Projeto de Pipeline de Dados

Mestrado Profissional de Computação Aplicada
Disciplina: Coleta, Armazenamento e Visualização de Dados
Professor Fabio Lopes
Turma: [Turma 01A] - 2022/2
Aluno: Marcos Antonio Speca Junior – 72256826

Índice

Resumo

O objetivo deste trabalho foi estruturar um pipeline de dados para realizar a coleta, armazenamento, processamento e visualização de dados a respeito de um tema específico. O tema escolhido foi a comparação de processos judiciais do Tribunal de Justiça do estado de São Paulo (TJSP) de empresas que atuam no mesmo segmento.

O pipeline desenhado tem como objetivo realizar a coleta de processos e suas decisões de 1ª instância, armazenar os dados brutos, aplicar algumas tratativas de dados para sintetizá-los e posteriormente disponibilizar em uma camada analítica para que fosse possível realizar a comparação entre empresas do mesmo segmento e suas respectivas taxas de êxito no tribunal citado.

De uma maneira resumida o pipeline consistia em scripts de *webscrapping* em linguagem *Python*, armazenamento de dados brutos em um banco *NoSQL* (MongoDB), processamento e carga na camada analítica também utilizando scripts *Python*, armazenamento dos dados tratados em um banco relacional (PostgreSQL) e posteriormente análise e visualização dos dados em Microsoft Power BI.

Contexto Judicial no Brasil

Atualmente no Brasil existem 77,3[1] milhões de processos em tramitação nos tribunais do judiciário, sendo que em 2021 foram baixados aproximadamente 26,9 milhões e entraram (casos novos) 27,7 milhões de processos no mesmo ano. Estes números refletem processos de diversas naturezas e assuntos, em diversas áreas da justiça do Brasil.

Além dos números elevados de processos, um outro fato relevante é que a grande maioria destes processos já tramita em meios eletrônicos, ou seja, os dados destes processos estão armazenados em sistemas dos próprios tribunais e são disponibilizados para consulta, uma vez que que os dados e julgamentos destes processos devem ser um dado de acesso público conforme art. 93 inciso IX da Constituição Federal, excetuando os casos que tramitam em segredo de justiça.

O grande volume de processos, em número absoluto, tramita em tribunais estaduais, e compreendem processos da esfera cível, direito do consumidor e execuções fiscais (tributário).

Objetivo do Pipeline

Muitas grandes empresas no Brasil figuram nos processos como parte passiva do processo, e com isso gasta-se milhões de reais em defesas e indenizações, como condenações de danos materiais e danos morais.

[1] Fonte: Relatório Justiça em Números 2022: https://www.cnj.jus.br/wp-content/uploads/2022/09/justica-em-numeros-2022.pdf

Considerando este contexto de número elevado de processos, digitalização e disponibilização de forma pública, juntamente com os custos elevados de litígio que as empresas no brasil sofrem surge uma necessidade de se analisar estes dados e otimizar a performance de defesa e diminuição de gastos com processos judiciais.

Portanto o pipeline de coleta e análise de dados realizado neste trabalho visa produzir análises a respeito de processos judiciais do mesmo tema para empresas do mesmo segmento (concorrentes) e avaliar suas respectivas taxas de êxito em processos de consumidor.

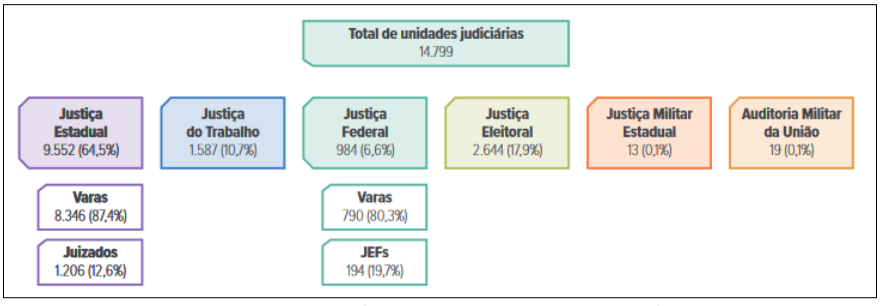
Visto que a análise destes dados pode trazer vantagem competitiva, o desafio é realizar esta coleta de dados visto que existem diversos problemas e dificuldades práticas no acesso e interpretação destes dados, como veremos no próximo tópico.

Coleta de dados dos tribunais

Desde dezembro de 2006 o judiciário brasileiro através da lei nº 11.419 passo a utilizar meios eletrônicos para tramitação de processos judiciais. Apesar de um avanço tecnológico e necessário a digitalização dos processos não foi realizada com a possibilidade de análise de informações consolidadas em mente.

Isso por que na lei de 2006 os tribunais ficaram livres para desenvolver (ou escolher) seus próprios sistemas e softwares para tramitação destes processos, isso fez com que a diversidade de sistemas e formatos dificultasse a consolidação de tais informações.

Esta dificuldade se potencializa pois na justiça brasileira estamos falando de aproximadamente 14.799 unidades judiciárias, em seus diversos ramos de justiça, espalhadas por 27 tribunais de justiça estadual, 24 tribunais de justiça do trabalho, 5 tribunais de justiça federal, além de outros ramos como justiça eleitoral, criminal e militar.



Fonte: Justiça em Números (panorama do poder judiciário)

Considerando que cada tribunal pôde escolher e desenvolver seu próprio sistema de tramitação de processos a unificação destes dados mesmos é um problema constante.

Mas o problema central de acesso a estes dados de forma massiva é a não existência de APIs de tais tribunais, o que reflete na necessidade de coleta de dados públicos através de alternativas como *webscrappers* via scripts automatizados (“robôs de coleta”).

Delimitação do Problema

Considerando este cenário de coleta de dados complexo, da diversidade dos tribunais e da ausência de APIs oficiais para coleta massiva de dados, escolhemos delimitar a análise dos dados ao Tribunal de Justiça do Estado de São Paulo (TJSP).

O TJSP é o tribunal de maior volume do brasil, tanto em numero de processos em tramitação quanto em valores de despesa e numero de magistrados e servidores consequentemente[2].

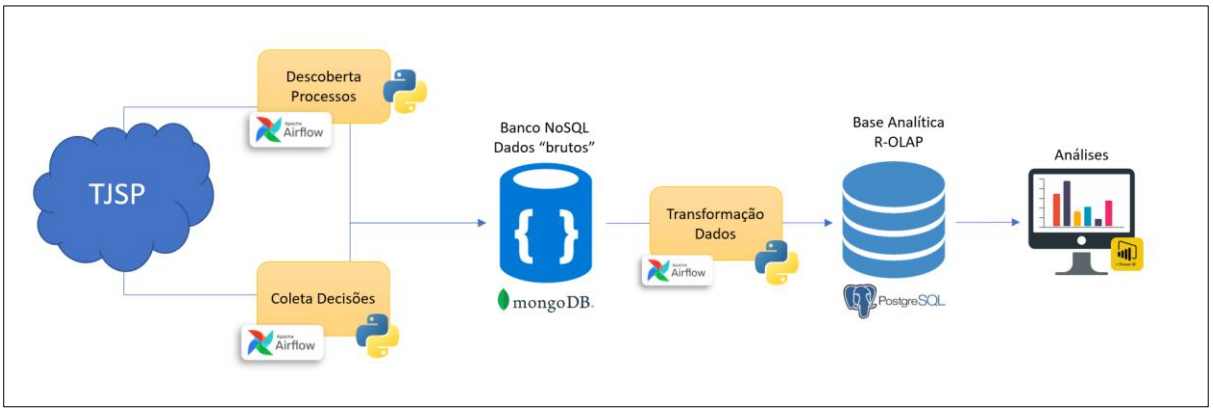
Além de ser o maior tribunal em termos de volume também é em termos de qualidade e acesso aos dados públicos, permitindo a consulta processual de diversas formas (por nome, documentos da parte, acesso a decisões de 1ª e 2ª instâncias etc.) enquanto outros tribunais do Brasil não oferecem as mesmas possibilidades.

Considerando este cenário resolvemos então considerar a coleta de dados apenas do TJSP.

E para realizar a coleta, armazenamento, processamento e apresentação das análises dos dados públicos do TJSP, desenhamos um pipeline de dados contendo diversas etapas e tecnologias, como veremos no próximo tópico.

Arquitetura do Pipeline

O diagrama abaixo apresenta o pipeline, seus componentes e suas etapas.



Fonte: do autor.

O pipeline foi considerado para realizar as seguintes tarefas:

1. Descoberta de processos: Identificar processos judiciais da empresa pelo seu CNPJ.
2. Coleta de Decisões de 1º Grau (Sentença) dos processos coletados.
3. Armazenamento de dados do processo em banco de dados NoSQL (MongoDB).
4. Transformação de Dados e preparação para análise.
5. Armazenamento dos dados analíticos em banco relacional, utilizando modelagem R-OLAP

[2] Fonte: Relatório Justiça em Números: https://www.cnj.jus.br/wp-content/uploads/2022/09/justica-em-numeros-2022.pdf

6. Análise e Visualização de dados através de ferramenta “self-service B.I.” (Power BI Desktop).

Nos tópicos a seguir detalharemos cada etapa e cada componente do pipeline arquitetado para esta solução.

Coleta de Dados

Na etapa de coleta de dados optamos por utilizar scripts *Python* para realizar o *webscrapping* dos processos na pagina do tribunal. Desenvolvemos dois scripts, um para a descoberta de processos e outro para a coleta de decisões dos processos identificados.

Descoberta de Processos

O primeiro passo do pipeline é a descoberta de processos dado um CNPJ. Para isso o script acessa a página do tribunal de busca de processos e pesquisa pelo CNPJ da empresa em questão. A busca pode ser realizada para todos os foros de uma só vez, porém o tribunal limita o retorno em 1000 processos, portanto o script foi desenvolvido considerando a busca por foro dado uma lista inicial de foros (cuja os códigos foram baixados de uma lista oficial do próprio TJSP).

Durante a busca de processos, o tribunal pode retornar algumas situações:

- Não foram encontrados processos (neste caso significa que para o CNPJ específico e foro específico não há processos)
- Retornar apenas um processo, e neste caso a página já exibe informações detalhadas do processo.
- Retornar mais do que um processo, e neste caso a página de retorno traz uma lista de processos e a paginação a cada 25 processos.

Utilizando a biblioteca *beautifulsoap* para realizar a tratativa de dados, os dados do processo são tratados e armazenados em uma *collection* do banco de dados NoSQL (MongoDB).

Coleta de Decisões

Decisões de 1ª Instância são os julgamentos dos processos, e conforme já comentado no início do trabalho, o conteúdo destas decisões é um dado público, bem como algumas informações relevantes como data da decisão e o magistrado que a proferiu (juiz).

Para a coleta de decisões, o script *Python* busca os processos na *Collection* através de um parâmetro específico e através do número do processo ele acessa uma página específica do tribunal de decisões. Neste script há uma tratativa caso não retorne nenhuma informação, algo comum pois alguns processos ainda não tiveram julgamento.

Também tratamos e estruturamos os dados das decisões utilizando a biblioteca *beautifulsoap* e carregamos os dados das decisões em uma *collection* específica no banco de dados NoSQL (mongo DB).

Esta etapa e coleta de decisões é uma etapa importante para ser executada de forma recorrente a fim de buscar as decisões de processos já coletados.

Orquestração e execução dos scripts

Para a execução dos scripts de forma recorrente optamos por utilizar a ferramenta Apache Airflow. O Airflow é uma plataforma de orquestração e execução de workflows baseada em *Python*, utilizando os conceitos de DAGs (Directed Acyclic Graphs), o que a torna uma ferramenta poderosa para executar códigos Pyhton de forma agendada e com diversas etapas dentro de um mesmo workflow.

A decisão de utilizar o *Airflow* foi baseada nos seguintes pontos:

- É uma plataforma *opensource* mantida pela comunidade Apache
- É uma ferramenta construída com framework *Python*, mesma linguagem utilizada para os scripts de *webscrapping*.
- Possibilidade de conexão com diversas fontes e destinos de dados, possibilitando a utilização de diversas tecnologias no pipeline (como é nosso caso).
- Comunidade ativa e ampla utilização da ferramenta, isso auxilia na manutenção e possível correção de problemas.
- Plataforma focada em pipelines massivos (batch) que é o caso do nosso pipeline.

Armazenamento de Dados

Considerando as etapas do nosso pipeline entendemos que precisaremos de duas abordagens distintas para o armazenamento das informações.

Uma base para o armazenamento do dado semiestruturado em sua forma bruta, considerando a característica textual da estrutura do processo judicial, será necessária uma base que favoreça a tratativa deste tipo de dados.

A outra base a ser considerada para armazenamento será uma base analítica, onde os dados serão armazenados em forma de tabelas, utilizando a modelagem dimensional para favorecer a análise utilizando ferramentas de visualização de dados.

Armazenamento dados brutos

Os dados dos processos são em sua maioria informações categorias e textuais, com variações em termos de atributos e conteúdos a depender de seu estágio, natureza e tribunal em tramitação.

Devido a estas características escolhemos utilizar um banco de dados NoSQL orientado a documentos para armazenar os dados brutos, ou seja, recém coletados dos sites dos tribunais.

O armazenamento dos dados com bancos de dados orientados a documentos favorece a persistência de dados textuais, processos com atributos diferentes entre si, processos contendo atributos aninhados como decisões e andamentos e o armazenamento de textos longos como julgamentos e texto das movimentações processuais.

Além disso, a característica “schemaless” deste tipo de banco de dados simplifica o processo inicial e a evolução da coleta de dados via webscrapping.

Como tecnologia selecionamos o banco de dados NoSQL MongoDB, este por ser uma das tecnologias mais utilizadas e por oferecer diversas opções de utilização em ambiente de nuvem, como AWS, Azure, Google além de sua própria oferta de nuvem, além disso também é um software de código aberto, podendo ser utilizado sem a necessidade de licenciamento.

Transformação dos dados

A fim de facilitar o acesso aos dados sumarizados e analíticos, estruturamos um processo de transformação de dados que consome os dados do banco de dados NoSQL e aplica uma série de tratativas.

Entre elas, uma podemos destacar que é o processo de interpretação do resultado da decisão, indicando se o processo foi procedente, improcedente, procedente em parte, extinto ou se houve algum acordo homologado pelo tribunal.

Neste momento, para simplificar o pipeline, optamos por aplicar algumas regras utilizando regex (regular expressions) para sintetizar as decisões de acordo com seu resultado.

Após este processamento, informações adicionais são incluídas nas decisões na própria base NoSQL.

Este processo também é realizado através de um script Python que pode ser executado na plataforma Apache Airflow.

Armazenamento dados analíticos

Após a transformação e enriquecimento dos dados, os dados analíticos são carregados em um banco de dados relacional (PostgreSQL), isso para que facilite o cruzamento e a visualização destes dados através de ferramentas de visualização.

Este armazenamento irá permitir a modelagem dos dados de forma dimensional, utilizando o modelo “star-schema” e potencialmente evolução dos data marts inserindo novas visões como decisões de 2ª instância e outras análises referentes ao mesmo conceito.

Escolhemos para esta camada a utilização de banco de dados PostgreSQL por ser um banco de dados relacional open source que favorece a análise de dados, pois diversas ferramentas de análise de dados já possuem conectores disponíveis para este banco de dados, além de

permitir a utilização da linguagem SQL o que acelera a análise de dados por parte dos analistas de dados.

Além disso existem diversas opções de utilização deste banco em nuvens dos principais players de mercado.

Análise e Visualização de Dados

Para a análise e visualização de dados optamos por uma ferramenta "self-service BI" que facilita a análise exploratória bem como a criação de dashboards para a apresentação de tais informações.

Como em um primeiro momento a análise de dados será de cunho exploratório, ou seja, não temos certeza de quais indicadores e informações podemos extrair, o uso de uma ferramenta que acelera a visualização dos dados sem codificação pesou na decisão da escolha.

Escolhemos a plataforma Power BI por ser uma plataforma gratuita (versão Desktop), ser uma ferramenta líder de mercado (conforme relatório do Gartner[3]) e permitir a visualização dos dados de maneira ágil pois já possui conector nativo com a base de dados PostgreSQL.

Além disso a plataforma Power BI está sendo amplamente utilizada no mercado, o que favorece a curva de aprendizado de seus usuários.

Posteriormente gráficos e análises poderão ser gerados em outras tecnologias, visto que os dados analíticos serão tratados e persistidos no banco de dados PostgreSQL e não em etapas de transformação de dados existentes na ferramenta Power BI.

Prova de Conceito do Pipeline

Além da arquitetura desenhada acima, realizamos o desenvolvimento de algumas etapas do pipeline como uma "prova de conceito" para avaliação e validação da arquitetura proposta.

[3] Fonte: Quadrante Mágico do Gartner para soluções de B.I. https://info.microsoft.com/ww-landing-2022-gartner-mq-report-on-bi-and-analytics-platforms.html

Abaixo um exemplo de tela de retorno do TJSP:

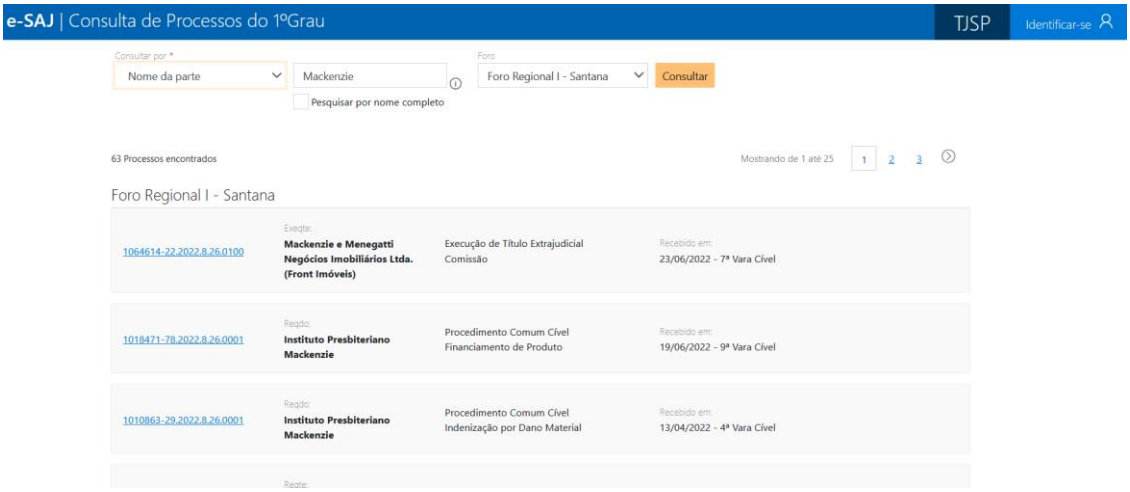


Imagem: exemplo de tela de pesquisa do TJSP

Abaixo algumas imagens ilustrando alguns trechos do pipeline:

```python
def scrapping_foro(foro):
    # Monta a URL de Busca
    url_busca = url_pesquisa + "&cdForo=" + str(foro)

    # Tenta fazer o request
    try:
        html = requests.get(url_busca).content
    except:
        print("Não foi possível carregar a URL")

    # Faz o parser do HTML
    soup = BeautifulSoup(html, 'html.parser')

    # Se retornou muito ou não retornou resultados, a página exibe mensagem
    mensagem = soup.find('td',attrs={'id':'mensagemRetorno'})

    # Em alguns casos, se tive um processo só, ele retorna o processo
    processo_unico = soup.find('span',attrs={'id':'numeroProcesso'})

    # Caso retorne mais do que um processo, ele mostra a quantidades (Util para saber as páginas)
    qtde_processos = soup.find("span",attrs={'id':'contadorDeProcessos'})
```

Imagem: trecho da função webscrapping do foro

```python
def carrega_processo(linha):
    json_str = '{"processo":' + json.dumps(linha) + '}'
    col_processos.find_one_and_update({'processo.nro_processo':linha['nro_processo']},{'$set':json.loads(json_str)},upsert=True)
```

Imagem: trecho da função que carrega os dados no banco NoSQL

```
dados = []

for proc in procs:
    url = "https://esaj.tjsp.jus.br/cjpg/pesquisar.do?conversationId=&dadosConsulta.pesquisaLivre=&tipoNumero=UNIFICA

    try:
        html = requests.get(url).content
    except:
        pass
    soup = BeautifulSoup(html, 'html.parser')

    aviso = soup.select_one('div:-soup-contains("Não foi encontrado nenhum")')

    if aviso:
        print("Processo: " + proc + " sem resultados")

    else:
        print("Processo: " + proc + " com resultados")
        classe = pega_valores(soup,'Classe')
        assunto = pega_valores(soup,'Assunto')
        magistrado = pega_valores(soup,'Magistrado')
        comarca = pega_valores(soup,'Comarca')
        foro = pega_valores(soup,'Foro')
        vara = pega_valores(soup,'Vara')
        data_disp = pega_valores(soup,'Data de Disponibilização')
        texto_decisao = soup.find('div',attrs={'align':'justify','style':'display: none;'}).text.strip()

        dados.append([proc,classe,assunto,magistrado,comarca,foro,vara,data_disp,texto_decisao])
```

Imagem: trecho da função que busca decisões

Abaixo algumas imagens dos dados no banco NoSQL (MongoDB).

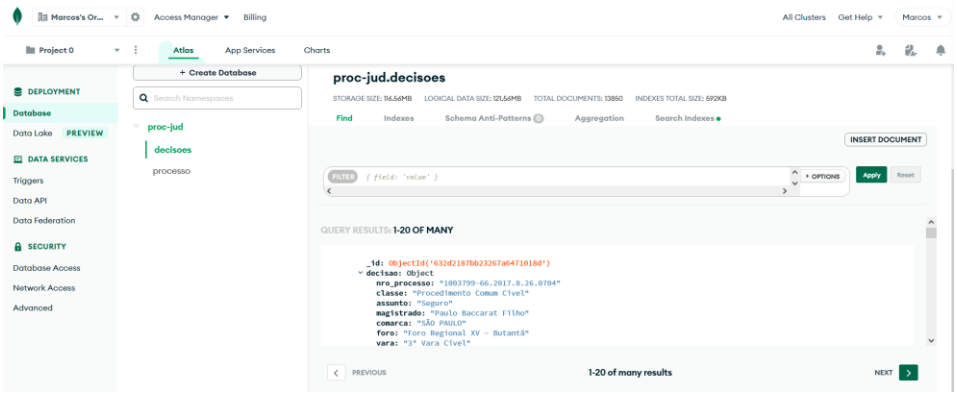


Imagem: tela de collections do banco MongoDB (visualização de documento decisão)

Imagem: decisão com resultado sintetizado na chave “resultado”.

Abaixo imagens do processo de tratamento e transformação de dados:

Tags "Parcialmente Procedente"

```
col_decisoes.update_many(
    {'$and': [
        {'decisao.texto_decisao':{'$regex':"( PARCIALMENTE PROCEDENTE)"}},
        {"decisao.resultado": {"$exists": False}}
    ]},
    {'$set':{'decisao.resultado':'Parcialmente Procedente'}}
)


col_decisoes.update_many(
    {'$and': [
        {'decisao.texto_decisao':{'$regex':"( Parcialmente Procedente)"}},
        {"decisao.resultado": {"$exists": False}}
    ]},
    {'$set':{'decisao.resultado':'Parcialmente Procedente'}}
)
```

Imagem: trecho do código que sintetiza a decisão com base no texto do julgamento.

Abaixo imagens dos dados no banco Analítico (PostgreSQL) na nuvem Google.

```
3   select
4       cod_empresa,
5       count(pr.nro_processo) as "Qtde Processos",
6       count(dc.nro_processo) as "Qtde Decisões"
7   from
8       public.dw_processos pr
9       left join public.dw_decisoes dc on pr.nro_processo = dc.nro_processo
10  group by
11      cod_empresa
```

Results | Messages

	cod_empresa	Qtde Processos	Qtde Decisões
1	EMS	91	0
2	TOKIOMARINE	2832	1192
3	*NULL*	3	0
4	AZUL	5682	1
5	MLUIZA	1775	0
6	ALLIANZ	4315	1991
7	MAPFRE	5281	2336
8	N/I	38	0
9	GOL	4195	2
10	TAM	6239	2
11	SULAMERICA	3317	1866
12	PORTOSEGURO	14436	6970

Imagem: consulta no banco de dados PostgreSQL com base nos processos e decisões

Abaixo imagem do painel de visualização dos dados criados em Power BI Desktop

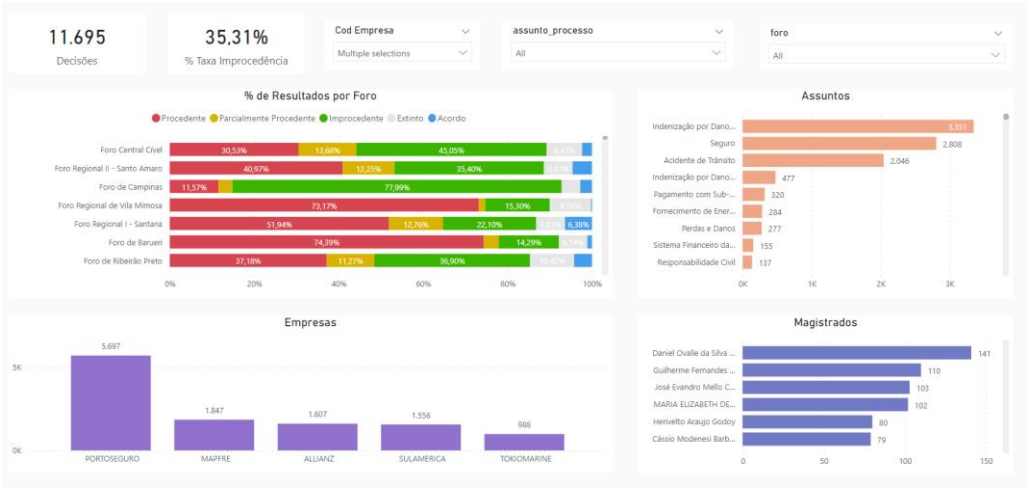


Imagem: Painel de visualização criado com dados das decisões dos tribunais

Todos os códigos estão disponíveis em: https://github.com/marcos-speca/mestrado-cavdados-projeto

Limitações do Trabalho

O objetivo deste trabalho foi estruturar um pipeline de dados para a coleta de informações dos tribunais de justiça, especificamente do tribunal de justiça do estado de São Paulo e criar mecanismos para analisar processos de empresas do mesmo seguimento.

Entendemos que o objetivo foi cumprido, no entanto vale ressaltar algumas limitações:

- A coleta considerou apenas o tribunal de justiça de São Paulo, e para as empresas que possuem um volume grande de processos faz sentido expandir a análise para os demais tribunais de justiça do brasil.

- Na prova de conceito não foi possível implantar a ferramenta apache airflow, portanto mesmo entendendo que a ferramenta é adequada as necessidades expostas análises futuras de sua viabilidade devem ser realizadas.

- No processo de transformação de dados apenas algumas transformações simples foram aplicadas podendo ser melhoradas através de modelos classificadores para identificar melhor o resultado das decisões.

- Coletamos dados apenas de algumas empresas do segmento de Seguros, e ainda sim as comparações realizadas devem ser feitas com cuidado pois mesmo se tratando de empresas do mesmo segmento os processos podem ser diferentes entre si.

Conclusão

O objetivo de desenhar a arquitetura de pipeline de dados para a coleta, processamento e visualização dos dados de processos judiciais do Tribunal de Justiça de São Paulo foi concluído com sucesso.

Foi possível validar a utilidade de scripts de webscrapping, o armazenamento de dados brutos, as etapas de tratativas de dados, o armazenamento de dados analíticos a sua visualização em ferramentas de dashboards.

Como etapas futuras para dar continuidade neste desenvolvimento precisaríamos construir um servidor Apache Airflow, adaptar os scripts ao framework Airflow e publicar os dashboards para que o processo todo seja automatizado.