incríveis e que, desde já, deixam saudades.

Também agradeço a minha esposa por me apoiar e me aturar ao longo desses anos, mesmo que isso significasse, em muitos momentos, estar longe. Sei que não sou uma pessoa fácil de se lidar e, mesmo assim, permaneceu ao meu lado sempre trazendo boas palavras — palavras de conforto, incentivo e carinho. Simplesmente não consigo imaginar uma pessoa melhor para ser minha companheira e a cada dia que vivo, agradeço a sorte de tê-la ao meu lado.

Minha mãe é uma das pessoas mais importantes em minha jornada. Se hoje eu estou onde estou, foi por tudo que ela me proporcionou. Agradeço por todo o amor incondicional que me foi dado e pelo alicerce que ela é em minha vida. Sua participação no meu desenvolvimento como pessoa é inestimável. Sou extremamente afortunado por ter uma mulher tão forte e tão incrível como mãe.

Por fim, e muito mais importante, eu agradeço ao meu pai. Ele é uma pessoa fantástica em minha vida. É verdade que não conversávamos muito, normalmente era dito somente o necessário, mas isso não me impediu de aprender bastante com sua vivência ou tornou menor o meu amor por ele. Sua vida guiou os meus passos, ele sempre foi um exemplo de amizade, lealdade, força, empatia e amor. A cada dia tento imitá-lo, ser, ao menos, uma fração do homem que ele é. Calçar os seus sapatos não é fácil e a minha admiração por ele é elevada a cada tentativa. Vê-lo lutar contra o câncer por dois anos quebrou o meu coração, mas vê-lo em meio ao sofrimento de dores inexprimíveis e, mesmo assim, ter forças para ser otimista, companheiro, paciente e caridoso em pró do bem-estar de toda a família foi o maior legado que eu poderia ter. Peço desculpas por não ter conseguido me formar antes de sua partida, sei que ficaria muitíssimo feliz com essa conquista, mas, infelizmente, só consegui quase dois meses após sua despedida e foi extremamente doloroso fazê-lo sem a sua presença.

“Não corrigir nossas falhas é o mesmo que cometer novos erros.”
(Confúcio)

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

C933p Cristovam, Edson Fagner da Silva
Processo de ETL voltado para jurimetria / Edson Fagner da Silva Cristovam. - 2021.
27 f.

Orientadora: Silvana Bocanegra.
Inclui referências.

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, Bacharelado em Sistemas da Informação, Recife, 2021.

1. ETL. 2. Data warehouse. 3. SQL. 4. Python. 5. Jurimetria. I. Bocanegra, Silvana, orient. II. Título

CDD 004

Edson Fagner da Silva Cristovam

Processo de ETL Voltado para Jurimetria

Artigo apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Aprovada em: 20 de Dezembro de 2021.

BANCA EXAMINADORA

Victor Wanderley Costa de Medeiros
Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco

Rodrigo Soares
Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco

Processo de ETL Voltado para Jurimetria

Edson Fagner da Silva Cristovam [1]

[1]Departamento de Estatística e Informática – Universidade Federal Rural de Pernambuco
Rua Dom Manuel de Medeiros, s/n, - CEP: 52171-900 – Recife – PE – Brasil

fagner.cristovam@hotmail.com

Resumo. *Grandes empresas podem ter dificuldades em gerenciar numerosos processos judiciais em que são rés, pois uma equipe do tamanho necessário para gerir esses processos pode ser muito dispendiosa. Com a digitalização dos processos judiciais, uma estratégia que automatize a organização das informações processuais e que seja de fácil consulta pode ser muito proveitosa. Diante disso, o data warehouse parece ser uma técnica muito indicada para tal, por normalizar e padronizar as informações de diversas fontes que estão contidas nele. Mas para carregar os dados em um data warehouse, é necessário prepará-los com um método comumente conhecido por ETL. Com esse método podemos criar fluxos de extração, transformação e carga de dados que serão escritos no data warehouse. Com um data warehouse preenchido com dados judiciais, podemos utilizá-lo para auxiliar a tomada de decisão do setor jurídico das empresas por meio da jurimetria, que consiste na aplicação de ferramentas estatísticas ao Direito, fornecendo novos pontos de vista baseado em dados.*

Palavras-chaves*: ETL, Data Warehouse, SQL, Python, Jurimatria*

Abstract. *Large companies may have difficulty managing numerous court cases they are defendants in, as a team of the size needed to manage these cases can be very costly. With the digitization of judicial processes, a strategy that automates the adjustment of process information and is easy to consult can be very useful. Therefore, the data warehouse seems to be a very suitable technique for this, as it normalizes and standardizes information from different sources that are contained in it. But to load the data into a data warehouse, it's necessary to prepare it with a method commonly known as ETL. With this method we can create data extraction, transformation and load flows that will be written in the data warehouse. With a data warehouse filled with legal data, we can use it to help decision-making in the legal sector of companies through jurimetry, which consists of applying statistical tools to Law, providing new points of view based on data.*

Keywords*: ETL, Data Warehouse, SQL, Python, Jurismetry*

1. Introdução

A Intelivix é uma empresa B2B (*business-to-business* ou "de empresa para empresa", isto é, empresas que têm como clientes outras empresas) que oferece suporte jurídico para outras empresas usando diversas tecnologias da informação. O seu objetivo, em suas próprias palavras, é disponibilizar "*inteligência artificial para gestão de risco*

jurídico" [Intelivix 2021]. A companhia trata de todas as etapas desse processo, desde a obtenção dos dados dos processos nos sites dos tribunais em todo o Brasil, passando pela "limpeza" e estruturação desses dados, até a disponibilização das informações, obtidas por meio desses dados, em forma de painéis informativos.

1.1. Contexto/Problema

Muitas empresas que possuem uma grande quantidade de clientes e têm uma operação que abrange uma ampla área podem ter problemas para gerenciar uma grande quantidade de processos judiciais. Muitas vezes pode não compensar ter uma equipe jurídica do tamanho necessário para lidar com o volume de processos que são iniciados contra a empresa, devido aos grandes gastos que podem prejudicar a empresa financeiramente. Além disso, existe a dificuldade de lidar com a grande massa de ações judiciais dos mais diversos tipos, que avançam em ritmos distintos e que podem estar em diversas comarcas – "*território em que o juiz de primeiro grau irá exercer sua jurisdição e pode abranger um ou mais municípios, dependendo do número de habitantes e de eleitores, do movimento forense e da extensão territorial dos municípios do estado, entre outros aspectos*" [CNJ 2021].

1.2. Objetivos

Diante desse problema, o objetivo é capturar e armazenar continuamente os dados dos processos judiciais o mais breve possível na própria fonte, os sites dos tribunais, e, da mesma maneira, as suas atualizações.

Os dados que foram capturados dos sites tem diversas origens e, com isso, diversos formatos, isto é, não seguem uma mesma convenção. Então, após isso, é necessário realizar a limpeza desses dados e uma padronização para facilitar as manipulações posteriores. Depois de limpos, os dados serão extraídos, transformados e, a partir deles, serão geradas as mais diversas informações sobre os processos judiciais. É nesse ponto dos procedimentos que este artigo irá se abordar.

Por fim, uma vez transformadas, as informações serão cruzadas e relacionadas para serem apresentadas de uma maneira que facilite no direcionamento dos departamentos jurídicos, além de contribuir para a tomada de decisão dos gestores jurídicos, fornecendo-os uma visão única e simplificada.

2. Referencial Teórico

Antes de tratar sobre as definições mais específicas que este trabalho vai utilizar, gostaria de trazer uma definição mais geral em que o problema está incluído. Podemos compreender Sistemas de Informação como sistemas, sejam eles manuais ou automatizados, inter-relacionados que atuam armazenando, processando e transmitindo dados que representam informações para as partes interessadas (*stakeholders*). De forma mais geral, esses sistemas "*atuam em conjunto para o cumprimento de uma tarefa ou um objetivo*" [Turban et al. 2009].

O grupo básico de operações de um Sistema de Informação é constituído pelas entradas, pelo processamento e pelas saídas (Figura 1). A entrada é o conjunto de dados que será usado pelo sistema, o processamento é composto pela transformações e combinações que os dados de entradas serão submetidos e, por fim, a saída é o produto obtido por meio

do processamento dos dados de entrada. Em alguns casos, o sistema pode ser retroalimentado (*feedback*), fazendo com que o resultado de saída seja manipulado como dados de entrada em uma nova execução. Nesse processo, também é comum que ocorra o armazenamento dos dados de entrada e de saída para serem utilizados posteriormente, como pode ser visto na Figura 1.

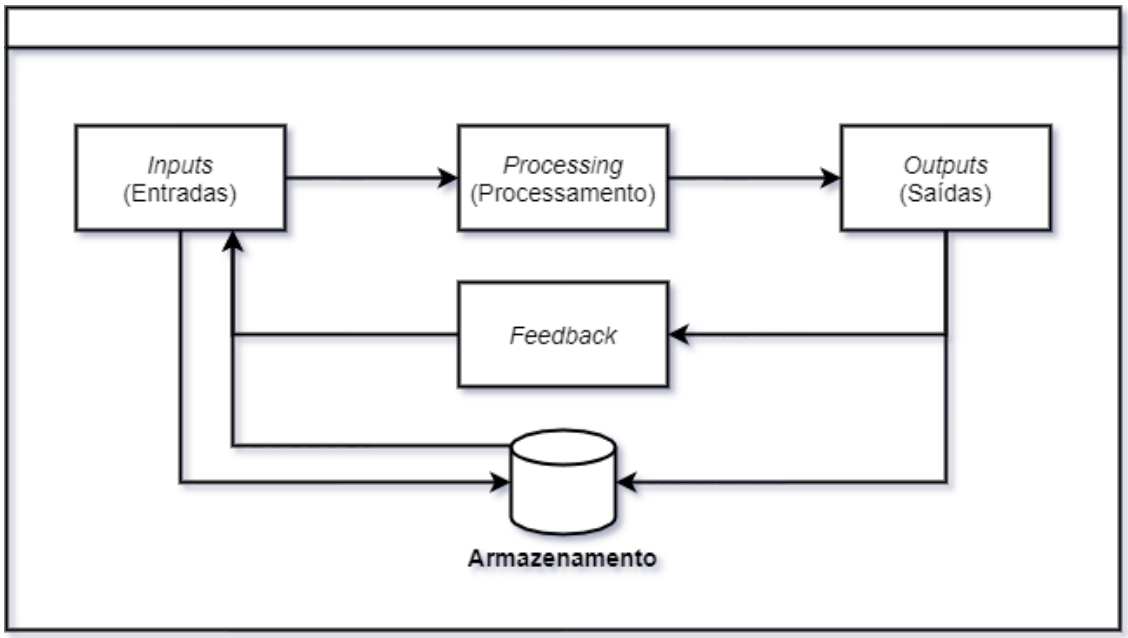

Figura 1. Operações básicas de um Sistema de Informação.

Os dados que servem como entrada do sistema "*são a menor parte de uma informação*", [Silva et al. 2016], e podem ser definidos como conhecimento bruto ou fatos isolados. Nesse estado, podemos tê-los como não adequadamente tratados para fornecer *gnose* aos *stakeholders*.

Por sua vez, a informação é a saída do sistema e descreve "*qualquer conhecimento do mundo real... mas que apresenta algum significado ou valor para quem o detém*" [Silva et al. 2016]. A informação é formada pelos dados de entrada do sistema após serem transformados, combinados "*relacionados logicamente e organizados para atingir um resultado definido*", [da Silva Vida et al. 2021], passando a fornecer conhecimento relevante para o processo de tomada de decisão.

Dessa maneira, temos os dados como matéria-prima do sistema e a informação como o produto. O nome de uma pessoa, uma data e um valor monetário são bons exemplos de dados, isolados eles não significam muito. No entanto, quando combinamos esses dados e atribuímos um significado ou contexto, como um depósito bancário, eles, os dados, passam a ter valor e se tornam informação.

Depois do processamento dos dados, e também antes, como já foi dito, os dados que compõem/comporão a informação precisam ser armazenados de maneira organizada. Para realizar essa organização é comum usar um banco de dados relacional (Figura 2), no qual "*são conjuntos de dados organizados e relacionados entre si com registros sobre fatos, pessoas, empresas, coisas ou lugares.*" [Rezende 2015]. Em vista disso, os bancos

de dados são essenciais para os sistemas de informação.

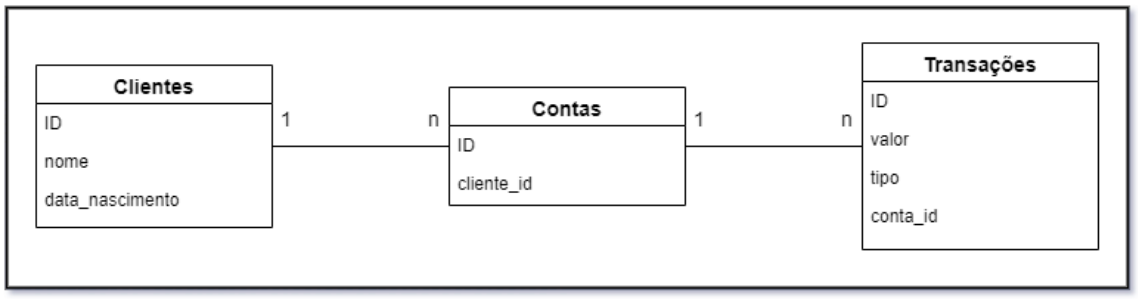

Figura 2. Exemplo de descrição das tabelas de um banco de dados relacional

Vale destacar que um conjunto de dados composto por dados aleatórios não pode ser qualificado como um banco de dados, pois existem algumas características que devem ser atendidas para que seja classificado como tal.

- **Minimundo**: O banco de dados deve representar uma parte do mundo real [da Silva Vida et al. 2021].
- **Dados com significado**: O banco de dados também tem que ser constituído por um conjunto lógico de dados que apresente algum significado [da Silva Vida et al. 2021].
- **Dados com objetivo**: Os dados que integram o banco necessitam ter ou cumprir um objetivo claro para o usuário ou aplicação, sendo prescindível o armazenamento dos dados que não contribuem com o propósito [da Silva Vida et al. 2021].

2.1. Data Warehouse

Ao longo das décadas, os computadores tiveram a sua capacidade de processamento e armazenamento aumentadas e juntamente com eles, houve um aumento da necessidade das empresas de investigar um grande volume de dados para nortear a sua tomada de decisão. Diante dessa necessidade, muitas abordagens foram criadas e "*em 1988, o braço irlandês da IBM lançou o termo business data warehouse para valorizar os dados empresariais que estavam sendo armazenados. Em 1990, surgiu uma abordagem que ficou conhecida como data warehouses, sendo responsável pela cópia dos dados armazenados em arquivos e em banco de dados para outro local.*" [da Silva Vida et al. 2021].

Os dados que ficam armazenados nesse "novo" banco de dados (*data warehouse*) são provindos de diversas base de dados oriunda das distintas áreas da empresa. Com isso, os *data warehouses* foram concebidos para consolidar o conhecimento dessa pluralidade de base de dados. De maneira simplória, podemos dizer que os *data warehouses* são nada mais que imensos banco de dados ou, como o nome sugere, armazém de dados que mantêm muitas informações históricas que, normalmente, não serão apagadas. Ainda podemos considerar os *data warehouses* como um conjunto ampliável e organizado de dados arquitetado com a finalidade de analisar dados históricos, relevantes para o negócio, de diversas fontes. Um *data warehouse* também deve manter a consistência dos dados, garantindo que não sejam modificados ou corrompidos. Os passos para a criação de um *data warehouse* podem ser descritos por "*coleta, consolidação, análise e pesquisa*", [da Silva Vida et al. 2021].

Dentre as vantagens do uso de *data warehouses* estão a diminuição da redundância de informações, já que ele centraliza os dados de diversas fontes e elimina as cópias desnecessárias. A padronização é outra vantagem, pois facilita a sua manipulação e análise, além de manter a integralidade da informação e torna a consulta mais acessível.

Existem quatro termos que apresentam características importantíssimas do *data warehouse*: "*orientado por assuntos, integrado, variável no tempo e não volatilidade*", [da Silva Vida et al. 2021].

Um *data warehouse* orientado por assuntos contém dados importantes e concernente ao negócio, ou seja, a organização das informações estão voltadas para o "assunto", diferente dos bancos de dados convencionais que são orientados a processos ou funções. Dessa maneira, um *data warehouse* não deve armazenar dados que não são "realmente" necessários para o negócio.

Todos os dados do *data warehouse* estão integrados, isso quer dizer que nele não existem dados não integrados, em nenhum caso [Turban et al. 2009]. Para ser consistente é necessária a padronização dos dados que entram no *data warehouse*, tornando-os simples e universais. As práticas mais comuns que introduzem inconsistências no banco são "*Codificação e Forma dos atributos*", [da Silva Vida et al. 2021]. A codificação ocorre quando uma informação pode ser expressada de diversas maneiras. Um exemplo disso é um dado "*boolean*" que pode ser expresso por "verdadeiro" e "falso", "0" e "1", "sim" e "não", etc. Já a forma dos atributos pode introduzir inconsistência quando são utilizadas medidas diferentes como graus celsius e kelvin, metro e milha, grama e onça.

A variável de tempo é essencial em cada registro de um *data warehouse*. Isso é indispensável para se ter um sistema histórico de dados. Esse sistema pode ser apresentado de maneira simples, como uma linha do tempo, ou mais complexa e detalhada. "*Os dados em um data warehouse podem ser considerados uma longa série de snapshots*", [Silva et al. 2016], ou seja, semelhante a uma coleção de registros de vários momento no tempo (diário, semanal, mensal, anual, etc.) e análogo a documentos que não podem ser alterados (apesar de ocorrer em momentos excepcionais, mesmo sendo incorreto).

A não volatilidade do *data warehouse* diz respeito às operações feitas nele. Enquanto um banco de dados comum tem por operações ordinárias a seleção, inclusão, atualização e exclusão, o *data warehouse* executa somente a carga inicial e a consulta dos dados, isto significa que o dados não são alterados ou excluídos dentro dele o que o torna, nesse aspecto, mais simples que um banco de dados comum.

Existem duas abordagens quanto ao modelo de um *data warehouse*, a abordagem relacional e a abordagem multidimensional. Na abordagem relacional os dados são organizados em um formato de tabela de duas dimensões (linhas e colunas). Nessa perspectiva, cada registro é uma linha da tabela, que por sua vez é dividida em colunas. Já o modelo multidimensional arranja as tabelas de um banco de dados relacional de modo a expressar os dados em várias dimensões por meio de tabelas de fatos e tabelas de dimensão usando esquemas em estrela (Figura 3) ou em floco de neve (Figura 4). O esquema em estrela usa uma tabela de fato central que contém as métricas desejadas para a análise. Ela é ligada a várias tabelas de dimensão, nas quais cada tabela desse tipo representa uma dimensão e possui a descrição dos elementos que serão analisados. Esse tipo de esquema é muito utilizado por sua simplicidade, mas apresenta redundância de dados,

pois as tabelas não estão normalizadas. Enquanto isso, o esquema em floco de neve é uma extensão do esquema em estrela, pois ele possui a mesma organização em fatos e dimensões, porém, adicionando tabelas de "subdimensão" relacionadas às tabelas de dimensão. Enquanto as tabelas do esquema em estrela não são normalizadas, isto é, podem ter dados repetidos, as tabelas do esquema em floco de neve são normalizadas graças as tabelas adicionais, o que reduz a redundância dos dados, tornando menor o armazenamento necessário.

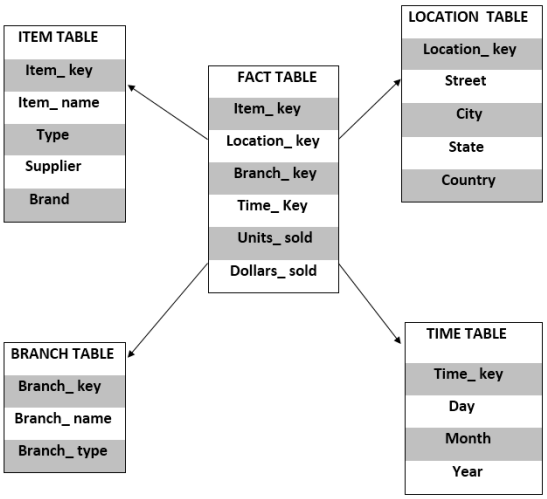

Figura 3. Esquema em estrela.
Fonte: https://www.educba.com/star-schema-vs-snowflake-schema/

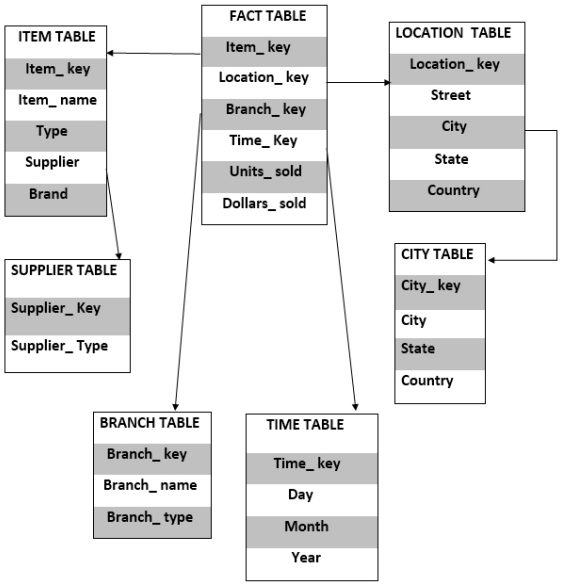

Figura 4. Esquema em floco de neve
Fonte: https://www.educba.com/star-schema-vs-snowflake-schema/

2.2. ETL

O *data warehouse* é uma ótima maneira de armazenar grandes quantidades de dados provenientes de diversas fontes. Essa enorme massa de dados é agrupada com o

objetivo de criar um ambiente preparado para efetuar análises complexas para auxiliar na tomada de decisão por meio da descoberta de informações sobre os diferentes setores do seu negócio.

A fim de que um *data warehouse* se mantenha atualizado para fornecer boas inspirações, é necessário que ele seja carregado de tempos em tempos (diária ou semanalmente, por exemplo, sempre observando o impacto que a carga causa aos sistemas de origem e também atentando à necessidade de atualização dos dados), transcrevendo os dados das variadas fontes de origem para dentro dele. Esse processo de copiar os dados das fontes, transformá-los para se ajustar ao esquema do *data warehouse* e carregá-los no próprio *data warehouse* é conhecido por ETL (Figura 5), sigla em inglês para *Extract, Transform and Load* ou extração, transformação e carga, em uma tradução livre. Desse modo, os dados importantes das fontes são enriquecidos para gerar informações úteis que serão depositadas no *data warehouse*.

As principais etapas do ETL são definidas da seguinte forma:

- **Extração**: consiste no processo de ler os dados de uma fonte de dados particular e extrair dela um subconjunto de dados [Goar et al. 2010].
- **Transformação**: corresponde ao método de modificação/combinação dos dados extraídos previamente para adequá-los aos requisitos de uma nova base de dados [Goar et al. 2010].
- **Carga**: o processo de escrever os dados transformados anteriormente na base de dados de interesse (*data warehouse*) [Goar et al. 2010].

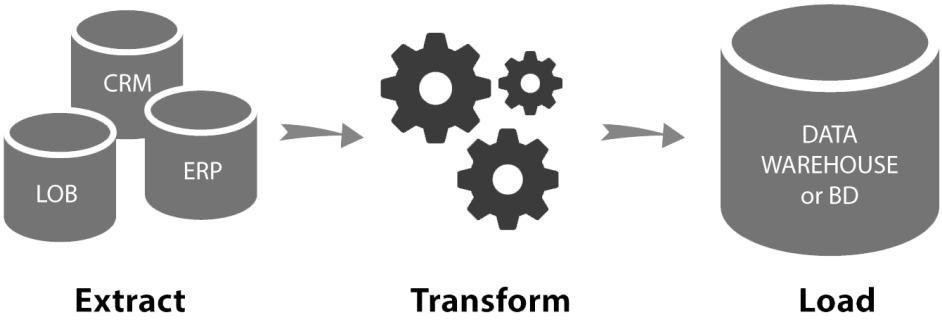

Figura 5. Representação do processo de ETL
Fonte: https://blog.indicium.tech/etl-vs-elt-diferencas/

Essa tarefa de carga do *data warehouse* pode ser bastante complexa, pois os dados que são oriundos das diversas bases de dados podem seguir padrões distintos uns dos outros. Por isso, se faz necessária a extração, limpeza e formatação, transformação e integração apropriada para não prejudicar a qualidade das consultas realizadas no *data*

warehouse. De acordo com especialistas do setor, aproximadamente 60-80 por cento do esforço empregado em um projeto de *data warehouse* é aplicado no processo de ETL [Goar et al. 2010].

2.2.1. Extração

A extração é a primeira etapa do ETL, nela os dados serão extraídos das mais diversas fontes, bancos de dados, sistemas de arquivos, páginas da *web*, etc, para uma área intermediária conhecida como *staging area* ou área de preparação. Essa transferência de dados é feita para reduzir a carga sobre os sistemas que detêm as fontes de dados e também do próprio *data warehouse*. Nesse ponto, os dados, sejam eles estruturados ou não estruturados, são importados em uma organização estável e de fácil consulta para um único local. Na área de preparação serão feitas as transformações necessárias para adequar os dados aos requisitos do destino, o *data warehouse*, por isso, ainda não é possível criar qualquer tipo de relatório para o usuário final ou realizar consulta neste sentido.

2.2.2. Transformação

A partir da área de preparação, segue-se a etapa de transformação dos dados. Essa etapa é responsável por modificar os dados brutos e estruturá-los para que estejam adequados para serem introduzidos no *data warehouse* de destino. As transformações aplicadas aqui seguem as regras de negócio já estabelecidas para enriquecer os dados, aumentando a sua qualidade. Essas conversões também têm o objetivo de tornar os dados mais acessíveis e fazer com que os relatórios gerados no *data warehouse* atendam os requisitos do negócio.

Existem diversos tipos de atividades de transformação de dados que são realizadas nessa etapa. Abaixo estão listadas algumas da mais comuns:

- Limpeza, filtragem, eliminação de informações duplicadas, validação e autenticação de dados [da Silva Vida et al. 2021].
- Execução de cálculos, tradução ou resumo dos dados brutos, conversão de moedas ou unidades de medida, padronização da codificação das *strings* [da Silva Vida et al. 2021].
- Ocultação de dados sensíveis ou protegidos por regulamentações governamentais ou de setores específicos da empresa [da Silva Vida et al. 2021].
- Estruturar os dados em forma de tabelas para condizer com o esquema do *data warehouse* final [da Silva Vida et al. 2021].
- "*Outras tarefas com o objetivo de melhorar a qualidade dos dados.*", [da Silva Vida et al. 2021].

A etapa de transformação é a parte mais custosa, computacionalmente, do ETL, devido a isso, é recomendado que ela seja executada na área de preparação para não afetar negativamente o desempenho da própria etapa, dos sistemas que originam os dados e do *data warehouse*.

2.2.3. Carga

Por fim, temos a etapa de carga. Ela é a responsável por transferir os dados após serem transformados, copiando-os da área de preparação para o *data warehouse* de destino para serem usados na geração de visões que guiarão a tomada de decisão. A carga pode ser realizada de duas maneiras, a carga total ou a carga incremental, que ocorre em períodos programados.

O carregamento total é quando "*tudo o que vem da linha de montagem de transformação*", [da Silva Vida et al. 2021], vai para o *data warehouse*. Ela é bastante útil quando se quer realizar pesquisas, no entanto, os dados podem crescer desmedidamente, o que torna esse processo muito custoso. É comum que esse tipo de carga ocorra fora do horário comercial, de maneira automatizada, para evitar sobrecarregar o *data warehouse* enquanto ele é usado pelos usuários finais.

Já o carregamento incremental introduz os dados no *data warehouse* frequentemente, em cargas programadas. No entanto, é feito um carregamento inicial com todos os dados, para, somente após isso, os dados serem inseridos regularmente. Nesse método, os dados só serão incluídos caso não existam registros idênticos no sistema objetivo, escrevendo somente as informações novas e ignorando linhas de lançamentos antigos. Esse tipo de abordagem permite a construção de *data warehouses* menos custosos por baratear a sua manutenção, além de permitir que novos dados sejam incluídos rapidamente no *data warehouse*, viabilizando uma tomada de decisão mais veloz. Contudo, é comum que sejam feitas cargas completas no *data warehouse* para substituir todos os dados contidos nele, mas isso ocorre em uma frequência menor.

O ETL de um *data warehouse* pode ser organizado em forma de *pipeline*, no qual as etapas são conectadas em série de modo que a saída de um passo sirva como entrada do passo posterior. Dessa forma, os estágios do ETL podem ser executados em paralelo, permitindo que vários registros sejam processados simultaneamente, eliminando a necessidade de esperar que um registro passe por todo o ETL para então iniciar o processamento do próximo registro [da Silva Vida et al. 2021].

Assim, o ETL é projetado para fornecer valor ao negócio da empresa por meio da união dos dados, do enriquecimento, da estruturação, da padronização e da eliminação de redundância que eles podem conter. Ele também eleva a qualidade dos dados e permite que os *stakeholders* possam acessar e manipular as informações oriundas de diversas fontes com facilidade e clareza. Outro ponto positivo é que o ETL aumenta o campo de visão da análise, devido à integralidade dos dados proveniente do máximo de fontes relevantes reunidas em um único local, no caso, o *data warehouse*. À vista disso, podemos dizer que o processo de ELT amplia os horizontes da empresa tanto pela qualidade e completude das informações que são entregues, permitindo que as decisões tomadas sejam apoiadas em dados reais, quanto pelo tempo que ele dá aos *stakeholders* por retirar deles o trabalho de reunir essa grande quantidade de dados.

3. Ferramentas Utilizadas

Esta seção irá tratar das principais ferramentas utilizadas no projeto para a execução de todas as etapas do ETL, desde a extração dos dados, passando pela transformação e finalizando com a sua carga.

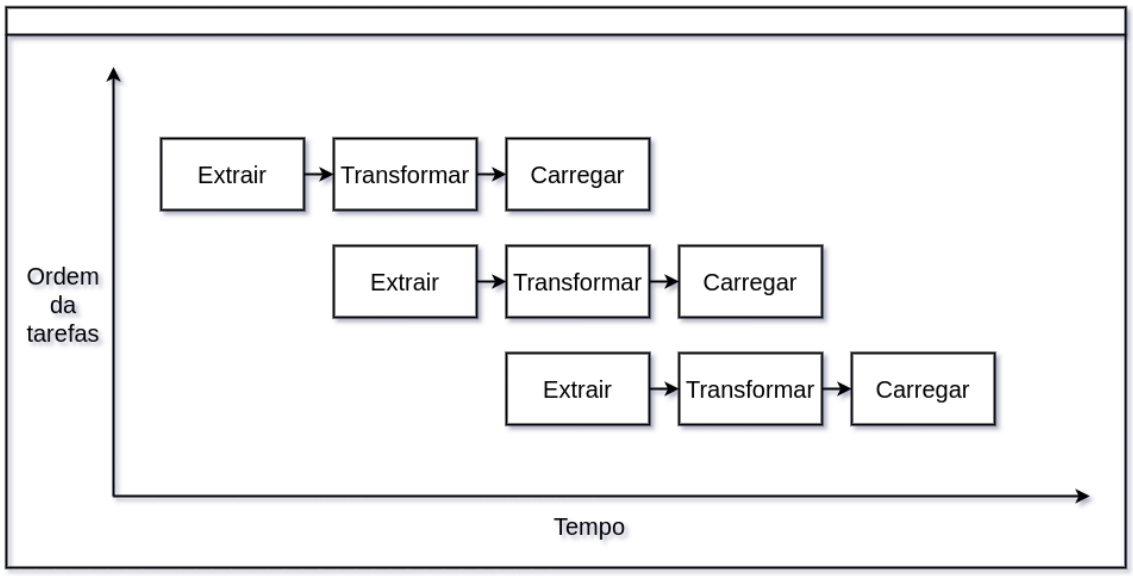

Figura 6. Paralelização dos *pipelines* de um ETL

3.1. Python

Python é uma linguagem de programação desenvolvida no final dos anos 80. Ela foi concebida pelo matemático holandês, Guido van Rossum, que a criou quando sentiu a necessidade de uma linguagem mais simples de se trabalhar que a C e que não tivesse as limitações do *shell script*, linguagem de comando utilizada por ele [de Barros Maciel 2020]. Segundo pesquisa do Stack Overflow, um site de perguntas e respostas para programadores, Python é a terceira linguagem mais popular na plataforma [Overflow 2021]. Ela é bastante conhecida no seu uso em automação de tarefas, desenvolvimento web e aprendizado de máquina devido à sua sintaxe sucinta e breviloquente [de Barros Maciel 2020].

Python é uma linguagem de alto nível, isto significa que, teoricamente, ela está mais próxima da linguagem natural do que da linguagem de máquina. Também é uma linguagem interpretada, na qual o código fonte é convertido para o formato intermediário chamado de *bytecode* e o interpretador fica responsável por converter o *bytecode* para o formato próprio para ser executado no sistema operacional desejado. Python é uma linguagem de programação multiparadigma, que se encaixa em muitos paradigmas como o imperativo e funcional. A tipagem do Python é dinâmica, pois a linguagem determina o tipo de dado de uma variável baseada no valor que essa variável armazenará, não havendo necessidade de declarar o tipo explicitamente [de Barros Maciel 2020].

3.2. Amazon Web Services

Amazon Web Services, ou simplesmente AWS, é a plataforma de nuvem da Amazon que disponibiliza mais de 200 serviços de *datacenters* ao redor do globo. A plataforma oferece infraestrutura em nuvem como computação, armazenamento de arquivos ou em banco de dados, criptografia, ferramentas de *machine learning*, *data lakes* e *internet* das coisas, além de suporte ao Docker e às linguagens de programação: .NET, Python, Java, JavaScript e PHP. A AWS dispõe de uma infraestrutura que satisfaz requisitos de segurança militares, compatível com 90 normas de segurança e certificações. Sua rede

oferece alta redundância e taxa de transferência com baixa latência, estando presente em 81 zonas de disponibilidade que estão distribuídas em 25 regiões geográficas ao redor de globo [AWS 2021c].

3.2.1. AWS Athena

O AWS Athena, ou somente Athena, é um serviço de consultas de dados *serverless*, isto é, que segue um modelo de execução de computação em nuvem alocando recursos de processamento sob demanda e que elimina a necessidade do usuário de gerenciar a infraestrutura do servidor. O serviço é baseado no *software* de código aberto chamado Presto, da The Presto Foundation, ele usa o padrão SQL (*Structured Query Language* ou Linguagem de Consulta Estruturada) para acessar os dados dos arquivos que ficam armazenados no Amazon S3 (Seção 3.2.2). É possível configurar o Athena para que ele trate um ou mais conjuntos de arquivos de forma análoga ao modelo relacional, estruturando-os em forma de tabelas e atribuindo tipos para as colunas, isso graças à sua integração com o AWS Glue Data Catalog. O AWS Athena tem suporte para arquivos organizados em linhas como CSV, Avro e JSON e os formatos colunares ORC e Parquet [AWS 2021b].

3.2.2. Amazon S3

Amazon Simple Storage Service (Amazon S3) é um serviço da AWS para armazenamento de arquivos de forma escalável. Os dados armazenados no S3 podem ser usados em diversos casos de uso como *data lakes*, *sites* e aplicações móveis. O serviço também oferece suporte à criptografia e ao gerenciamento de autorizações, permitindo que a organização administre o nível de acesso dos seus colaboradores. Ele fornece uma durabilidade para os arquivos de 99,99%, pois realiza cópias automáticas de todos os documentos e os conserva em diversas regiões, aumentando sua disponibilidade e protegendo-os contra erros ou falhas [AWS 2021a].

3.3. MongoDB

O MongoDB é um *software* de banco de dados orientado a documentos com o código aberto, sendo classificado como um NoSQL, também conhecido como *Not only SQL* (Não somente SQL) – um termo usado para categorizar bancos de dados não relacionais, isto é, uma classe de bancos de dados que armazena e recupera as informações que são estruturadas de uma maneira diferente à da abordagem com relações tabulares. Ele é escrito na linguagem C++ e é distribuído em múltiplas plataformas. A organização dos documentos é feita em um formato similar ao JSON, em uma disposição composta por um par de campo e valor, em que os valores dos campos podem conter dados de tipos primitivos, outros documentos ou vetores, o que fornece a possibilidade de se criar hierarquias complexas que podem ser indexadas, o que facilita a recuperação das informações. Os documentos são agrupados em coleções, uma perspectiva análoga às tabelas do modelo relacional [MongoDB 2021b].

O MongoDB apresenta suporte a buscas por campo, intervalo de valor, expressões regulares e também por funções JavaScript personalizadas [MongoDB 2021b].

As agregações de dados do MongoDB podem ser feitas usando um *pipeline* de agregação ou com o *map-reduce*, que está depreciado e possui um desempenho pior que o do *pipeline* [MongoDB 2021a]. Além disso, ele é capaz de executar transação ACID – acrônimo em inglês para: *Atomicity*, *Consistency*, *Isolation*, *Durability*, ou em uma tradução livre, Atomicidade, Consistência, Isolamento e Durabilidade — que é um conjunto de propriedades para as transações em banco de dados [Kobielus 2021].

3.4. PostgreSQL

PostgreSQL é um Sistema Gerenciador de Banco de Dados (SGBD) de código aberto baseado na estrutura objeto-relacional. Ele utiliza o padrão SQL, estando de acordo com 170 dos 179 requisitos obrigatórios da norma, combinado com recursos que escalam as cargas de trabalho [Group 2021a].

Ele foi desenvolvido na Universidade da Califórnia em Berkeley como parte do projeto POSTGRES (1986) com liderança do Professor Michael Stonebraker e patrocínio da *Defense Advanced Research Projects Agency* (DARPA), do *Army Research Office* (ARO), da *National Science Foundation* (NSF) e do ESL Inc. Somente em 1996 recebeu o nome que é usado até o momento, PostgreSQL, iniciando na versão 6, pois levou em consideração as versões desde o projeto original [Group 2021b].

O PostgresSQL é compatível com transações ACID e oferece suporte a dados geoespaciais com o "PostGIS", a documentos JSON e XML, a funções personalizadas e a linguagens procedurais como: PL/PGSQL, Perl, Python. Com ele também é possível realizar consultas paralelizadas, particionamento de tabelas, indexação com *b-tree*, *multicolumn*, *expressions* ou *partial* e expressar caminhos no formato SQL/JSON. Além disso, sobre segurança, o PostgresSQL permite autenticação com GSSAPI, SSPI, LDAP, SCRAM-SHA-256 e certificado, bem como um sólido controle de acesso. Por fim, o PostgresSQL é altamente extensível, permitindo que o usuário possa construir soluções usando o próprio PostgresSQL como base [Group 2021a].

4. Abordagem proposta

Empresas que atuam em uma grande área do território nacional atendendo a um grande número de clientes podem ser ré de diversas ações judiciais iniciadas pelos consumidores que se sentiram lesados com os serviços prestados. Desse modo, essas numerosas ações judiciais podem ocorrer em diferentes locais ou comarcas, "*território em que o juiz de primeiro grau irá exercer sua jurisdição e pode abranger um ou mais municípios*" [CNJ 2021], o que dificulta o acompanhamento dos processos forenses. Assim sendo, manter uma equipe jurídica do tamanho necessário para cuidar dessa volumosa soma de processos, manualmente, pode ser muito custoso para a empresa, sendo capaz de prejudicar o bem-estar financeiro e jurídico dela.

Diante desse cenário, surgiram empresas que ofertam "inteligência jurídica", acompanhando os inúmeros processos jurídicos por meio da sua coleta, armazenando-os em bancos de dados. Esses dados processuais são disponibilizados pelos sistemas de justiça providenciados por meio das suas plataformas digitais como o Projudi do TJPE [de Justiça de Pernambuco 2021b] e o e-SAJ CE [de Justiça do Estado do Ceará 2021]. Essa coleta otimiza o trabalho das equipes jurídicas das empresas, no entanto, esses dados podem ser usados para fornecer muito mais informações para o setor jurídico da empresa

e até contribuir para o seu *business intelligence* (BI), revelando novas perspectivas que servirão de base para a tomada de decisão.

Mas no Brasil existem diversas plataformas digitais de justiça, em que cada uma disponibiliza os dados à sua maneira. Essa diferenciação na representação de um mesmo tipo de informação é uma problema quando se deseja usar técnicas de *business intelligence* por meio de um *data warehouse* (Seção 2.1), pois sem um tratamento prévio, o agrupamento dos dados nessa situação não apresentará resultados satisfatórios. Então é necessário utilizar alguma estratégia para padronizar esses dados, sendo o ETL (Seção 2.2) uma das técnicas mais utilizadas para tal.

De forma geral, o projeto está organizado da seguinte maneira: Os dados dos processos que foram capturados dos sites dos tribunais são armazenados em um banco de dados MongoDB (seção 3.3). Esses dados processuais serão copiados do MongoDB para serem computados por um *pipeline* de algoritmos em Python (Seção 3.1) chamado "Step Pipeline". Os dados fornecidos pelas empresas em formato CSV também podem ser usados no processamento do *pipeline*. Essa etapa de cópia dos dados do banco e dos arquivos CSV para o Step Pipeline equivalem a "Extração" do ETL (Seção 2.2.1). Após serem computados, os dados dos processos serão armazenados no Amazon Simple Storage Service (Seção 3.2.2) para, em seguida, serem convertidos em tabelas pelo AWS Athena (Seção 3.2.1). Todo esse fluxo iniciado no Step Pipeline e finalizado no AWS Athena configura a fase de "Transformação" do ETL (Seção 2.2.2). Em conclusão, após serem transformados, os dados em formato de tabela serão copiados do Athena para o PostgreSQL (Seção 3.4). Nesse ponto, os dados estão prontos para serem carregados em qualquer sistema de análise de negócios, o sistema utilizado pela empresa é o Power BI da Microsoft. Desse modo, esse estágio final, do PostgreSQL ao Power BI, corresponde à "Carga" do ETL (Seção 2.2.3). Todo esse fluxo pode ser visto na Figura 7.

A minha função na empresa consiste em dar manutenção nos códigos responsáveis pela extração dos dados, no Step Pipeline, nos algoritmos que copiam os dados para o S3, que criam as tabelas no Athena e as transcrevem para o PostgreSQL. Também fico encarregado de atualizar esses códigos à medida que novos requisitos são solicitados, bem como a criação de novos steps ao passo que mais regras de negócio surgem.

4.1. Processos

Os elementos contidos no processo são imprescindíveis para a construção do *data warehouse* focado na obtenção de conhecimento jurídico e na aplicação da jurimetria, isto é, "*a estatística aplicada ao Direito*" [Law 2021].

Os principais dados de um processo judicial, mas não se limitando somente a eles, são a sua Numeração Processual Única (NPU) ou somente número, as partes envolvidas, os andamentos ou movimentações e a fonte.

A NPU é composta por vinte dígitos e foi instituída pelo Conselho Nacional de Justiça (CNJ) com o intuito de facilitar a consulta das informações referente a um processo, pois os números dos processos mudavam frequentemente em cada instância ou recurso. A Numeração Processual Única vale para os tribunais de todo o país e apresentam a seguinte estrutura "NNNNNNN-DD.AAAA.J.TR.OOOO" no qual [de Justiça de Pernambuco 2021a]:

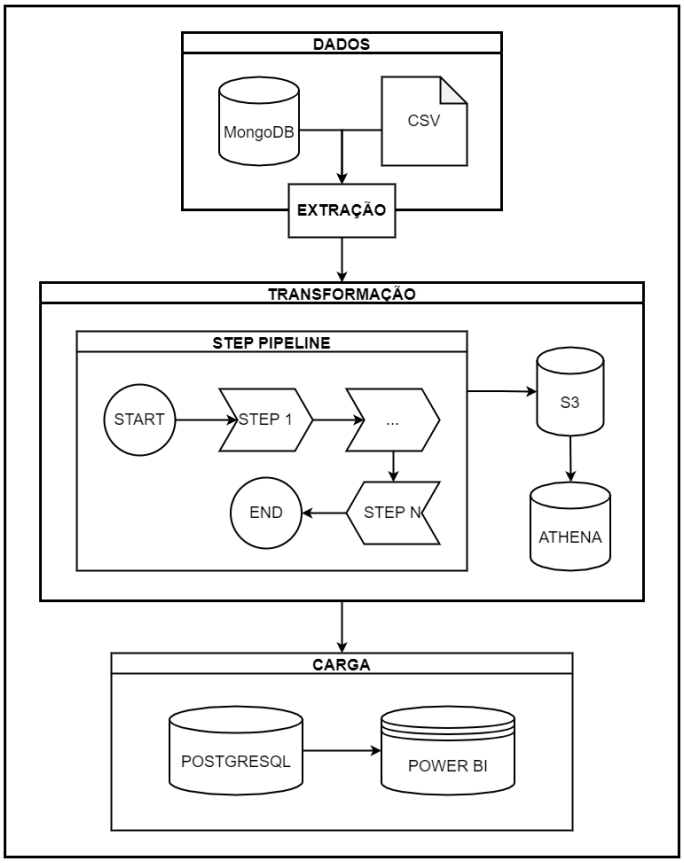

Figura 7. Visão geral da arquitetura do projeto.

- **NNNNNNN-DD**: representam o número sequencial do processo e seu dígito verificador [da 4ºRegião 2021].
- **AAAA**: diz respeito ao ano de avaliação do processo [da 4ºRegião 2021].
- **J**: informa qual o órgão ou segmento do Poder Judiciário que o processo pertence [da 4ºRegião 2021].
- **TR**: indica o tribunal do respectivo segmento ou circunscrição judiciária [da 4ºRegião 2021].
- **OOOO**: remete à unidade de origem do processo [da 4ºRegião 2021].

As partes do processo são compostas pelo autor, que desempenha o papel de polo ativo do processo, ou seja, aquele que recorre à tutela jurídica do estado, tomando a posição ativa, e pelo réu, que realiza o papel do polo passivo, estando sujeito à ação processual iniciada pelo autor.

As movimentações de um processo são um conjunto de informações que comunicam a evolução processual ao longo da tramitação judicial. Elas podem conter os mais diferentes tipos de conhecimentos a respeito do processo, como os informes sobre o dia de uma audiência, notificação de emissão de intimação ou até mesmo a notícia da sentença dada pelo juiz. Nesse sentido, é usando as informações dos andamentos que o ETL fará a maior parte do seu processamento. As movimentações podem ser compostas pelos seguintes campos:

- **Título**: informa o assunto que o andamento trata.

- **Texto**: corpo do texto (opcional).
- **Complemento**: informações complementares (opcional)
- **Data**: dia e hora que o andamento foi disponibilizado no sistema de justiça.

Por fim, a fonte informa de maneira explícita em qual sistema o processo foi extraído. Essa informação é útil para definir quais rotinas serão aplicadas na etapa de transformação (Seção 2.2.2) do ETL.

4.2. ETL para Jurimetria

A fase de "Transformação" da técnica de ETL usada neste trabalho segue o modelo de *pipeline*, no qual cada uma das etapas do fluxo de trabalho possuem uma tarefa muito bem estabelecida e são conectadas em série (Figura 8). Cada nó dessa fila é uma estrutura que recebe o nome de *step*.

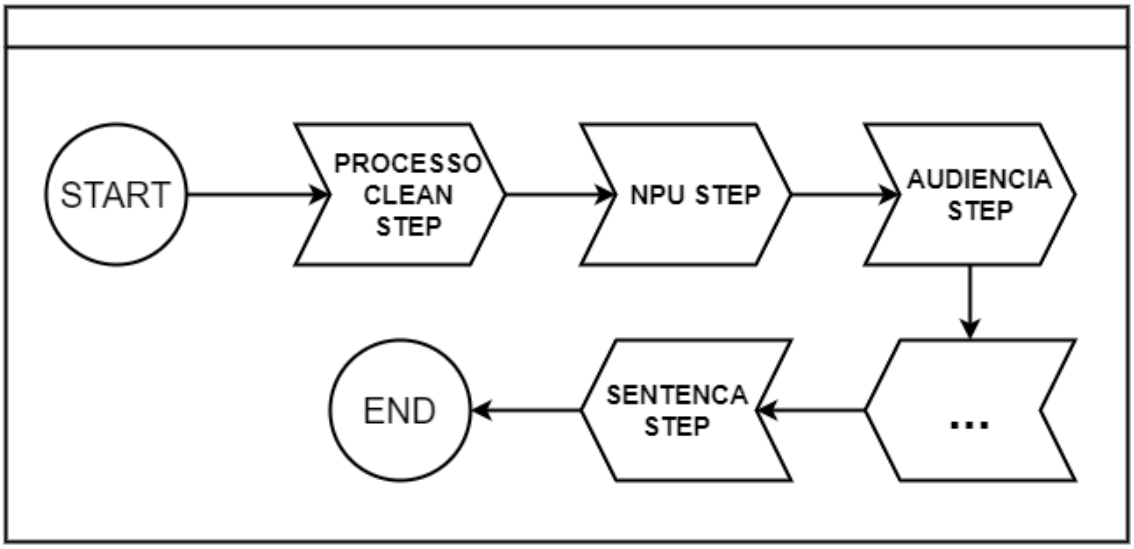

Figura 8. Fluxo de execução dos *steps*.

O *step* é um conjunto de operações que serão aplicadas ao processo para realizar um objetivo bem definido. No projeto, ele é codificado como uma classe Python (Seção 3.1) com um método obrigatório chamado *transform*. O *transform* é a função que receberá o processo e aplicará as inferências e transformações necessárias para alcançar o seu objetivo.

Os processos são extraídos dos vários sistemas jurídicos pela equipe de extração e armazenados como documentos em um banco de dados MongoDB (Seção 3.3). Esses documentos serão usados no processamento do ETL, podendo ser combinados com informações complementares fornecidas pela empresa (CSV), que preparará os dados e também criará novos para serem inseridos no *data warehouse*. No ETL, os documentos são lidos do MongoDB e estruturados como um dicionário do Python (Figura 9). A cópia desses processos e das informações fornecidas pela empresa equivalem à fase de "Extração" do ETL.

```
processo = {
    "fonte": "e-saj CE",
    "partes": [
        "autor": "nome do autor",
        "reu": "nome do réu"
    ],
    "numero": "01234567899876543210"
    "andamentos": [
        {
            "titulo": "Título da movimentação.",
            "texto": "Texto da movimentação.",
            "complemento": "Complemento da movimentação."
            "data": "03/11/2021 16:30:00"
        }
    ]
}
```

Figura 9. Exemplo de um processo como um dicionário do Python.

4.2.1. Steps

Para esse trabalho, não serão abordados todos os *steps* que são efetivamente realizados pela empresa, pois essas informações são sigilosas, por esse motivo, serão descritos apenas dois *steps*, o "Processo *Clean Step*" e o "Audiencia *Step*". Com exceção do primeiro *step*, que trata da normalização dos dados, cada um dos demais *steps* são responsáveis por adicionar um novo campo na raiz do dicionário que representa o processo (Figura 10). Normalmente esse campo tem o nome do próprio *step* que o gerou e seu valor é uma lista de dicionários. Esses novos campos darão origem às tabelas no banco de dados final, em que o nome da tabela será o nome do próprio campo, as suas colunas serão as chaves dos dicionários que estão na lista e os valores que popularão a tabela serão os valores dos dicionários em que a chave corresponda ao nome da coluna (Figura 11).

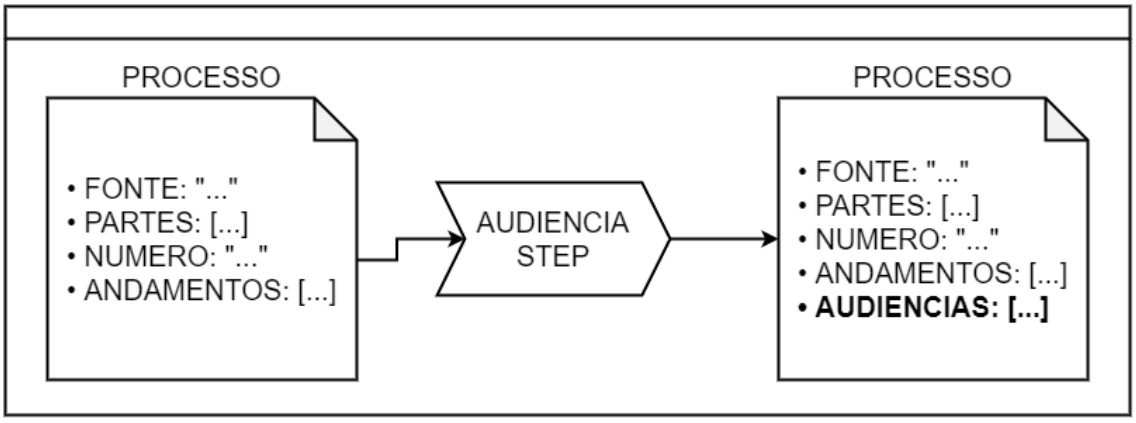

Figura 10. Exemplo de *step* enriquecendo o documento (processo) com novas informações.

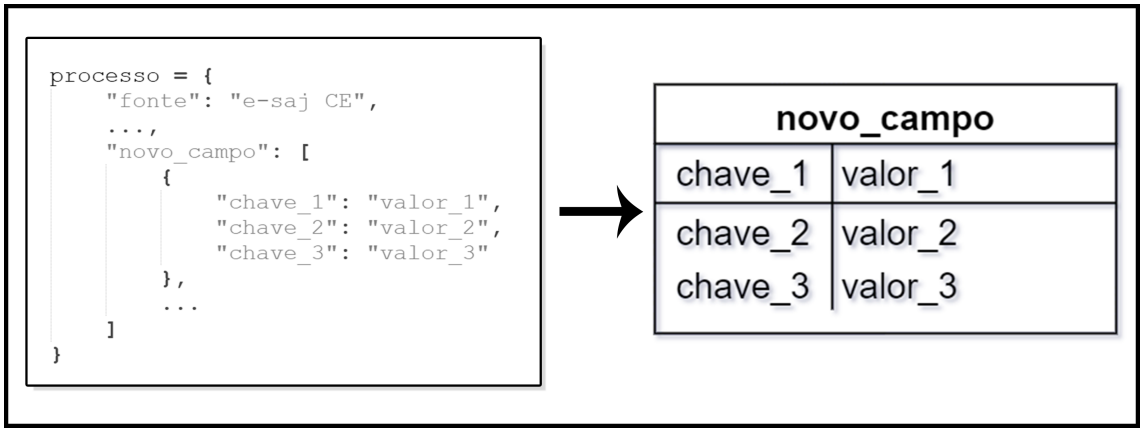

Figura 11. Conversão de novo campo inserido por um *step* em tabela.

O primeiro *step*, chamado de "Processo *Clean*", é o responsável pela normalização dos dados de um processo por meio de diversas conversões e substituições. Esse passo é necessário, pois as diversas informações que estão no processo são representadas como o tipo de dado *string*, o que tornaria algumas das comparações difíceis e menos diretas, e por serem capturadas de diversos sistemas, um mesmo tipo de informação pode não seguir um mesmo padrão. Além disso, alguns campos de textos podem estar mal formatados e conter caracteres especiais que aumentam a complexidade das buscas necessárias para o enriquecimento dos dados. Os principais casos que esse *step* trata são os tipos numéricos, as datas e os textos.

As *strings* que representam algum tipo numérico como inteiros (*integer*) ou números de ponto flutuante (*float*) devem ser convertidas para os seus respectivos tipos, pois somente dessa forma as agregações realizadas pelo *data warehouse*, como somas e médias, podem ser executadas corretamente. Para esse fim, as expressões regulares, ou "*regex*" (abreviação do inglês *regular expression*), são uma ótima ferramenta na preparação desses textos para serem transformados, porque conseguem representar uma infinidade de padrões de maneira bem simples. Expressões como "[0-9]+" ou "\d+" representam uma cadeia de números com 1 ou mais dígitos. Com isso, se torna muito simples a remoção de qualquer "impureza", ou seja, tudo aquilo que não exprime um número, antes de modificar a *string* para *interger* ou *float*.

As datas também precisam passar do formato de texto para um que reflita melhor as suas características e facilite a execução dos passos em que elas são essenciais. No nosso caso, o formato para o qual as datas em formato *string* serão convertidas será o *datetime* da linguagem Python e o *timestamp* quando forem inseridas no banco de dados final. Com ele é possível realizar facilmente diversas operações que seriam muito mais difíceis de serem executadas sobre *strings* e também padronizamos as várias formas que as datas podem apresentar (Ex: "03/11/2021", "03-11-2021" e "2021.11.03"). Com o *datetime* é possível comparar se uma data é maior que outra sem erros, ser maior significa que uma data é posterior a outra. Em um exemplo simples, colacionando duas datas em padrão de texto, em que a primeira data é "10/11/2021" e a segunda "11/10/2021", se efetuarmos a operação de maior que entre a primeira e segunda data, obteremos o resultado falso, porque esse tipo de operação entre *strings* é feita comparando carácter a carácter, porém, a primeira data é posterior a segunda, em outras palavras, o resultado

esperado é verdadeiro. Além de tudo, outras operações como verificar a distância (número de dias) entre datas se torna impraticável quando elas são *strings*.

E por último, será executada a normalização dos textos que já estão na tipagem correta (*string*). A finalidade dessa etapa é alterar o conteúdo de modo a facilitar a pesquisa de informações sobre ele, eliminando tudo aquilo que for desnecessário e simplificando algumas informações. Inicialmente serão substituídos todos os espaços múltiplos (mais de um espaço em sequência) e outros espaçamentos especiais como tabulação, quebra de linha, quebra de página, etc, por um espaço simples. Isso facilita a definição de alguns padrões de busca com múltiplos termos como "decreto a sentença conforme descrito no artigo XX...", tornando a escrita de uma busca exata ou de uma pesquisa com *regex* muito mais descomplicada. Outra padronização feita com o mesmo objetivo da transformação dos espaços extras e especiais é uniformização da "caixa das letras" dos textos, deixando todas em caixa alta ou caixa baixa, isso é feito, pois não é verdadeiro o resultado da comparação entre uma mesma letra maiúscula e minúscula. O efeito negativo que isso causa nas pesquisas pode ser exemplificado com a busca do termo "juntada de protocolo" no trecho "Juntada de protocolo de petição: cumprido em 01/05/2020.", em que o termo não seria encontrado, visto que a letra "J" da palavra "Juntada" está em caixa diferente em ambos os textos. Existe um tratamento "especial" que deve ser feito com os textos em português também com o objetivo de facilitar as buscas. Refere-se à substituição das vogais acentuadas, como "à" e "é", e da letra "cê-cedilha" pelos seus equivalentes na tabela ASCII (*American Standard Code for Information Interchange*). O problema que a não substituição dessas letras gera é semelhante ao que ocorre com a comparação entre letras em caixas diferentes.

Os demais *steps* não fazem alterações nos dados presentes no processo, mas adicionam novas informações com base em inferências feitas sobre os dados já presentes. De forma geral, as investigações executadas em cima dos dados são efetuadas usando três técnicas: uma pesquisa exata do termo, busca de padrão usando *regex* e o rastreio de expressões utilizando busca *fuzzy*, que são buscas focadas em encontrar correspondências por aproximação, ao invés de exatidão. Com base nos resultados obtidos dessas investigações, cada *step* pode introduzir novas informações ao processo, que juntamente com ele, servirá como entrada para o *step* seguinte.

O "Audiencia Step" ou *step* de audiência é responsável por buscar nos andamentos as designações de audiências, suas classificações ou tipos, o horário e o local em que irão ocorrer. As informações de uma audiência são descritas nos campos "titulo" e "texto" de um andamento, mas elas apresentam padrões distintos de sistema para sistema. Por exemplo, uma audiência de conciliação para o dia 03/11/2021 às 16:30 pode ser representada das seguintes formas:

- **Sistema TJPE**:
  "título": "audiencia de conciliacao"
  "texto": "designada para 03/11/2021 16:30 - 13a vara civel"
- **Sistema e-SAJ CE**:
  "título": "designacao de audiencia"
  "texto": "audiencia de conciliacao designada (03/11/2021 16:30 na 13a vara civel)"

Dessa maneira, o algoritmo do *step* de audiência (Figura 12) inicia com um fluxo

condicional que seleciona o padrão descrito como *regex* que extrai as informações das audiências que estão nos andamentos. Essa escolha é feita com base no atributo "fonte" do processo.

```
class AudienciaStep:
    def transform(processo):
        fonte = processo['fonte']
        if fonte == 'TJPE':
            pattern = tjpe_pattern()
        elif fonte == 'e-SAJ CE':
            ...
        else:
            pattern = default_pattern()

        audiencias = list()

        for andamento in processo['andamentos']:
            titulo = andamento['titulo']
            texto = andamento['texto']
            texto_completo = titulo + ' ' + texto
            match = pattern.search(texto_completo)
            if match:
                audiencia = {
                    'tipo': match.group(0),
                    'horario': match.group(1),
                    'local': match.group(2),
                }
                audiencias.append(audiencia)

        processo['audiencias'] = audiencias
```

Figura 12. Pseudocódigo do "Audiencia Step".

Após isso, uma estrutura de repetição percorrerá todos os andamentos do processo, um a um, da seguinte maneira: os campos "título" e "texto" do andamento serão concatenados com um espaço simples entre eles para formar uma única *string* chamada "texto_completo". Para a audiência do TJPE, descrita anteriormente, o "texto_completo" seria "audiencia de conciliacao designada para 03/11/2021 16:30 - 13a vara civel". Em seguida, o padrão *regex* é aplicado no "texto_completo" para buscar três grupos de dados: o tipo da audiência, o horário da audiência e o local da audiência. O padrão *regex* para o TJPE é expresso como "**(audiencia.*) designada para (\d{2}/\d{2}/\d{4} \d{2}:\d{2}) - (.*)**", em que cada valor entre parênteses representa um grupo. Caso o padrão *regex* encontre os três grupos no "texto_completo", um dicionário será criado com as chaves "tipo", "horário" e "local" e seus valores serão os respectivos grupos encontrados pelo padrão *regex* (Figura 13). Esse dicionário será armazenado em uma lista criada antes da estrutura de repetição ser iniciada. No entanto, se o padrão *regex* não encontrar os grupos, a estrutura de repetição passa para o próximo andamento.

Depois que a estrutura de repetição for concluída, um novo campo será criado no processo com um nome relacionado ao nome do *step*. Nesse caso, o nome do novo campo será "audiencias" e o seu valor vai ser a lista com os dicionários criados na estrutura de

```
{
    'tipo': 'audiencia de conciliacao',
    'horario': '03/11/2021 16:30',
    'local': '13a vara civel'
}
```

Figura 13. Dicionário com os dados que representam uma audiência.

repetição. Com isso, o *step* de audiências é concluído e o próximo *step* será iniciado.

Finalmente, após todos os processos passarem pelos *steps*, os dados serão armazenados em formato JSON no Amazon S3 (Seção 3.2.2). Em seguida, o Athena (Seção 3.2.1) é utilizado para estruturar esses dados que estão em JSON como tabelas de um modelo relacional a partir de arquivos no formato CSV (*Comma Separated Values*) que o próprio Athena criará. Depois disso, esses arquivos CSV são utilizados para exportar as informações para o banco de dados PostgreSQL (Seção 3.4), sendo esse o formato final pré *data warehouse*. Por mais contra-intuitivo que isso pareça, exportar o dados diretamente do Python para o PostgreSQL oferecia um desempenho bastante abaixo do que a abordagem supracitada, isso porque a maneira mais rápida de se carregar dados no PostgreSQL é utilizando CSV, exceto por usar dados em formato binário, o que diminuiu drasticamente o tempo de carga. Concluído o armazenamento no PostgreSQL, os dados estão prontos para serem carregados em qualquer sistema ou serviço de análise de negócios como o Power BI da Microsoft ou o Google Data Studio. Desse modo, essa etapa final, do PostgreSQL ao serviço de análise de negócios, corresponde à "Carga" do ETL.

5. Conclusões

Lidar com um imenso volume de dados é um problema muito comum para as empresas na "era do *Big Data*". Ainda nesse sentido, a tendência é que essas informações cresçam cada vez mais à medida que mais e mais sistemas são informatizados. Hoje, essa conjuntura já está presente na área jurídica.

Para tratar dessas questões, a estratégia da jurimetria aplicada em um *data warehouse* se mostra bastante eficaz para superar essa enorme massa de dados, usando-os para extrair informações e novas perspectivas a respeito do ambiente em que se está inserido.

No entanto, uma etapa anterior ao *data warehouse*, definida em três passos, é essencial para o seu êxito: a extração, a transformação e a carga desses dados deve padronizar e estruturar toda a porção de dados que irá compor o *data warehouse*.

Dessa maneira, esse aglomerado de dados se torna valioso para o negócio, mostrando para a empresa seus pontos fortes e fracos, nesse caso no âmbito jurídico, guiando as suas ações para as áreas que necessitam de sua atenção e podendo contribuir com a antecipação de problemas futuros por meio de uma abordagem estatística.

Atualmente, a maior parte dos dados que são utilizados pelo projeto percorrem um longo caminho até o seu destino. Eles transitam desde o MongoDB, passando pelo pro-

cessamento do Python e pela reestruturação do Athena, até desembocar no PostgreSQL. No entanto, o Athena é uma ferramenta "poderosa" o suficiente para realizar todas as operações necessárias para aplicar o ETL sobre o conjunto de dados, dispô-los em um formato colunar, como o Apache Parquet ou *Optimized Row Columnar* (ORC), que são formatos mais rápidos para a pesquisa quando comparados aos formatos em linha, e, finalmente, servir como um banco de dados fim que fornecerá as informações que serão carregadas no *data warehouse*. Destarte, essa seria uma ótima atualização para ser feita no futuro por diminuir o tempo de processamento total e por simplificar a metodologia empregada, já que diminuiria o número de sistemas que precisariam de manutenção.

Referências

AWS (2021a). Armazenamento s3 - simple storage service - amazon web services. https://aws.amazon.com/pt/s3/. Acessado: 30/11/2021.

AWS (2021b). Aws athena - serviço de consultas interativas sem servidor. https://aws.amazon.com/pt/athena/. Acessado: 30/11/2021.

AWS (2021c). O que é aws? como funciona amazon web services. https://aws.amazon.com/pt/what-is-aws/. Acessado: 30/11/2021.

CNJ, C. N. d. J. (2021). Cnj serviço: Saiba a diferença entre comarca, vara, entrância e instância - portal cnj. https://www.cnj.jus.br/cnj-servico-saiba-a-diferenca-entre-comarca-vara-entrancia-e-instancia/. Acessado: 15/11/2021.

da 4ºRegião, T. R. F. (2021). Código das unidades de origem dos processos. https://www.trf4.jus.br/trf4/controlador.php?acao=pagina_visualizar&id_pagina=895&idSede=1. Acessado: 29/11/2021.

da Silva Vida, E., Alves, N. S. R., Ferreira, R. G. C., de Souza, D. C., Barboza, F. F. M., de Oliveira, H. S., de Marque, L., Maschietto, L. G., and de Fátima Gonçalves, P. (2021). *Data Warehouse*. Sagah.

de Barros Maciel, F. (2020). *Python e Django: Desenvolvimento web Moderno e ágil*. Alta Books.

de Justiça de Pernambuco, T. (2021a). Judiciário adota novo sistema de numeração processual. https://tj-pe.jusbrasil.com.br/noticias/2049972/judiciario-adota-novo-sistema-de-numeracao-processual. Acessado: 29/11/2021.

de Justiça de Pernambuco, T. (2021b). (sistema cnj - processo judicial digital). https://www.tjpe.jus.br/projudi/. Acessado: 28/11/2021.

de Justiça do Estado do Ceará, T. (2021). e-saj. https://esaj.tjce.jus.br/esaj/portal.do. Acessado: 28/11/2021.

Goar, V., Sarangdevot, S., Tanwar, G., and Sharma, D. A. (2010). Improve performance of extract, transform and load (etl) in data warehouse. *International Journal on Computer Science and Engineering*.

Group, T. P. G. D. (2021a). Postgresql: About. https://www.postgresql.org/about/. Acessado: 02/12/2021.

Group, T. P. G. D. (2021b). Postgresql: Documentation: 14: 2. a brief history of postgresql. https://www.postgresql.org/docs/current/history.html. Acessado: 02/12/2021.

Intelivix (2021). Intelivix — inteligência artificial para gestão de risco jurídico. https://intelivix.com/. Acessado: 03/11/2021.

Kobielus, J. (2021). Mongodb drives nosql more deeply into enterprise opportunities - wikibon research. https://wikibon.com/mongodb-drives-nosql-deeply-enterprise-opportunities/. Acessado: 01/12/2021.

Law, N. (2021). Jurimetria: o que é e como usá-la na advocacia - new law. https://newlaw.com.br/jurimetria-direito/. Acessado: 30/11/2021.

MongoDB, I. (2021a). Aggregation pipeline — mongodb manual. https://docs.mongodb.com/manual/core/aggregation-pipeline/. Acessado: 01/12/2021.

MongoDB, I. (2021b). Introduction to mongodb — mongodb manual. https://docs.mongodb.com/manual/introduction/. Acessado: 01/12/2021.

Overflow, S. (2021). Stack overflow developer survey 2021. https://insights.stackoverflow.com/survey/2021. Acessado: 30/11/2021.

Rezende, D. A. (2015). *Inteligência Organizacional Como Modelo De Gestão Em Organizações Privadas E Públicas: Guia Para Projeto de Organizational Business Intelligence*. Atlas.

Silva, L. A. S., Peres, S. M. P., and Boscarioli, C. (2016). *Introdução à Mineração de Dados - Com Aplicações em R*. GEN LTC.

Turban, E., King, D., Aronson, J., and Sharda, R. (2009). *Business Intelligence: Um enfoque gerencial para a inteligência do negócio*. Bookman.