

Faculdade

XPe



RELATÓRIO

PROJETO
APLICADO

PÓS-GRADUAÇÃO

XP Educação
Relatório do Projeto Aplicado

Série temporal de dados das maiores empresas da bolsa norte americana

IGOR BEZERRA MENDES

Orientador(a): Professor Davidson Oliveira

28/10/2024



IGOR BEZERRA MENDES

XP EDUCAÇÃO

RELATÓRIO DO PROJETO APLICADO

SÉRIE TEMPORAL DE DADOS DAS MAIORES EMPRESAS DA BOLSA NORTE AMERICANA

Relatório de Projeto Aplicado
desenvolvido para fins de conclusão do
curso de pós-graduação em Engenharia e
Arquitetura de Dados.

Orientador (a): Professor Davidson
Oliveira

SÃO PAULO

28/10/2024



Sumário

1. CANVAS do Projeto Aplicado	5
1.1 Desafio	5
1.1.1 Análise de Contexto	5
1.1.2 Personas	8
1.1.3 Benefícios e Justificativas	10
1.1.4 Hipóteses	12
1.2 Solução	13
1.2.1 Objetivo SMART	13
1.2.2 Premissas e Restrições	13
1.2.3 Backlog de Produto	14
2. Área de Experimentação	16
2.1 Sprint 1	16
2.1.1 Solução	16
• Evidência do planejamento:	16
• Evidência da execução de cada requisito:	16
• Evidência dos resultados:	18
2.1.2 Lições Aprendidas	24
2.2 Sprint 2	25
2.2.1 Solução	25
• Evidência do planejamento:	25
• Evidência da execução de cada requisito:	25
• Evidência dos resultados:	32
2.2.2 Lições Aprendidas	38
2.3 Sprint 3	38
2.3.1 Solução	38
• Evidência do planejamento:	38
• Evidência da execução de cada requisito:	39
• Evidência dos resultados:	40
2.3.2 Lições Aprendidas	41
3. Considerações Finais	42
3.1 Resultados	42
3.2 Contribuições	42
3.3 Próximos passos	43



1. CANVAS do Projeto Aplicado

Figura conceitual, que representa todas as etapas do Projeto Aplicado.



1.1 Desafio

1.1.1 Análise de Contexto

Em um cenário fictício, estamos trabalhando em uma corretora de valores com clientes Institucionais como fundos de investimentos com grandes valores em patrimônio líquido e clientes individuais, geralmente clientes que possuem um maior valor podem possuir o interesse em investir em empresas de mercados fortes como o mercado dos estados unidos.

Na empresa, existe um cenário frequente de distribuição de recomendações por parte dos analistas de investimento de empresas da bolsa norte americana, sendo a principal custodiante a NYSE (New York Stock Exchange) e índices são consultados como o SP&500, composto pelas 500 maiores empresas dos estados unidos.

A extração desses dados pelos analistas é feita de forma manual em consulta pública na internet. O que traz uma dificuldade para os analistas de obter informações consolidadas das empresas além da informação do valor da ação no momento. Exemplos: números de funcionários, dados cadastrais dessas empresas, seu valor de mercado e histórico de recompensação de dividendos, uma modalidade de

remuneração do investidor muito atrativa no longo prazo, bem como o retorno histórico dessa empresa de forma rápida. Sendo está a principal causa do problema da empresa e oportunidade de melhoria.

Propõe-se, para o embasamento da decisão e diante do cenário uma fonte de dados única da verdade através de um banco de dados relacional em um sistema de armazenamento corporativo, que seja de possibilidade de uso para toda a empresa e de fácil acesso para esses usuários, sendo possível o acesso destes por uma simples página da web.

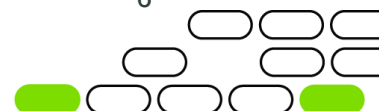
O Time de engenharia atuante no projeto sugere a criação de um relatório público (no ambiente da empresa) com intuito de usar análises comumente realizadas pelos analistas de forma automatizada, gerando uma melhoria no dia a dia dessas pessoas, reduzindo o tempo delas em pesquisa, estudos que podem ser automatizados em retorno financeiro para empresa, uma vez que esses analistas podem se alocar em atividades mais rentáveis para a corretora.

Tecnicamente, propõe-se o uso de ferramentas de SGBD para a equipe técnica e a equipe de negócio, este sendo contemplado pelo PostgreSQL, que possui a capacidade de uso administrativo com escalabilidade e permissionamento de usuários, bem como a capacidade de incrementação de dados por uso de `staging` e `upserts`. Alinhado a isso a ferramenta também possui uma versão web em que o usuário simplesmente logará com seu usuário e senha. Eliminando instalação e configuração do seu ambiente local.

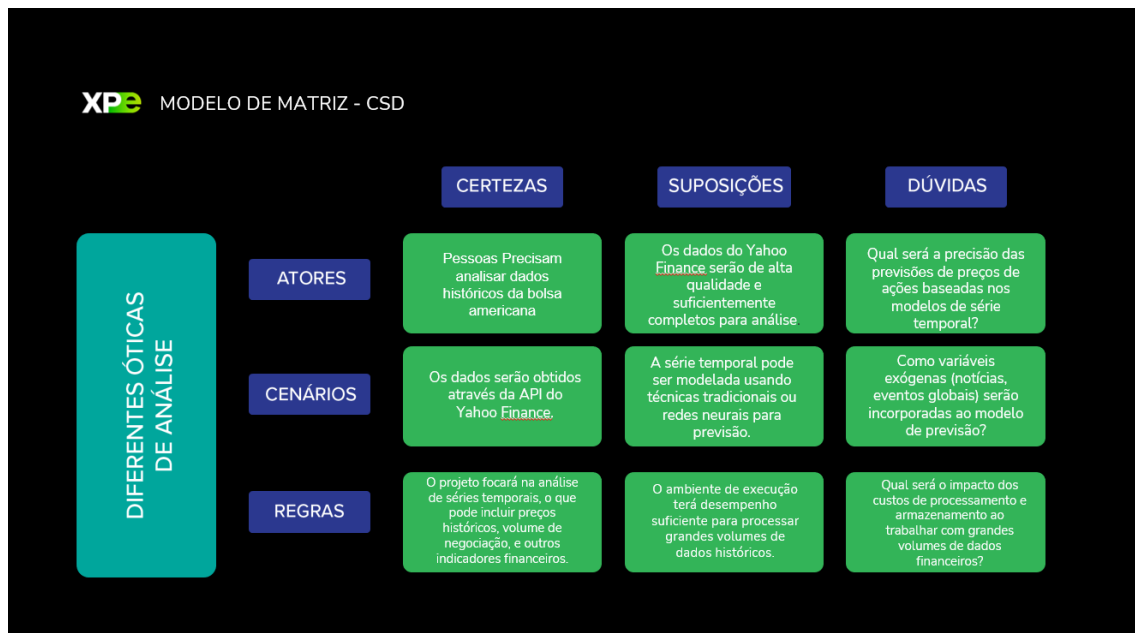
A série temporal dos dados, será visualizada pela área de negócios tanto no SGBD no postgres bem como em um relatório usando o Power BI em um relatório público no ambiente local da empresa que será disponibilizado em um link de acesso de forma simples para área de negócios, com as principais análises realizadas manualmente a partir dos dados.

Quais ferramentas devem ser utilizadas?

A fonte de dados única da verdade será a API gratuita do yahoo finance, que utiliza intrinsecamente os dados de fontes confiáveis como a própria NYSE e índices compostos como S&P500 e NASDAQ. Além de ser gratuita, bem como as demais licenças dos produtos acima gerando nenhum custo inicial para empresa. Os dados pretendem ser atualizados sempre a meia noite, pois a necessidade em si é de dados históricos de forma incremental e não em tempo real (NRT).



A figura abaixo apresenta conceitualmente a matriz de certezas suposições e dúvidas diante do cenário fictício criado:



Destaca-se na perante a matriz CSD a definição de uma fonte única de dados sendo a fonte da verdade e de algumas possibilidades como desafios intrínsecos do projeto em si como a confiabilidade da informação e a fácil distribuição para as pessoas.

De forma objetiva a análise do contexto do problema vem a seguir:



1.1.2 Personas

Diante do cenário, temos as seguintes personas de forma objetiva: um engenheiro de dados que irá atuar dentro da arquitetura do projeto, responsável por realizar as tarefas do projeto e um analista de investimento que é responsável por esse tipo de análise da corretora. Personificando o Product Owner da entrega.

Persona: João Silva

Nome Fictício: João Silva

Idade: 45 anos

Profissão: Analista de Investimentos Sênior em uma corretora de valores

Características Comportamentais: João é metódico, gosta de ter todas as informações antes de tomar decisões e busca soluções eficientes para otimizar o tempo de análise.

Pessoal: Casado, pai de dois filhos, gosta de acompanhar tendências de mercado no tempo livre e está sempre em busca de aperfeiçoamento técnico.

Social: Participa de eventos do mercado financeiro e grupos online de discussão sobre investimentos.

Intelectual: João tem vasta experiência no mercado financeiro, sendo especializado em análise de ações e fundos de investimentos, com forte domínio de ferramentas como Excel, Power BI e bases de dados financeiros.

Profissional: Seu foco é otimizar o tempo de análise de dados para entregar recomendações de investimento rápidas e confiáveis. Ele busca ferramentas que o ajudem a substituir processos manuais por sistemas mais automatizados e integrados.

Para o Engenheiro de Dados:

Persona: Marcos Ribeiro

Nome Fictício: Marcos Ribeiro

Idade: 32 anos

Profissão: Engenheiro de Dados

Características Comportamentais: Marcos é analítico, organizado e focado em otimizar processos de ETL e pipelines de dados. Valoriza soluções escaláveis e bem estruturadas.



Pessoal: Solteiro, gosta de resolver problemas complexos e de manter-se atualizado em relação às novas tecnologias de big data e machine learning.

Social: Participa de conferências e eventos de tecnologia, networking com profissionais da área e fóruns online sobre arquitetura de dados.

Intellectual: Especializado em arquitetura de dados e plataformas de big data, com forte domínio de ferramentas como Spark, Hadoop, bancos de dados relacionais e não relacionais (PostgreSQL, MongoDB), além de integrações com APIs.

Profissional: Seu objetivo é garantir que os dados fluam de forma eficiente e organizada entre as diferentes camadas do projeto, implementando pipelines robustos e escaláveis.

Mapa de Empatia

O que ele pensa e sente?

Acredita que o projeto precisa de uma arquitetura de dados bem estruturada para suportar o crescimento e atender as necessidades de análise de dados de forma ágil.

O que ele escuta

Frequente demanda de criação e automação de pipelines que tragam dados de fontes externas e os armazenem de forma eficiente para análises posteriores.

O que ele fala e faz?

Defende a necessidade de boas práticas de engenharia de dados, incluindo versionamento de dados, documentação e automação de processos para reduzir falhas e retrabalho.

O que ele vê?

Observa o rápido avanço de tecnologias de big data e a crescente demanda por arquiteturas flexíveis que possam lidar com volumes massivos de dados em tempo real.

Quais são seus medos, frustrações e obstáculos?

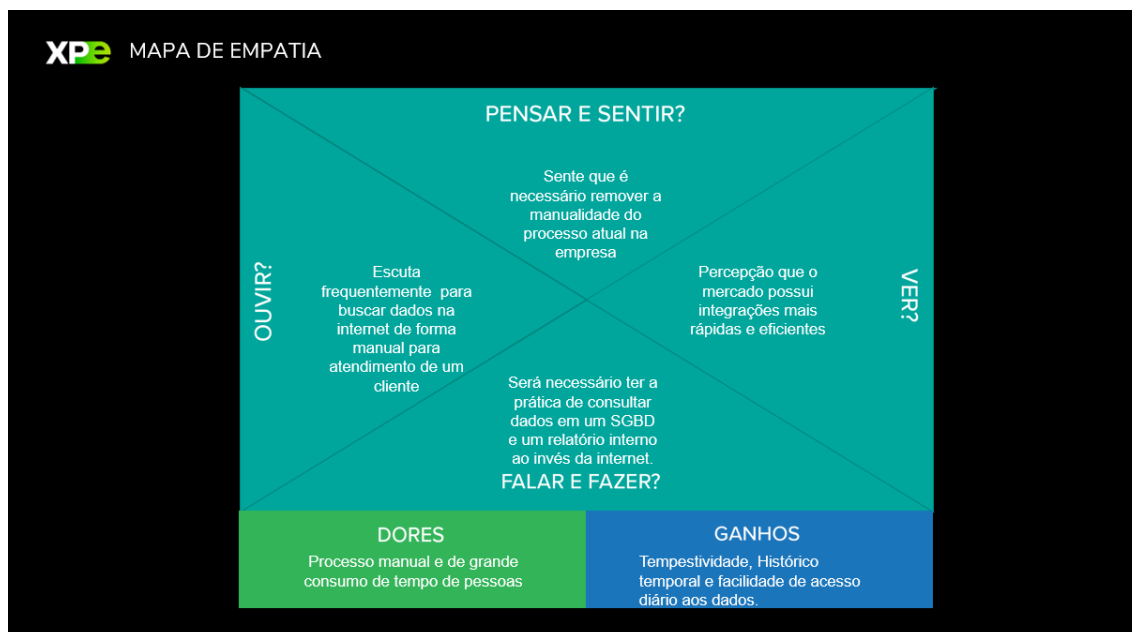
Frustra-se com sistemas legados mal documentados e com a falta de ferramentas adequadas para a automatização de processos complexos.

Quais são suas necessidades?



Marcos precisa de ferramentas que permitam a integração, transformação e armazenamento eficiente de grandes volumes de dados, além de suporte técnico e metodologias ágeis para o desenvolvimento contínuo do projeto.

A Imagem abaixo tem como ideia demonstrar o mapa de empatia dos envolvidos em um breve resumo



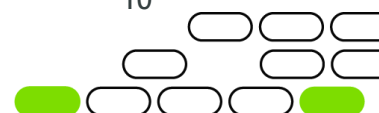
1.1.3 Benefícios e Justificativas

Cenário Atual:

O projeto em questão visa atender a demanda de uma corretora de valores que, atualmente, utiliza processos manuais para acessar dados financeiros públicos, extraí-los e analisá-los de forma fragmentada e demorada. Esses dados são fundamentais para a criação de relatórios de recomendações de investimentos. O processo atual apresenta várias limitações, como o tempo necessário para analisar grandes volumes de dados históricos e a falta de atualização diária automática, o que afeta diretamente a qualidade e agilidade nas tomadas de decisão.

Justificativa para o Desenvolvimento do Projeto:

Com base nas necessidades identificadas e nas dores das personas envolvidas (especialmente do analista de investimentos, que assume o papel de Product Owner), o projeto visa automatizar o processo de extração e análise de dados, integrando soluções tecnológicas que facilitem o acesso, atualização e consulta de informações financeiras.



A proposta de valor é trazer maior eficiência, precisão e praticidade para o processo de análise, além de fornecer ferramentas de automação que permitirão ao analista focar em atividades de maior valor, como a interpretação de dados e a formulação de estratégias de investimento, ao invés de se preocupar com tarefas manuais e repetitivas.

Benefícios Futuros Esperados:

Redução de Custos: A automatização dos processos de extração e análise de dados resultará em uma significativa redução de tempo e esforço manual, diminuindo assim os custos operacionais. Com menos horas sendo gastas em atividades operacionais, a equipe poderá ser direcionada para funções mais estratégicas.

Aumento da Eficiência: O tempo necessário para acessar, tratar e analisar os dados será drasticamente reduzido. Isso permitirá que o analista de investimentos tome decisões mais rapidamente, baseadas em dados sempre atualizados e precisos.

Novas Fontes de Receita: A corretora poderá criar produtos e serviços, como relatórios financeiros sob demanda ou assinaturas premium para análises aprofundadas, aumentando suas fontes de receita.

Impacto Social e Econômico: O projeto terá o potencial de influenciar positivamente o mercado de investimentos, oferecendo informações mais precisas e confiáveis para os investidores, o que pode resultar em melhores tomadas de decisão e, conseqüentemente, em maior geração de valor econômico.

Inovação: A automação, o uso de inteligência de dados e a criação de um banco de dados para facilitar o acesso a séries temporais de grandes empresas norte-americanas são inovações importantes no setor financeiro. Isso permite à corretora oferecer um serviço diferenciado aos seus clientes, criando uma vantagem competitiva frente a concorrentes.

Proposta de Valor:

A proposta de valor do projeto é clara: oferecer praticidade na criação de carteiras de investimento, ao mesmo tempo em que facilita a análise da solidez das empresas, utilizando dados financeiros históricos e atualizados em tempo real. O projeto visa otimizar as operações do analista de investimentos e agregar valor à tomada de decisão, tornando-a mais ágil, precisa e escalável.



Em resumo, o projeto não apenas automatiza tarefas críticas, mas também cria uma infraestrutura que possibilitará o crescimento sustentável e a inovação contínua dentro da corretora de valores.

A Imagem abaixo demonstra a explicação da proposição de valor de forma simplificada



1.1.4 Hipóteses

Automatização das tarefas manuais: A automatização dos processos de análise de dados financeiros será mais eficiente do que a coleta manual, reduzindo significativamente o tempo de processamento e a chance de erros humanos.

Benefícios de uma base de dados centralizada: A criação de uma base de dados para armazenar os dados financeiros permitirá uma análise mais rápida e precisa, além de facilitar o histórico temporal e o acompanhamento das empresas.

Facilidade no uso da ferramenta: A implementação de uma interface web fácil de usar proporcionará maior acessibilidade aos usuários leigos, aumentando a eficiência nas consultas e no uso dos dados financeiros.

Geração automática de relatórios: A automatização da geração de relatórios diários reduzirá o tempo necessário para a criação de carteiras de recomendação de investimentos, permitindo decisões mais rápidas e assertivas.

1.2 Solução

1.2.1 Objetivo SMART

S (Specific - Específico): Fornecer recomendações de investimento por meio de dados embasados em uma fonte única da verdade para atender os clientes individuais e institucionais

M (Mensurable - Mensurável): Gerar relatórios diários de forma automatizada e interativa, para que o usuário possa extrair os seus insights.

A (Attainable - Atingível): Utilizar SGBD (postgres) para armazenar os dados de forma incremental, tendo uma primeira carga de 10 anos incrementadas a cada diárias, bem como a atualização dos indicadores anteriormente armazenados

R (Relevant - Relevante): Melhorar o processo manual de obtenção de informações pelos usuários de negócio

T (Time based - Temporal): Implementar em 3 sprints de duas semanas cada.

1.2.2 Premissas e Restrições

Premissas:

Automação de Processos: A automação reduzirá a necessidade de intervenção manual, minimizando o esforço administrativo.

Impacto: Se a automação não for eficiente, o esforço manual pode aumentar, comprometendo a agilidade do projeto.

Uso de Ferramentas Open-Source: O projeto será desenvolvido com ferramentas open-source para minimizar custos com licenciamento. O código de geração dos dados pretende ser simples com o uso de notebooks para facilitar a sustentação futura.

Impacto: Se houver necessidade de ferramentas pagas, o custo do projeto pode aumentar, tornando-o inviável financeiramente.

Restrições:

Orçamento Limitado: O projeto deve ser concluído com um orçamento reduzido, restringindo o uso de recursos financeiros adicionais.



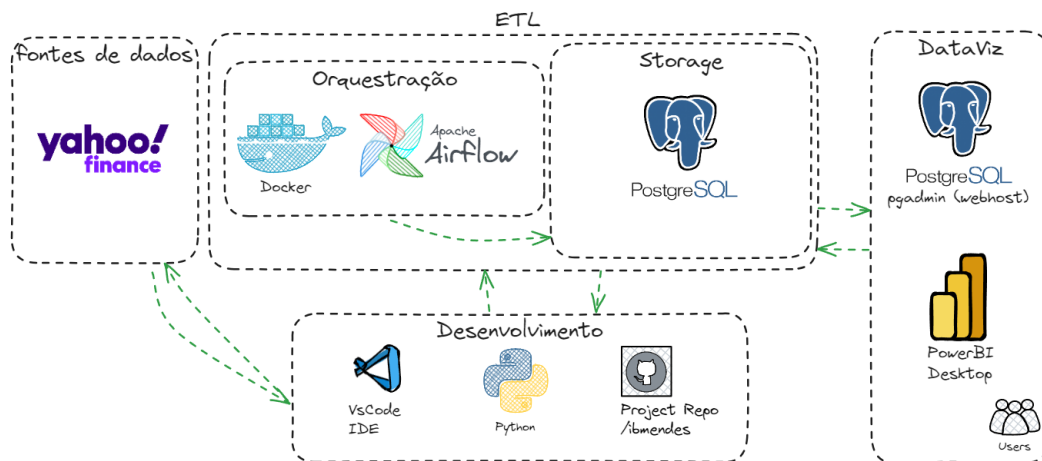
Consequência: A limitação orçamentária pode impactar o tempo de entrega e a qualidade final da solução.

Limitação de Equipe: O número de profissionais envolvidos no desenvolvimento será limitado para reduzir os custos operacionais.

Consequência: A equipe reduzida pode afetar a velocidade de desenvolvimento e a capacidade de resposta às demandas.

1.2.3 Backlog de Produto

Para melhor entendimento do desenho do produto a ser entregue anexa-se abaixo um macro desenho da ideia proposta:



Fonte: Excalidraw

Dentro da ideia de arquitetura proposta, o cronograma de planejamento pretende contemplar as seguintes etapas

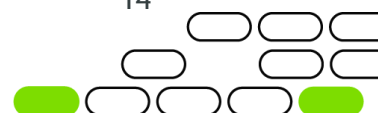
Sprint 1:

Configurar PostgreSQL: Implementar o banco de dados PostgreSQL para armazenar os dados financeiros, bem como a configuração de usuários para acesso. Tabelas e requisitos esperados para armazenamento no banco de dados

Configurar Airflow: Estabelecer o fluxo de trabalho automatizado usando Apache Airflow para execução das tarefas programadas, o projeto pretende executar uma rotina simples a cada 0h todos os dias. Para isso será necessário configurar o ambiente no Docker com a imagem customizada.

Sprint 2:

Configurar Aquisição de Dados da API do Yahoo Finance: Integrar a fonte de dados, API do Yahoo Finance, ao sistema para extrair dados relevantes e análises que são realizadas manualmente pelas pessoas.

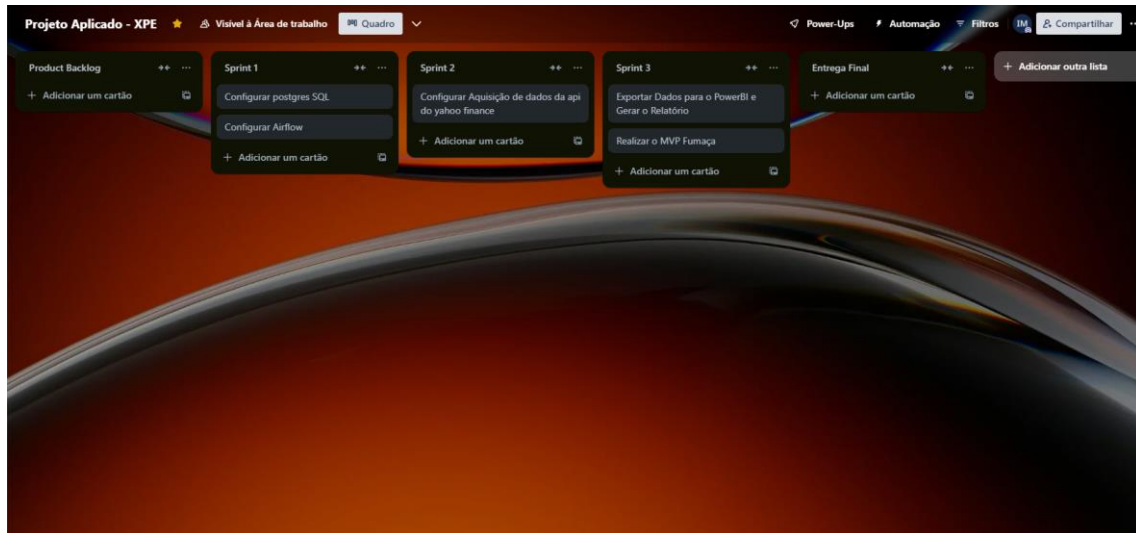


Sprint 3:

Exportar Dados para o Power BI e Gerar Relatório: Criar uma integração entre a base de dados e o Power BI, gerando relatórios financeiros para os usuários finais.

Realizar o MVP Fumaça: Desenvolver uma versão mínima do projeto para garantir que todos os sistemas estão integrados corretamente em testes pré release (entrega final do projeto aplicado).

A imagem a seguir possui o intuito de destacar as etapas na ferramenta orientada.



Fonte: Trello

As atividades terão seus itens movimentados e datas definidas no início das sprints no material a diante.

2. Área de Experimentação

2.1 Sprint 1

Essa sprint teve como objetivo conforme mencionado na seção 1.2.3 de realizar as configurações iniciais dos serviços integrados PostgreSQL como sistema gerenciador de banco de dados e o Apache Airflow como orquestrador das rotinas.

2.1.1 Solução

- Evidência do planejamento:

Definido para primeira sprint (30/09) temos duas entregas planejadas



Fonte: [Trello](https://trello.com/).

- Evidência da execução de cada requisito:

O repositório do projeto foi criado no github, podendo ser acessado pelo repositório https://github.com/ibmendes/paxpe_app/

Docker: para configuração dos requisitos, usaremos o Docker como ambiente de container por trás dos ambientes que criaremos, com as imagens do Airflow e do Postgres. Para posteriormente criarmos o script do python, as DAGs e a criação dos painéis do PowerBI.

Dentre algumas criações destacam-se as principais: Na imagem do Docker, definimos quais as imagens que usaremos de cada dentro do arquivo `paxpe_app/docker-compose.yaml`

1. Imagem do Airflow (Embutido com inicio de um banco postgres)

```

1  version: '3.8'
2
3  x-airflow-common:
4    &airflow-common
5    build:
6      context: .
7      dockerfile: Dockerfile
8    image: custom-airflow:latest
9    environment:
10     &airflow-common-env
11     AIRFLOW__CORE__EXECUTOR: CeleryExecutor
12     AIRFLOW__DATABASE__SQL_ALCHEMY_CONN: postgresql+psycopg2://airflow:airflow@postgres/airflow
13     AIRFLOW__CELERY__RESULT_BACKEND: db+postgresql://airflow:airflow@postgres/airflow
14     AIRFLOW__CELERY__BROKER_URL: redis://:@redis:6379/0
15     AIRFLOW__CORE__FERNET_KEY: ''
16     AIRFLOW__CORE__DAGS_ARE_PAUSED_AT_CREATION: 'true'
17     AIRFLOW__CORE__LOAD_EXAMPLES: 'false'
18     AIRFLOW__API__AUTH_BACKENDS: 'airflow.api.auth.backend.basic_auth,airflow.api.auth.backend.session'
19     AIRFLOW__SCHEDULER__ENABLE_HEALTH_CHECK: 'true'
20     _PIP_ADDITIONAL_REQUIREMENTS: ''
21   volumes:
22     - ./src/airflow-docker/dags:/opt/airflow/dags
23     - ./src/airflow-docker/logs:/opt/airflow/logs
24     - ./src/airflow-docker/config:/opt/airflow/config
25     - ./src/airflow-docker/plugins:/opt/airflow/plugins
26     - ./src/airflow-docker/jars:/opt/airflow/jars

```

2. PGAdmin

```

53
54  pgadmin:
55    image: dpage/pgadmin4:latest
56    environment:
57      PGADMIN_DEFAULT_EMAIL: admin@admin.com
58      PGADMIN_DEFAULT_PASSWORD: admin
59    ports:
60      - "5050:80"
61    depends_on:
62      - postgres
63    volumes:
64      - pgadmin-data:/var/lib/pgadmin
65

```

3. Postgres

```

35
36  services:
37    postgres:
38      image: postgres:13
39      environment:
40        POSTGRES_USER: airflow
41        POSTGRES_PASSWORD: airflow
42        POSTGRES_DB: airflow
43      ports:
44        - "5432:5432"
45      volumes:
46        - postgres-db-volume:/var/lib/postgresql/data
47      healthcheck:
48        test: ["CMD", "pg_isready", "-U", "airflow"]
49        interval: 10s
50        retries: 5
51        start_period: 5s
52      restart: always
53

```

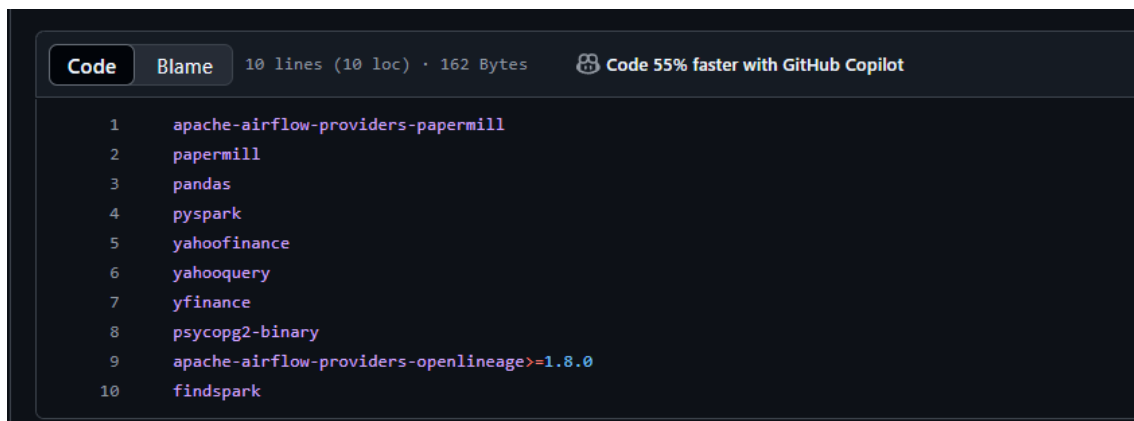


As demais configurações do Docker compose são padrão de sugestão da apache em sua documentação onde customizamos apenas as necessidades. Portanto configurações como o worker, scheduler do airflow se mantem em padrão como sugerido em [Running Airflow in Docker](#)

Bibliotecas e dependências

O Docker possui a possibilidade de colocarmos manualmente quais as libnames que iremos utilizar no provisionamento de recursos. Esses requisitos podem ser encontrados no arquivo criado paxpe_app/requirements.txt

Aqui definimos libnames nativas para uso do spark bem como libs para trabalhar com insert de dados no banco sql (postgres) e demais dependências do projeto

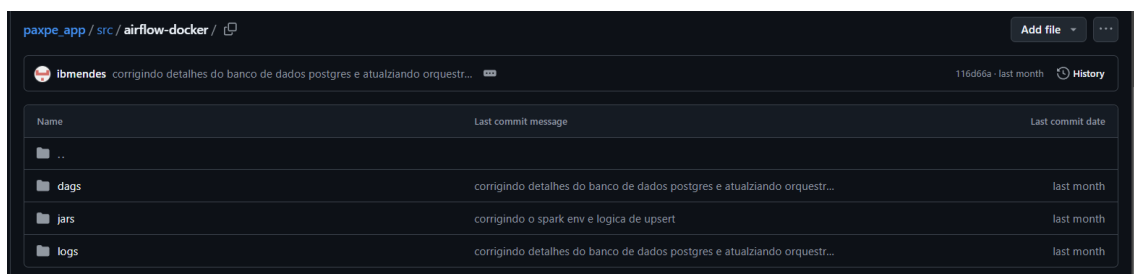


```

1  apache-airflow-providers-papermill
2  papermill
3  pandas
4  pyspark
5  yahoofinance
6  yahooquery
7  yfinance
8  psycpg2-binary
9  apache-airflow-providers-openlineage>=1.8.0
10 findspark
  
```

Estrutura física

O Docker também configura na imagem a divisão de locais para sabermos o que deve ser criado em cada lugar como pode ser visto em “src” pasta raiz para fontes do processo

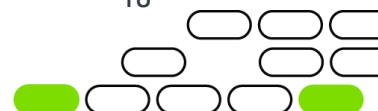


Name	Last commit message	Last commit date
..		
dags	corrigindo detalhes do banco de dados postgres e atualizando orquestr...	last month
jars	corrigindo o spark env e logica de upsert	last month
logs	corrigindo detalhes do banco de dados postgres e atualizando orquestr...	last month

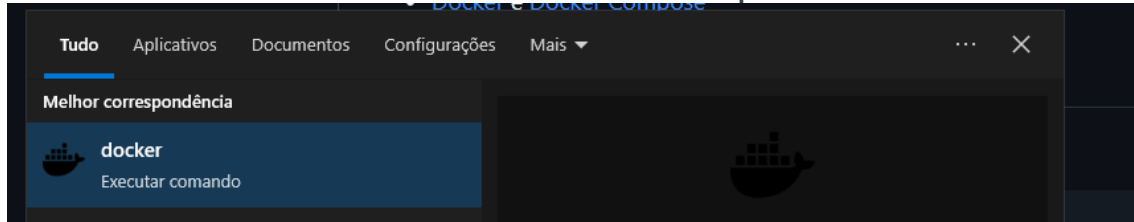
- Evidência dos resultados:

Os passos para executar a entrega dessa sprint estão contidos na Branch main no arquivo readme.md onde podemos ter uma ideia de como executar o projeto e o provisionamento de recursos.

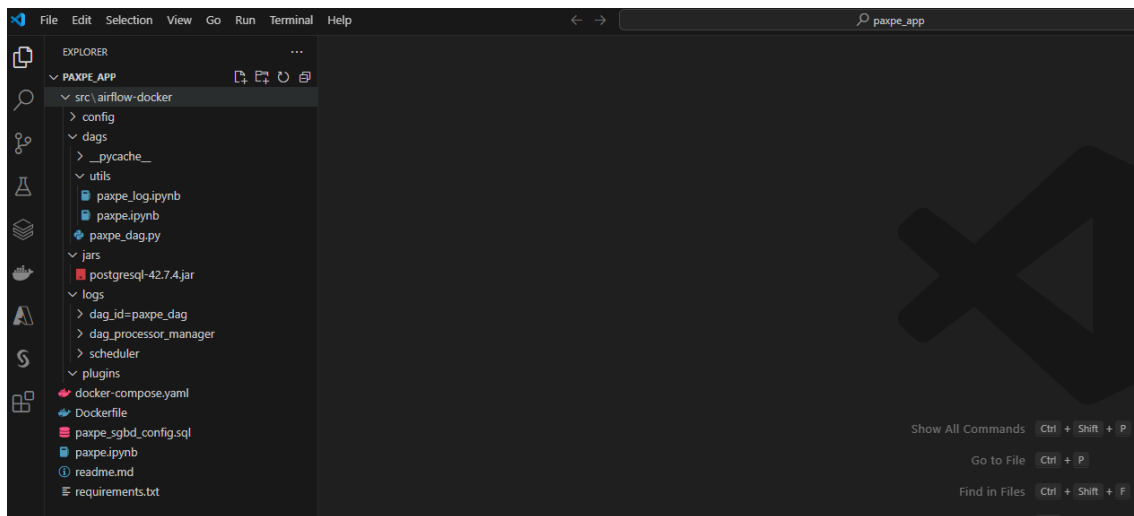
1. Provisionando o Docker



O Docker deve estar rodando no servidor em questão



Pelo vscode, podemos abrir o repositório, caso não tenhamos podemos cloná-lo
 git clone https://github.com/ibmendes/paxpe_app.git

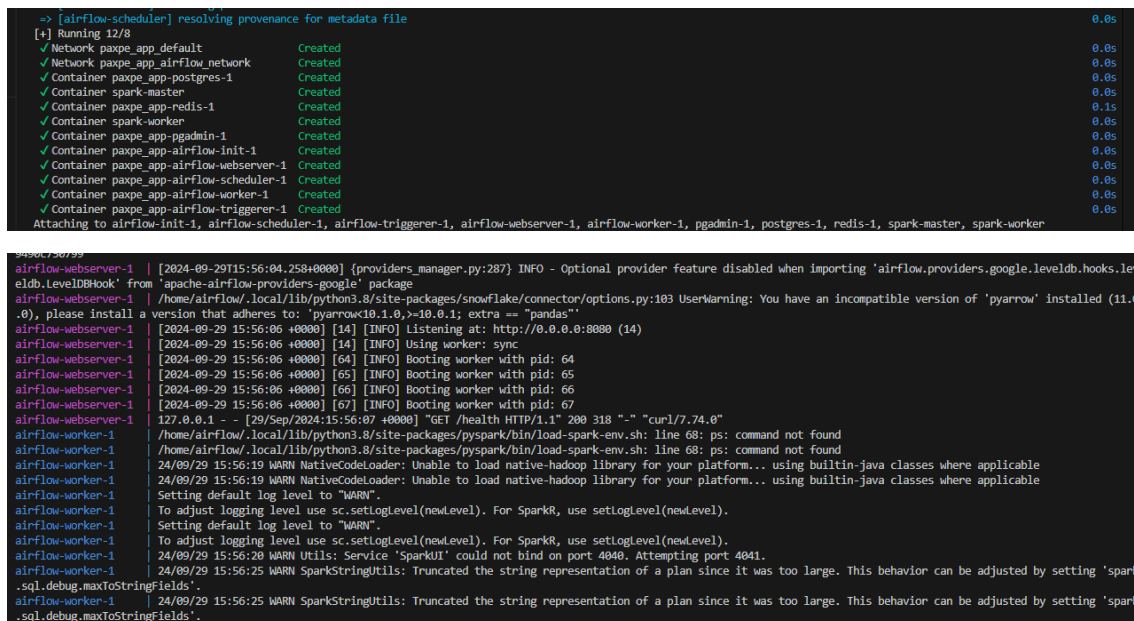


Usando uma linha de comand prompt podemos seguir os passos contido no
 readme e subir a imagem do Docker

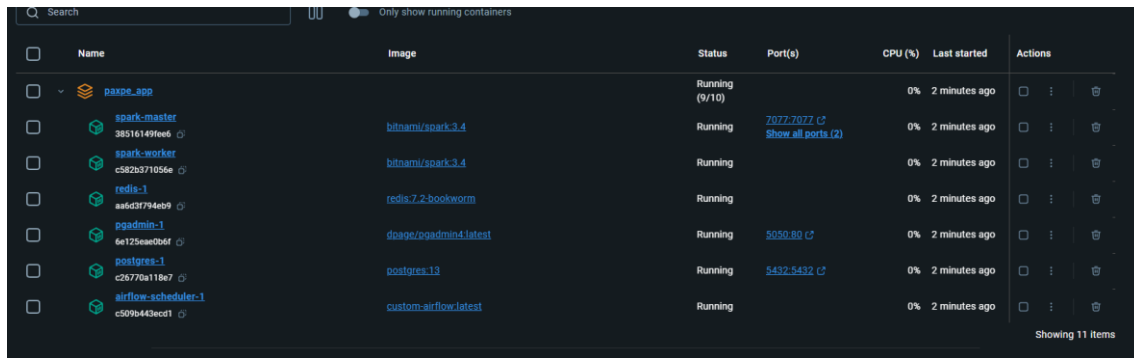
Usando para construir a imagem no Docker:

docker-compose up -build

log de criação da imagem:



Visualização via interface (Docker Desktop)



Name	Image	Status	Port(s)	CPU (%)	Last started	Actions
paxpe_app		Running (9/10)		0%	2 minutes ago	[Stop] [Refresh] [Delete]
spark-master	bitnami/spark-3.4	Running	7077:7077	0%	2 minutes ago	[Stop] [Refresh] [Delete]
spark-worker	bitnami/spark-3.4	Running		0%	2 minutes ago	[Stop] [Refresh] [Delete]
redis-1	redis/7.2-bookworm	Running		0%	2 minutes ago	[Stop] [Refresh] [Delete]
pgadmin-1	dpage/pgadmin4:latest	Running	5050:80	0%	2 minutes ago	[Stop] [Refresh] [Delete]
postgres-1	postgres:13	Running	5432:5432	0%	2 minutes ago	[Stop] [Refresh] [Delete]
airflow-scheduler-1	custom-airflow:latest	Running		0%	2 minutes ago	[Stop] [Refresh] [Delete]

Showing 11 items

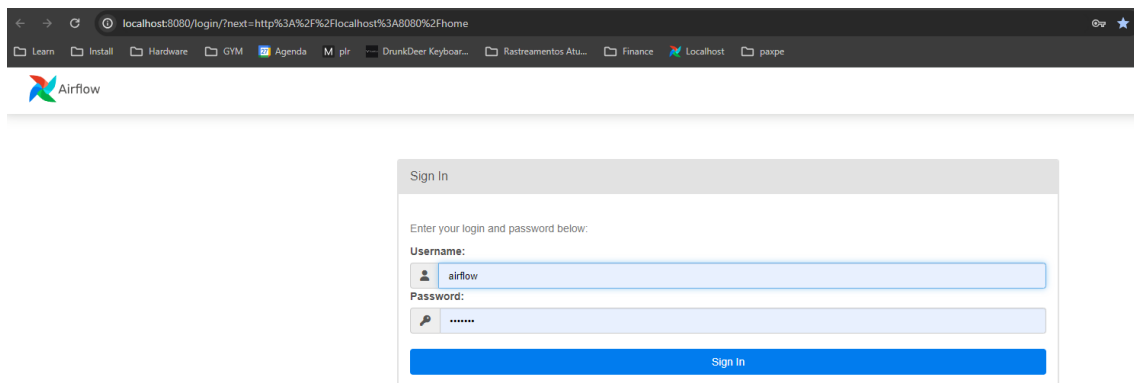
O Docker Compose configura e inicia vários serviços. Aqui estão as instruções para acessar cada recurso

1. Apache Airflow

Descrição: Plataforma para criar, agendar e monitorar workflows de dados.

URL de Acesso: <http://localhost:8080>

Entrando no airflow



localhost:8080/login?next=http%3A%2F%2Flocalhost%3A8080%2Fhome

Airflow

Sign In

Enter your login and password below:

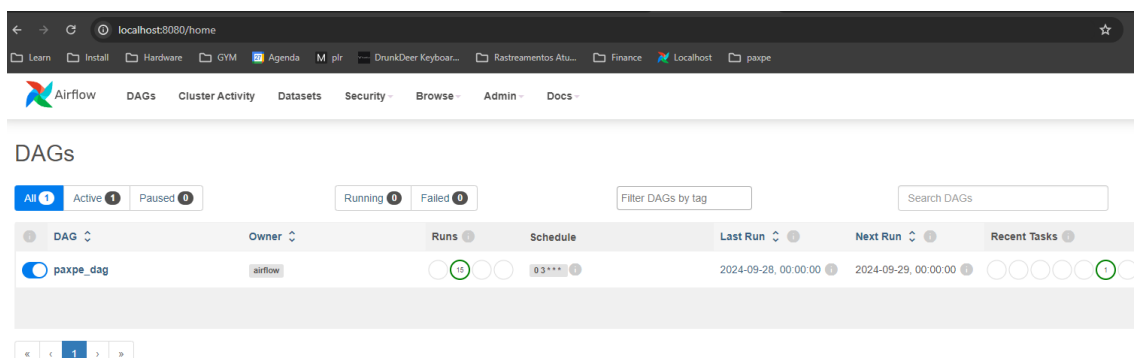
Username:

airflow

Password:

Sign In

Visualização da interface



localhost:8080/home

Airflow DAGs Cluster Activity Datasets Security Browse Admin Docs

DAGs

All 1 Active 0 Paused 0

Running 0 Failed 0

Filter DAGs by tag

Search DAGs

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks
paxpe_dag	airflow	10	03***	2024-09-28, 00:00:00	2024-09-29, 00:00:00	[Task 1] [Task 2] [Task 3] [Task 4] [Task 5]

< 1 >

2. pgAdmin

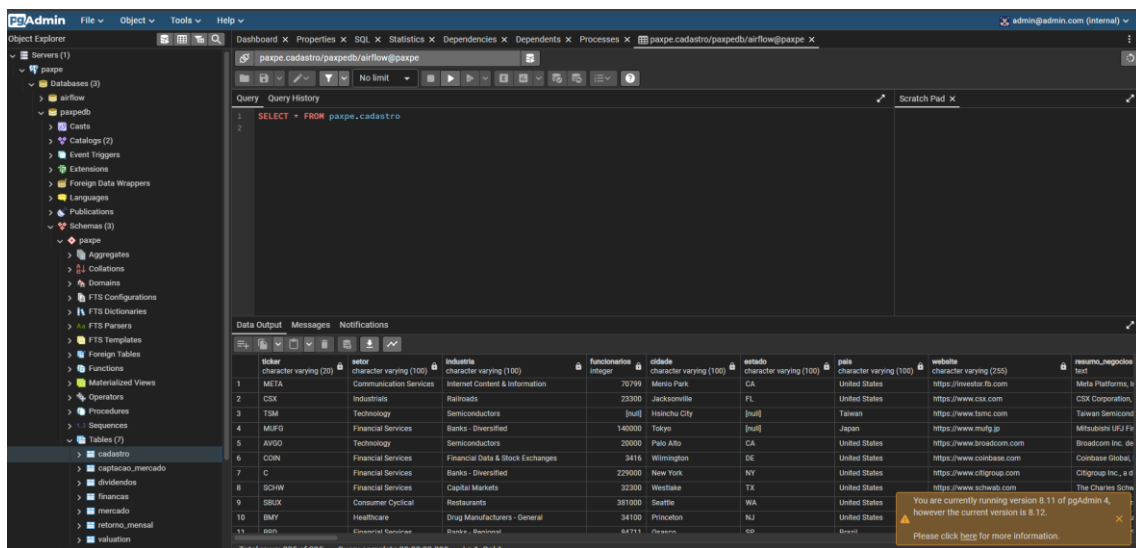
Descrição: Ferramenta para administração e gerenciamento de bancos de dados PostgreSQL.

URL de Acesso: <http://localhost:5050>

visualização da interface:



Após o acesso via credenciais o usuário pode configurar a sua interface e ter visualização aos dados (os dados da foto a seguir estão em amostragem)



3. PostgreSQL

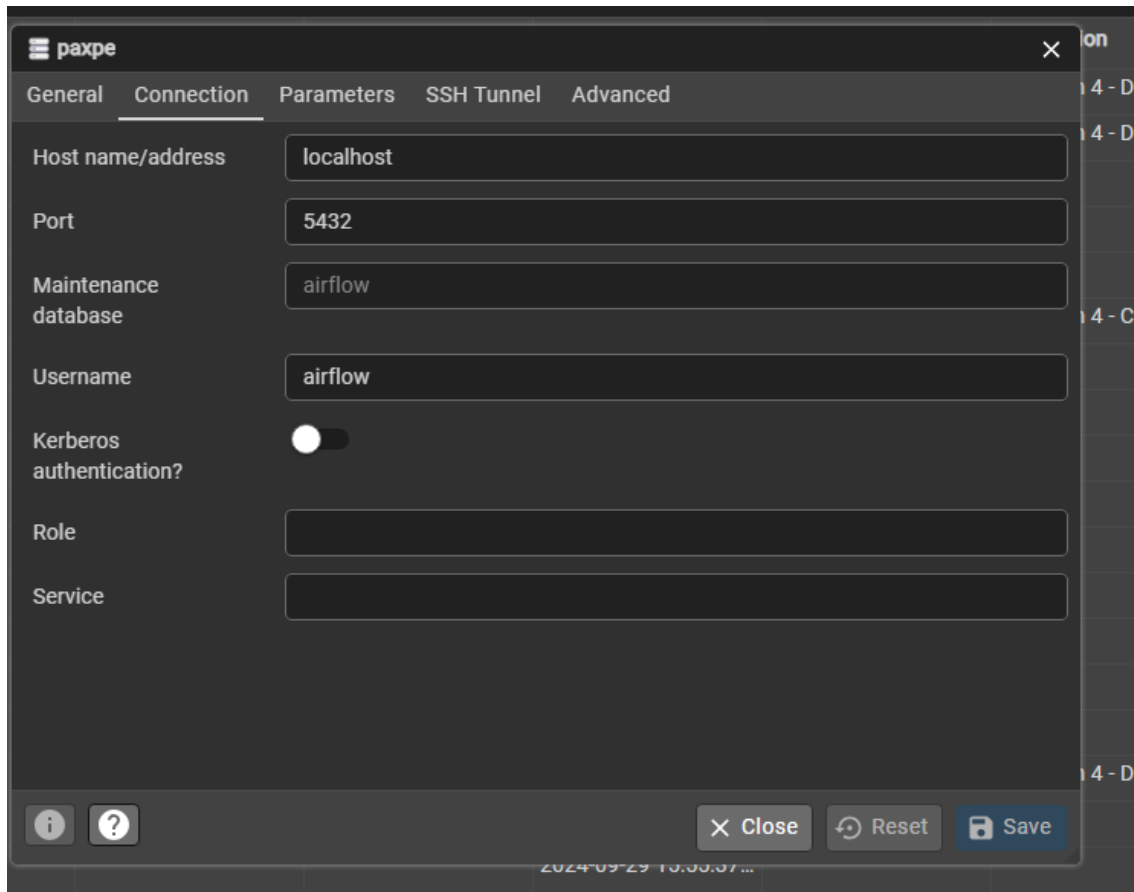
Descrição: Sistema de gerenciamento de banco de dados relacional.

Host: localhost

Porta: 5432

- Banco de Dados: paxpedb

Teste de conexão pelo pgadmin4, uma ferramenta de administração/visualização do banco de dados instalada localmente no servidor



Visualização dos dados no pgadmin4

	ticker	nome_curto	nome_extenso	preco_mercado_regul	mutanca_mercado_regul	mutanca_percentual_mercado_regul	volume_mercado_regul
	character varying (20)	character varying (100)	character varying (100)	double precision	double precision	double precision	bigint
1	JPM	JPMorgan Chase & Co.	JPMorgan Chase & Co.	220.18	1.00999	0.460827	5183
2	CRM	Salesforce, Inc.	Salesforce	254.2	-1.07999	-0.407112	5278
3	AMAT	Applied Materials, Inc.	Applied Materials	195.77	-0.40999146	-0.23441446	56546
4	MUFJ	Mitsubishi UFJ Financial Group,	[null]	10.32	-0.12	-1.14942	97125
5	SBUX	Starbucks Corporation	Starbucks	98.6	2.9700012	3.1057212	159891
6	GSK	GSK plc	[null]	44.15	1.61	3.78467	110875
7	CHQ	Canadian Natural Resources Ltd	Canadian Natural Resources	26.94	-0.470001	-1.25635	61534
8	TFC	Truist Financial Corporation	[null]	43.91	-0.310001	-0.701043	63296
9	ING	ING Group, N.V.	ING Groep	18.3	0.439999	2.4636	51704
10	VOD	Vodafone Group Plc	Vodafone	9.85	0.180003	1.8614302	55946
11	HST	Host Hotels & Resorts, Inc.	Host Hotels & Resorts	17.3	-0.040000916	-0.23084579	72372
12	WDC	Western Digital Corporation	Western Digital	62.85	0.15999965	0.25022817	56420
13	CEP	CenterPoint Energy, Inc (Holdin	CenterPoint Energy	27.22	0.839999	0.142143	74596
14	APTV	Aptiv Plc	Aptiv	70.83	-1.23999	-1.74781	58115
15	KEY	KeyCorp	[null]	16.07	-0.6750008	-6.443899	77965
16	ONON	On Holding AG	[null]	47.92	2.02	4.40896	95144
17	CAVA	CAVA Group, Inc.	CAVA	118.1	-7.7	6.12083	120431
18	SRI	Sirius XM Holdings Inc.	Sirius XM	3.16	0.00000019	1.6077231	122435
19	CHWY	Chewy, Inc.	Chewy	25.86	0.0500011	0.193728	72121

4. Apache Spark

Descrição: Plataforma para processamento de dados distribuídos. Usado aqui para poder ver em tempo real o uso computacional dos Workers do processo

URL de Acesso: <http://localhost:8081/>

Evidencia de visualização:

Podemos ver toda a árvore de execução dos processos no spark master e o nível de recurso físico do hardware da nossa máquina, bem como as aplicações que foram completadas.

Spark Master at spark://spark-master:7077

URL: spark://spark-master:7077
 Alive Workers: 1
 Cores in use: 32 Total, 0 Used
 Memory in use: 14.5 GiB Total, 0.0 B Used
 Resources in use:
 Applications: 0 Running, 0 Completed
 Drivers: 0 Running, 0 Completed
 Status: ALIVE

Workers (1)

Worker Id	Address	State	Cores	Memory	Resources
worker-2024092915538-172.19.0.3-46379	172.19.0.3:46379	ALIVE	32 (0 Used)	14.5 GiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

Em um ambiente de contêiner, temos a versatilidade de desprovisionar para fins de custos os recursos em qualquer momento. Neste caso usaremos via linha de comando a linha “docker compose down” para terminar a imagem em execução:

```
pose.yaml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 12/12
  ✓ Container paxpe_app-airflow-triggerer-1 Removed 1.8s
  ✓ Container paxpe_app-pgadmin-1 Removed 2.0s
  ✓ Container paxpe_app-airflow-webserver-1 Removed 2.9s
  ✓ Container spark-worker Removed 10.3s
  ✓ Container paxpe_app-airflow-worker-1 Removed 3.7s
  ✓ Container paxpe_app-airflow-scheduler-1 Removed 2.4s
  ✓ Container paxpe_app-airflow-init-1 Removed 0.1s
  ✓ Container paxpe_app-redis-1 Removed 0.6s
  ✓ Container paxpe_app-postgres-1 Removed 0.5s
  ✓ Container spark-master Removed 10.3s
  ✓ Network paxpe_app-airflow_network Removed 0.2s
  ✓ Network paxpe_app_default Removed 0.4s
```

Na interface do Docker, podemos ver que a imagem persiste, podendo ser criada na próxima vez sem construir a imagem como um todo, através das dependências abaixo:

Images [Give feedback](#)

Local Hub

0 Bytes / 5.11 GB in use 9 images Last refresh: 25 minutes ago

Name	Tag	Status	Created	Size	Actions
custom-airflow	latest	Unused	1 month ago	2.51 GB	
6c6bcd19e0df	<none>	Unused (dangling)	1 month ago	2.51 GB	
8ab08ec9ef5b	<none>	Unused (dangling)	1 month ago	2.51 GB	
e3884eb3b742	<none>	Unused (dangling)	1 month ago	2.51 GB	
35399c0fe5e6	<none>	Unused (dangling)	1 month ago	2.51 GB	
2c1706d57b09	<none>	Unused (dangling)	1 month ago	2.51 GB	
bitnami/spark	3.4	Unused	1 month ago	1.65 GB	
ba29701bd996		Unused	1 month ago	1.65 GB	
dpape/pgadmin4	latest	Unused	1 month ago	486.53 MB	
b5599f5af8aa		Unused	1 month ago	486.53 MB	
postgres	13	Unused	2 months ago	419.12 MB	
8434335a9e46		Unused	2 months ago	419.12 MB	
redis	7.2-bookworm	Unused	4 months ago	116.43 MB	
6ds5f9c25798		Unused	4 months ago	116.43 MB	

2.1.2 Lições Aprendidas

Destaca-se a dificuldade inicial da configuração mestre do projeto, o arquivo `dockercompose.yaml` que possui toda a forma que o projeto será criado em contêiner bem como a definição de quais pacotes iremos precisar para o cenário. Existiu uma dificuldade em testes com a execução de notebooks pelo airflow num primeiro momento onde foi contornado pelo pacote `papermill` onde se pode encontrar no próprio site da apache. A ser testado na próxima sprint, como pode ser consultado no [Trello](#).



2.2 Sprint 2

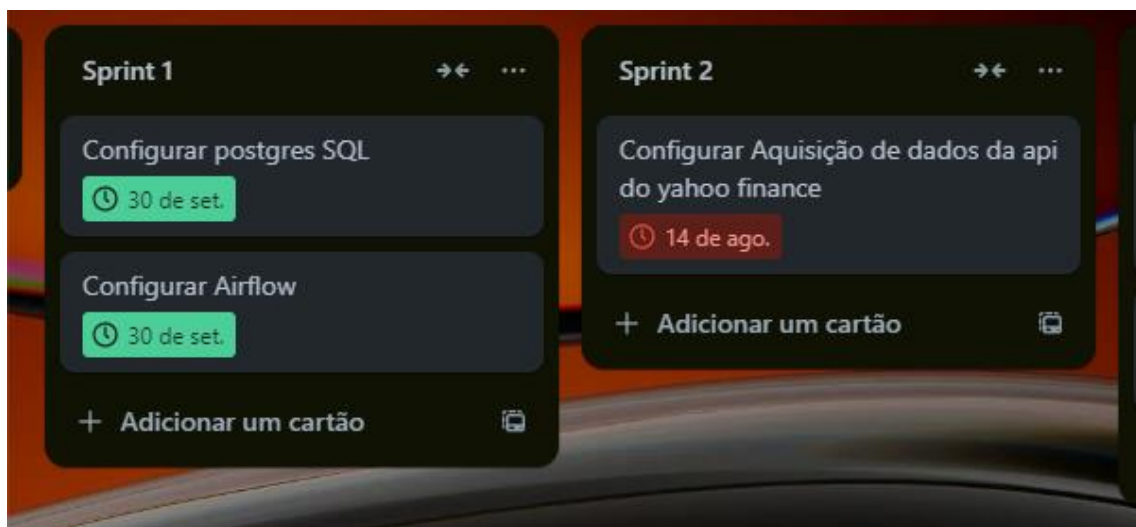
A sprint 2 é pautada pela configuração técnica da aquisição de dados dentro a api gratuita do yahoo finance, para inserção dentro do SGBD configurado anteriormente.

O resultado esperado dessa entrega e ter uma lógica de upsert de registros com uma série temporal de 10 anos das métricas desejadas dentro do apresentado na etapa de Inovação e Design Thinking.

2.2.1 Solução

- Evidência do planejamento:

Definido para sprint 2, a ser entregue no dia 14/10 temos uma entrega planejada



Fonte: [Trello](https://trello.com/).

- Evidência da execução de cada requisito:

Revisitando o desenho apresentado na disciplina de IDT, temos que o processo será pautado pela aquisição do modelo de dados pelo python, uma forma nativa de interagir com a api do yahoo finance para aquisição da fonte única da verdade de dados.

O Script mencionado nessa entrega pode ser consultado no repositório do projeto, em https://github.com/ibmendes/paxpe_app/tree/main como ipynb. O uso de estrutura de notebooks foi utilizado para facilitar a manutenção e entendimento de problemas como abbends, uma vez que se torna difícil entender problemas em algumas situações em uma arquitetura orquestrada por arquivos .py onde a log de abbend fica ao fim do processo.

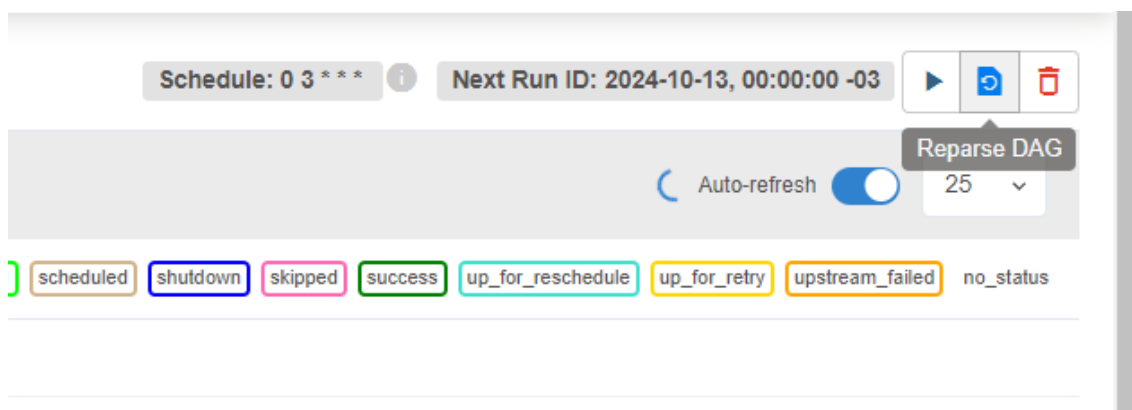
Dentro do arquivo Requirements do Docker incluímos a lib papermill - pache-airflow-providers-papermill do airflow para podermos trabalhar com a estrutura de notebooks (.ipynb) inclusive visualizando em tempo real a sua execução no orquestrador.



Homologação técnica do processo criado:

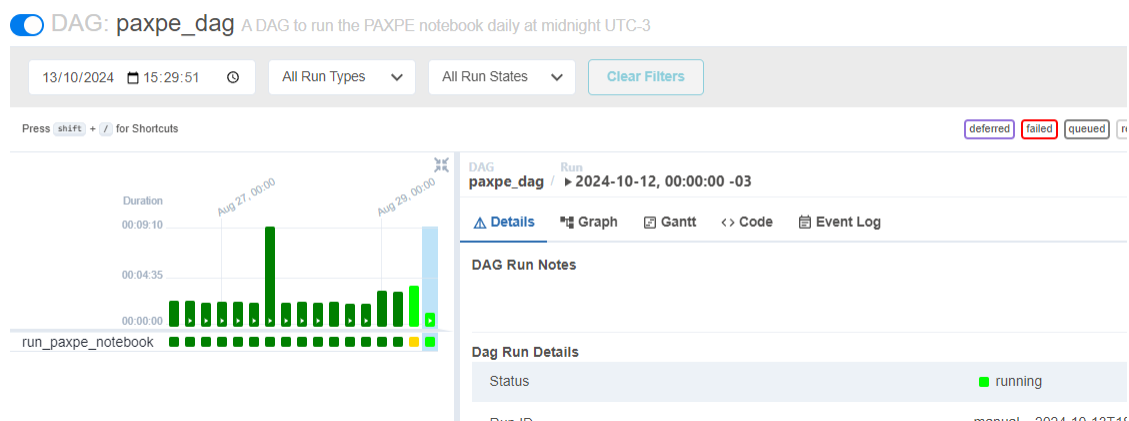
1. Para aquisição dos dados, foi configurado um batch para executar em d-0 contendo as informações do fechamento do dia, retroativo até 10 anos atrás com objetivo de atualizar cálculos sobre acumulativos para executar sempre às 0h em horário de Brasília (UTC-3).

Adicionalmente, é possível executar o dag manualmente tendo acesso de Administrador a console.



Fonte: Airflow

Podemos ver o gantt de execução dentro do canvas, coletando informações sobre algum eventual abend.



2. em tempo real é possível ver a execução do notebook seja pelo arquivo log gerado pelo papermill ou até pelo Spark Master contendo informações de uso computacional da rotina:

```

> config
  > dags
  > _pyscache_
  > utils
  > paxpe_log.ipynb
  > paxpe.py
  > paxpe_dag.py
  > jars
  > postgresql-42.7.4.jar
  > dags
  > dag_id=paxpe_dag
  > dag_processor_manager
  > scheduler
  > plugins
  > docker-compose.yaml
  > Dockerfile
  > paxpe_sgbd_config.sql
  > paxpe.ipynb
  > README.md
  > requirements.txt

spark
...
/home/airflow/.local/lib/python3.8/site-packages/pyspark/bin/load-spark-env.sh: line 68: ps: command not found
24/10/13 18:30:08 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
...
SparkSession - in-memory
SparkContext
Spark UI
Version
3.5.2
Master
local[*]
AppName
PAXPE
...
import pandas as pd

```

Fonte: Visual Studio Code: paxpe - github



Spark Master at spark://spark-master:7077

URL: spark://spark-master:7077

Alive Workers: 1

Cores in use: 24 Total, 0 Used

Memory in use: 14.5 GiB Total, 0.0 B Used

Resources in use:

Applications: 0 Running, 0 Completed

Drivers: 0 Running, 0 Completed

Status: ALIVE

Fonte: Spark Master

3. Para inserção de dados no postgres, iremos utilizar um pacote jar que atende a finalidade dentro da estrutura do projeto (src\airflow-docker\jars\postgresql-42.7.4.jar)

No início do spark, forçamos o uso do pacote

```

spark = (
    SparkSession
    .builder
    .appName("PAXPE")
    .config("spark.sql.session.timeZone", "America/Sao_Paulo") # Define o fuso horário para São Paulo
    .config("spark.driver.memory", "16g") # Memória do driver
    .config("spark.executor.memory", "12g") # Memória para cada executor (ajuste conforme a carga)
    .config("spark.executor.cores", "8") # Núcleos por executor
    .config("spark.cores.max", "24") # Total de núcleos disponíveis
    .config("spark.dynamicAllocation.enabled", "true")
    .config("spark.dynamicAllocation.minExecutors", "2")
    .config("spark.dynamicAllocation.maxExecutors", "10")
    .config("spark.dynamicAllocation.initialExecutors", "4")
    .config("spark.default.parallelism", "24") # Nível de paralelismo
    .config("spark.memory.fraction", "0.8") # Memória usada para armazenamento e execução
    .config("spark.memory.storageFraction", "0.5") # Memória usada para armazenamento
    .config("spark.jars", "/opt/airflow/jars/postgresql-42.7.4.jar")
    .getOrCreate()
)

```

Ao final do código, após criarmos os dataframes com spark tentaremos um force na conexão com o SGBD:

```

postgree upsert

SRVNAME = "postgres"
USER = "airflow"
PASSWORD = "airflow"
HOST = "postgres"
PORT = "5432"
DBNAME = "paxpedb"

# Parâmetros de conexão usando as variáveis
conn_params = {
    'dbname': DBNAME,
    'user': USER,
    'password': PASSWORD,
    'host': HOST,
    'port': PORT
}

```

```

def test_connection():
    try:
        # Connect to the PostgreSQL server using variables
        connection = psycopg2.connect(
            dbname=SRVNAME,
            user=USER,
            password=PASSWORD,
            host=HOST,
            port=PORT
        )
        print("Conexão com postgres sucedida")
        return True
    except OperationalError as e:
        print(f"Error: {e}")
        return False
    finally:
        if connection:
            connection.close()
if not test_connection():
    sys.exit(1)

```

... Conexão com postgres sucedida

Ao sucedir a conexão, o código irão se beneficiar de tabelas em staging dentro do banco, para fazer a comparação com o que já existe e atualizar aquilo que mudou nesses registros e incluir tudo aquilo que for novos resultados desde sua última atualização. Fazendo com que o ambiente tenha um armazenamento de dados históricos de 10 anos partindo do primeiro batch, sendo incremental a cada dia.

O código utiliza a conexão via Java para criação das tabelas stage sobrescrevendo totalmente o staging anterior:

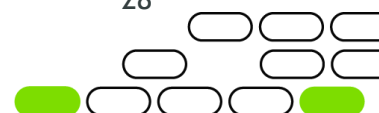
```

def write_to_postgres(df, table_name, schema_name="paxpestg"):
    df.write \
        .format("jdbc") \
        .option("url", f"jdbc:postgresql://{HOST}:{PORT}/{DBNAME}") \
        .option("dbtable", f"{schema_name}.{table_name}") \
        .option("user", USER) \
        .option("password", PASSWORD) \
        .option("driver", "org.postgresql.Driver") \
        .mode("overwrite") \
        .save()

# escrita das tabelas em stage
write_to_postgres(df_retorno_mensal, "temp_retorno_mensal")
write_to_postgres(df_ativas, "temp_captacao_mercado")
write_to_postgres(df_geral, "temp_cadastro")
write_to_postgres(df_financeira, "temp_financeiras")
write_to_postgres(df_mercado, "temp_mercado")
write_to_postgres(df_dividendos, "temp_dividendos")
write_to_postgres(df_valuation, "temp_valuation")

```

em uma def criamos a lógica de upsert com o banco de dados, fazendo o comparativo de colunas chave (PK) dentro do starschema a ser mostrado posteriormente.



Definimos o dataframe e a coluna que iremos comparar com o staging, afim de achar quais os registros serão comparados para atualização (upsert)

```
[19]
# colunas chave para cada tabela
key_columns_retorno_mensal = ["ticker", "data"]
key_columns_cadastro = ["ticker"]
key_columns_cap_mercado = ["ticker"]
key_columns_financas = ["ticker"]
key_columns_mercado = ["ticker"]
key_columns_dividendos = ["ticker", "data_exdividendo"]
key_columns_valuation = ["ticker"]

[20]
```

Passamos o uso desses parâmetros para a função que realiza essa lógica.

```
# realiza o upsert
upsert_data("retorno_mensal", "temp_retorno_mensal", key_columns_retorno_mensal)
upsert_data("cadastro", "temp_cadastro", key_columns_cadastro)
upsert_data("captacao_mercado", "temp_captacao_mercado", key_columns_cap_mercado)
upsert_data("financas", "temp_financas", key_columns_financas)
upsert_data("mercado", "temp_mercado", key_columns_mercado)
upsert_data("dividendos", "temp_dividendos", key_columns_dividendos)
upsert_data("valuation", "temp_valuation", key_columns_valuation)
```

A define em questão irá comparar linha a linha das bases para atualizar os registros novos.

Primeiro, definimos qual o schema fato: destino e o schema stage

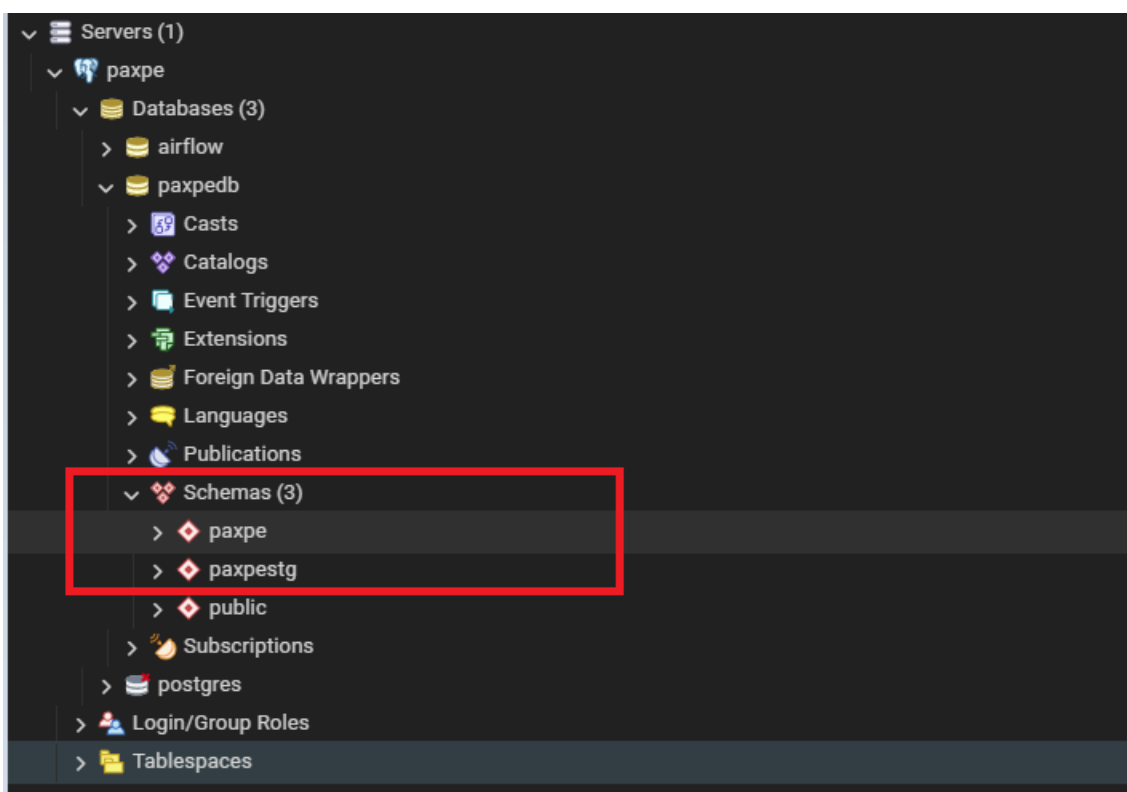
```
def upsert_data(table_name, temp_table_name, key_columns):
    try:
        # Conectar ao PostgreSQL
        connection = psycopg2.connect(**conn_params)
        cursor = connection.cursor()

        # Definir o fuso horário da sessão para UTC-3
        cursor.execute("SET TIME ZONE 'America/Sao_Paulo';")

        # Definir esquemas
        schema_fact = "paxpe"
        schema_stg = "paxpestg"
```

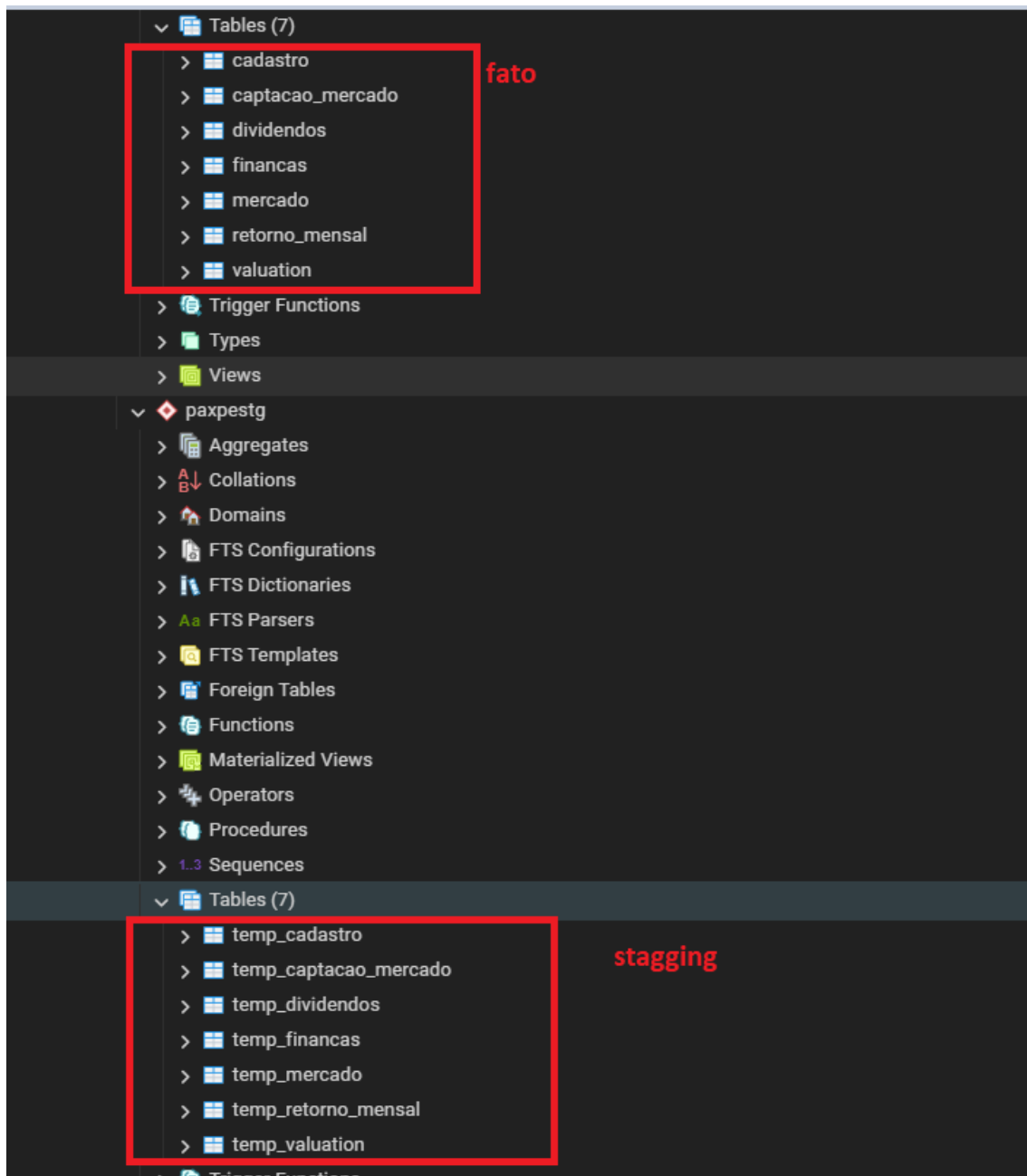
Visualização dos schemas no postgres





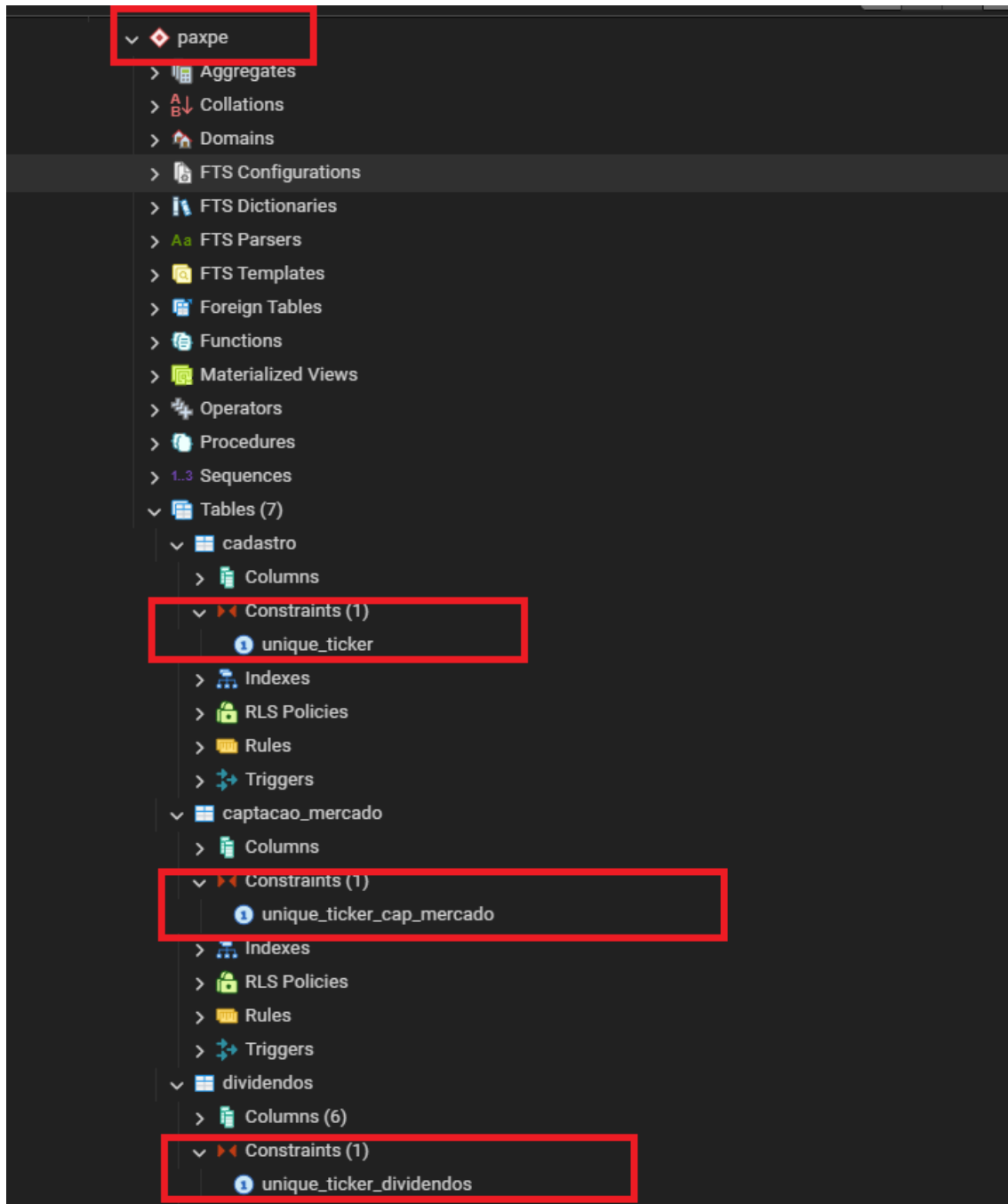
Fonte: pgadmin web.

Os dois schemas terão as mesmas estruturas tabulares dentro do modelo relacional estruturado



Fonte: pgadmin web.

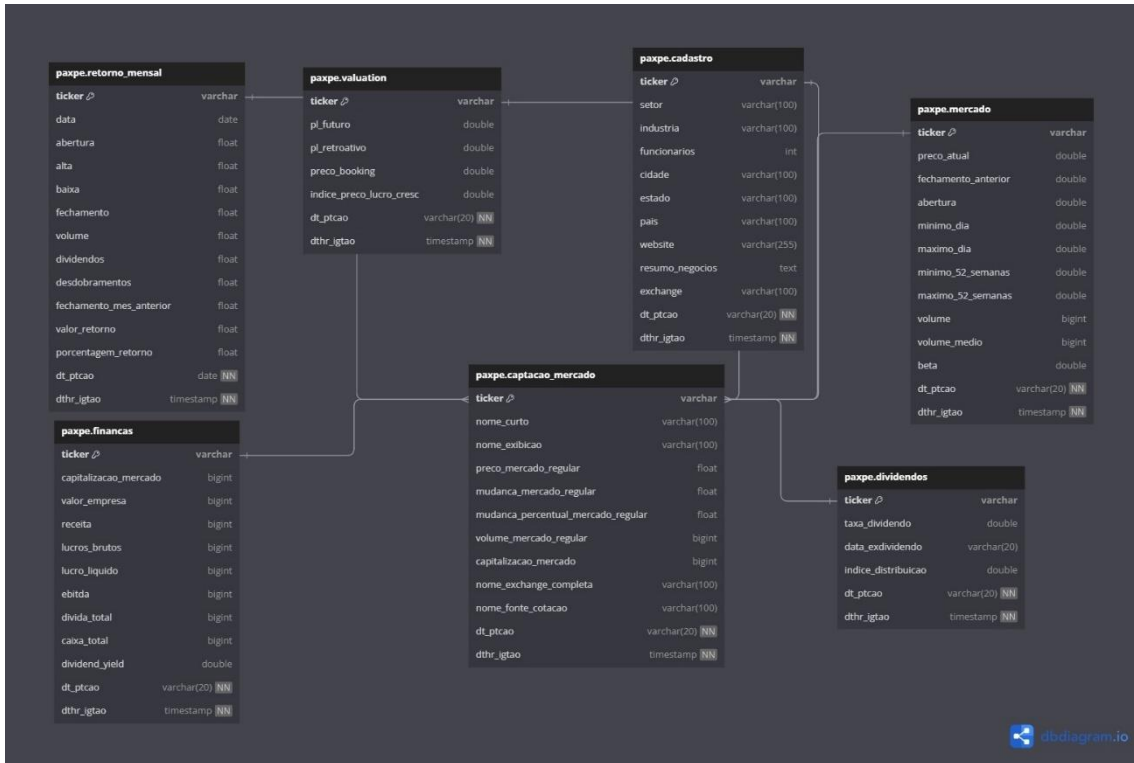
Nas tabelas de produção, no schema fato, temos as constraints. Os chaveamentos pertinentes para lógica de upsert sendo as chaves únicas das tabelas, onde se constroem os relacionamentos (pk)



- Evidência dos resultados:

para demonstrar a evidência de resultados, principalmente visualizando os dados, apresenta-se o starschema criado para a rotina a ser replicado na sprint 3 para criação do painel em powerbi:





fonte: [PAXPE - STARSHEMA - dbdiagram](#)

a aquisição de cada tabela representa um dataframe separado, para os dados cadastrais das empresas onde expandimos o limite permitido de empresas para se obter (dentro de 200) de forma gratuita, ordenamos pelo tamanho da empresa através da métrica market cap:

Tabela fato - maiores empresas segundo a api do yahoo finance

df_ativas = obter_empresas_ativas()

df_ativas.show()

df_ativas.printSchema()

24/10/13 18:35:51 WARN SparkStringUtils: Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.sql.debug.maxToStringFields'.

[Stage 0] (0 + 24) / 24]

ticker	nome_curto	nome_exibicao	preco_mercado_regular	mudanca_mercado_regular	mudanca_percentual_mercado_regular	volume_mercado_regular	capitalizacao_mercado	nome_exchange_completa	nome_fonte_cotacao	dt_ptcao	
APPL	Apple Inc.	Apple	227.55	-1.489982	-0.6595131	31759188	345969154208	NasdaqGS[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
MSFT	Microsoft Corporation	Microsoft	426.22	0.480811	0.11543166	14149444	389452118336	NasdaqGS[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
GOOG	Alphabet Inc.	Alphabet	164.52	1.3480116	0.8218624	18945971	2016636829696	NasdaqGS[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
AMZN	Amazon.com, Inc.	Amazon.com	188.82	2.1740134	1.162811	25731557	198179148800	NasdaqGS[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
META	Meta Platforms, Inc.	Meta Platforms	589.85	6.1129955	1.0405485	5821651	1422407138784	NasdaqGS[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
TSM	Taiwan Semiconductor	Taiwan Semiconductor	198.81	0.489988	2.7875	12138677	989555851264	NYSE[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
AVGO	Broadcom Inc.	Broadcom	181.48	-4.218087	-2.2672231	23878023	84761688332	NasdaqGS[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
TSLS	Tesla, Inc.	Tesla	217.48	-20.978001	-9.742511	143282874	69572526272	NasdaqGS[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
WMT	Walmart Inc.	Walmart	80.11	0.489988	0.615488	7176423	64386388952	NYSE	Delayed Quote	2024-10-13	2024-10-13
JPM	JPMorgan Chase & Co.	JPMorgan Chase & Co.	222.29	9.45	4.43995	16243847	625813828864	NYSE[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
COIN	Coinbase Global Corp.	Coinbase Global	123.61	0.478001	0.38168	8162025	549178212352	NYSE	Delayed Quote	2024-10-13	2024-10-13
ORCL	Oracle Corporation	Oracle	175.77	0.368001	0.285234	4834668	487869228864	NYSE	Delayed Quote	2024-10-13	2024-10-13
PG	Procter & Gamble	Procter & Gamble	171.89	2.14	1.26665	4365893	482011881472	NYSE	Delayed Quote	2024-10-13	2024-10-13
DIS	Walt Disney Company	Walt Disney Company	161.46	0.954012	0.591871	5252195	388672978944	NYSE[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
BAC	Bank of America Corp.	Bank of America	41.95	1.98	4.65731	58660833	325543395648	NYSE	Delayed Quote	2024-10-13	2024-10-13
KO	Coca-Cola Company	Coca-Cola Company	69.57	0.32	0.462893	5798989	299837652992	NYSE	Delayed Quote	2024-10-13	2024-10-13
MRK	Merck & Co., Inc.	Merck & Co., Inc.	189.73	0.338002	0.381647	8572149	278144712784	NYSE[Nasdaq Real Time ...]	2024-10-13	2024-10-13	
AMD	Advanced Micro Devices, Inc.	Advanced Micro Devices, Inc.	167.89	3.7188007	2.297191	42114275	27172668384	NasdaqGS[Nasdaq Real Time ...]	2024-10-13	2024-10-13	

Partindo dessa tabela fato, começamos a trabalhar nas demais tabelas abaixo por meio de uma tupla contendo a PK dessa tabela, os tickers.

```

> # Selecionar a coluna 'symbol' e coletar os valores como uma lista
> # O método para dimensionar os 100 maiores e retornar
symbol_list = df_ativas.select('ticker').rdd.flatMap(lambda x: x).collect()

> # Converter a lista para uma tupla
df_tickers = tuple(symbol_list)

> print(df_tickers)

('AAPL', 'MSFT', 'GOOG', 'GOGL', 'AMZN', 'META', 'TSM', 'AVGO', 'TSLS', 'WMT', 'JPM', 'COIN', 'ORCL', 'PG', 'DIS', 'BAC', 'KOD', 'MRK', 'AMD', 'BABA', 'CSCO', 'NFC', 'ROD', 'UBER', 'VZ', 'HGS', 'DIS', 'NEE', 'PFE', 'OCSA', 'T', 'COP',

```

Para criação da tabela retorno_mensal, precisamos obter os dados históricos das empresas no intervalo de 10 anos. Aqui esperamos obter o cálculo de fechamento da ação no último dia do mês menos o fechamento do próximo mês.

Definimos o intervalo, baseado em hoje

Dimensão - Retornos mensais 10 anos

```
# 10 anos passados
start_date = datetime.today() - timedelta(days=10*365)

# hoje
end_date = datetime.today()

# 'YYYY-MM-DD'
start_date_str = start_date.strftime('%Y-%m-%d')
end_date_str = end_date.strftime('%Y-%m-%d')

print(f"start_date: {start_date_str}")
print(f"end_date: {end_date_str}")
```

... start_date: 2014-10-16
end_date: 2024-10-13

Definimos o intervalo mensal

```
def obter_dados_historicos(symbols, start_date, end_date):
    dados = {}
    for symbol in symbols:
        ticker = yf.Ticker(symbol)
        historico = ticker.history(start=start_date, end=end_date, interval='1mo')
        dados[symbol] = historico

    return dados
```

Depois tratamos o arquivo que estamos recebendo da api para o formato tabular, pois o resultado coletado dessa função é similar a um arquivo json.



```
def retorno_mensal(dados):
    # Inicializar uma lista vazia para armazenar dados estruturados
    dados_estruturados = []

    # Iterar sobre os dados históricos de cada símbolo
    for symbol, df in dados.items():
        # Converter DataFrame do Pandas para PySpark
        df['symbol'] = symbol

        df_spark = spark.createDataFrame(df.reset_index())

        # Renomear colunas para português
        df_spark = df_spark.withColumnRenamed('symbol', 'ticker') \
            .withColumnRenamed('Date', 'data') \
            .withColumnRenamed('Open', 'abertura') \
            .withColumnRenamed('High', 'alta') \
            .withColumnRenamed('Low', 'baixa') \
            .withColumnRenamed('Close', 'fechamento') \
            .withColumnRenamed('Volume', 'volume') \
            .withColumnRenamed('Dividends', 'dividendos') \
            .withColumnRenamed('Stock Splits', 'desdobramentos')

        janela = Window.partitionBy('ticker').orderBy('data')

        # Calcular preço de fechamento do mês anterior (deslocar uma linha para cima)
        df_spark = df_spark.withColumn('fechamento_mes_anterior', lag('fechamento').over(janela))

        # Calcular Retorno em valor (diferença absoluta)
        df_spark = df_spark.withColumn('valor_retorno', col('fechamento') - col('fechamento_mes_anterior'))

        # Calcular Retorno em porcentagem
        df_spark = df_spark.withColumn('porcentagem_retorno', (col('valor_retorno') / col('fechamento_mes_anterior')) * 100)

        # Adicionar coluna com data atual no formato desejado
        df_spark = df_spark.withColumn('dt_ptcao', date_format(current_timestamp(), 'yyyy-MM-dd'))
        df_spark = df_spark.withColumn('dthr_igtao', current_timestamp())

    # Reordenar colunas
    df_spark = df_spark.select(
        'ticker',          # 'symbol' traduzido para 'ticker'
        'data',            # 'date' traduzido para 'Data'
        'abertura',        # 'open' traduzido para 'Abertura'
        'alta',            # 'high' traduzido para 'Alta'
        'baixa',           # 'low' traduzido para 'Baixa'
        'fechamento',     # 'close' traduzido para 'Fechamento'
        'volume',          # 'volume' mantido como 'Volume'
        'dividendos',      # 'dividends' traduzido para 'Dividendos'
        'desdobramentos',  # 'splits' traduzido para 'Desdobramentos'
    )
```

estrutura SGBD

Ao fim, espera-se obter o retorno de 10 anos, ou o começo da empresa na bolsa, por meio de cada ticker

[Stage 22]:(20 + 4) / 24[[Stage 20]:(0 + 20) / 24[[Stage 20]:(0 + 0) / 24]

24/10/13 18:36:39 M00N OMScheduler: Broadcasting large task binary with size 3.8 MiB

ticker	data	abertura	alta	baixa	fechamento	volume	dividendos	desdobramentos	fechamento_mes_anterior	valor_retorno	porcentagem_retorno	dt_ptcao	dthr_igtao
GOOG	2014-11-01 02:00:00	27.61096871988847	27.758474578742263	26.3666811367571	26.95113754272461	570413781	0.0	0.0	NULL	NULL	NULL	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2014-12-01 03:00:00	26.80539597299805	26.9384576237697	24.32332276608954	26.18363388432129	917297545	0.0	0.0	-0.7675837384033203	-2.847760888740692	-2.847760888740692	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-01-01 03:00:00	26.31345726937265	26.9804821152423	24.251694726376705	26.5875381361884	1084562476	0.0	0.0	26.18363388432129	0.4838963317871094	1.54252629652387	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-02-01 03:00:00	26.4407534437417	28.400212039378023	25.79316735532755	27.77534404632812	651980915	0.0	0.0	26.5875381361884	1.187814712524414	4.467523814894802	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-03-01 02:00:00	27.88129148920923	28.745789328285786	27.87801783668872	27.258037567138672	77272385	0.0	0.0	27.77534404632812	-0.5172872814941486	-1.862469338592583	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-04-01 01:00:00	27.2878251109641	28.4094386386858	25.91552872328964	26.80118179321289	847552576	0.0	1.0027455	27.258037567138672	-0.45685577392578125	-1.67684059965178	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-05-01 01:00:00	26.85554720886295	27.142841767525286	25.9984234352814	26.540321358097656	67318080	0.0	0.0	26.80118179321289	-0.2088884413152344	-0.77316942450412	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-06-01 01:00:00	26.77240354228681	27.12037901457736	25.95124708008214	25.981742481123847	686418080	0.0	0.0	26.540321358097656	-0.578579489746694	-2.17999772776232	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-07-01 01:00:00	26.17225125987367	33.84887196994136	25.69589590367327	31.203868865966797	1266380000	0.0	0.0	25.981742481123847	5.24212646464375	20.19173591686466	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-08-01 01:00:00	31.19040237543635	33.6623385135368	28.183287625257712	30.83677101135254	1074348000	0.0	0.0	31.203868865966797	-0.3670978546142578	-1.17644388956029244	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-09-01 01:00:00	30.84421494822838	32.46528931123239	29.39886252979594	30.346473693847656	959180000	0.0	0.0	30.83677101135254	-0.4902973175048828	-1.589974192138895	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-10-01 01:00:00	30.3439230738013	31.41852788716275	29.91982507283864	35.453433994876316	98772308	0.0	0.0	30.346473693847656	5.18964826638899	16.82884321332244	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-11-01 02:00:00	35.4659032974139	38.9417437464413	35.20683961596543	37.039039611816486	678696000	0.0	0.0	35.453433994876316	1.5856856213378986	4.472361187251215	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2015-12-01 03:00:00	37.26398821048021	38.98363358349625	36.1197866938647	37.85184751586914	862806000	0.0	0.0	37.039039611816486	0.8128079840527344	2.192382884184164	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2016-01-01 03:00:00	37.05891977466586	37.5878862595462	33.580631862480261	37.0564966650396	911224000	0.0	0.0	37.85184751586914	-0.7945518493652344	-2.09915418818206	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2016-02-01 03:00:00	37.43107664838258	39.39675841869364	33.071783117034316	34.88301592138672	1287348000	0.0	0.0	37.0564966650396	-2.2534637451171875	-6.88115717524293	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2016-03-01 02:00:00	35.0948146368082	37.801169677467065	34.1838522592331	37.156253814697266	834848000	0.0	0.0	34.88301592138672	2.353221893318547	6.761542783473629	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2016-04-01 01:00:00	36.839531486567594	38.40069627894537	34.36568729381475	34.56561668766816	843800000	0.0	0.0	37.156253814697266	-2.59863728783125	-6.97227772194459	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2016-05-01 01:00:00	34.79684748119498	36.895898191211786	34.3641038932395	36.6958888889250	699330000	0.0	0.0	34.56561668766816	-2.130264382228562	-6.16297560994613	2024-10-13 15:36:...	2024-10-13 15:36:...
GOOG	2016-06-01 01:00:00	36.6365262338235	36.770196814608147	33.882951772469436	34.52822171820580	787250000	0.0	0.0	36.6958888889250	-2.1756591796875	-5.92882098683558	2024-10-13 15:36:...	2024-10-13 15:36:...

only showing top 20 rows

As demais tabelas, são possíveis criar trabalhando pelo screener, uma forma nativa de se obter resultados da api, pois não é necessário criar cálculos para se obter resultados. Definimos quais escopos queremos em cada etapa:

Schemas:

```

from pyspark.sql.functions import from_unixtime, col

def criar_tabelas_spark(tickers):
    if isinstance(tickers, str):
        tickers = [tickers,]

    # Inicializa as tabelas gerais
    schema_geral = StructType([
        StructField('ticker', StringType(), False),
        StructField('setor', StringType(), True),
        StructField('industria', StringType(), True),
        StructField('funcionarios', IntegerType(), True),
        StructField('cidade', StringType(), True),
        StructField('estado', StringType(), True),
        StructField('pais', StringType(), True),
        StructField('website', StringType(), True),
        StructField('resumo_negocio', StringType(), True),
        StructField('exchange', StringType(), True)
    ])

    # Inicializa a tabela financeira
    schema_financeira = StructType([
        StructField('ticker', StringType(), False),
        StructField('capitalizacao_mercado', LongType(), True),
        StructField('valor_empresa', LongType(), True),
        StructField('lucros_brutos', LongType(), True),
        StructField('lucro_liquido', LongType(), True),
        StructField('divida_total', LongType(), True),
        StructField('caixa_total', LongType(), True),
        StructField('dividend_yield', DoubleType(), True)
    ])

    # Inicializa a tabela de mercado
    schema_mercado = StructType([
        StructField('ticker', StringType(), False),
        StructField('preco_atual', DoubleType(), True),
        StructField('fechamento_anterior', DoubleType(), True),
        StructField('abertura', DoubleType(), True),
        StructField('minimo_dia', DoubleType(), True),
        StructField('maximo_dia', DoubleType(), True),
        StructField('minimo_52_semanas', DoubleType(), True),
        StructField('maximo_52_semanas', DoubleType(), True),
        StructField('volume', LongType(), True),
        StructField('volume_medio', LongType(), True),
        StructField('beta', DoubleType(), True)
    ])

    # Inicializa a tabela de dividendos
    schema_dividendos = StructType([
        StructField('ticker', StringType(), False),
        StructField('taxa_dividendo', DoubleType(), True),
        StructField('data_dividendo', StringType(), True), # Temporariamente como StringType
        StructField('indice_distribuido', DoubleType(), True)
    ])

    # Inicializa a tabela de valuation
    schema_valuation = StructType([
        StructField('ticker', StringType(), False),
        StructField('pl_futuro', DoubleType(), True),
        StructField('pl_retroativo', DoubleType(), True),
        StructField('pvc_bookings', DoubleType(), True),
        StructField('indice_preco_lucro_cresc', DoubleType(), True)
    ])

    # Inicializa as listas de dicionários para cada tabela
    geral = []
    financeira = []
    mercado = []
    dividendos = []
    valuation = []

    for ticker in tickers:
        try:
            empresa = yf.Ticker(ticker)
            info = empresa.info

            # Filtra e trata valores infinitos
            def safe_get(key, default_value):
                value = info.get(key)
                if isinstance(value, str) and value in ('Infinity', '-Infinity'):
                    return default_value
                return value

            # Preencher dados da tabela geral
            geral.append({
                'ticker': ticker,
                'setor': info.get('sector'),
                'industria': info.get('industry'),
                'funcionarios': info.get('fullTimeEmployees'),
                'cidade': info.get('city'),
                'estado': info.get('state'),
                'pais': info.get('country'),
                'website': info.get('website'),
                'resumo_negocio': info.get('longBusinessSummary'),
                'exchange': info.get('exchange')
            })

            # Preencher dados da tabela financeira
            financeira.append({
                'ticker': ticker,
                'capitalizacao_mercado': safe_get('marketCap', 0),
                'valor_empresa': safe_get('enterpriseValue', 0),
                'lucros_brutos': safe_get('grossProfits', 0),
                'lucro_liquido': safe_get('netIncome', 0),
                'divida_total': safe_get('totalDebt', 0),
                'caixa_total': safe_get('totalCash', 0),
                'dividend_yield': safe_get('dividendYield', 0.0)
            })

            # Preencher dados da tabela de mercado
            mercado.append({
                'ticker': ticker,
                'preco_atual': safe_get('currentPrice', 0.0),
                'fechamento_anterior': safe_get('previousClose', 0.0),
                'abertura': safe_get('open', 0.0),
                'minimo_dia': safe_get('dayLow', 0.0),
                'maximo_dia': safe_get('dayHigh', 0.0),
                'minimo_52_semanas': safe_get('fiftyTwoWeekLow', 0.0),
                'maximo_52_semanas': safe_get('fiftyTwoWeekHigh', 0.0),
                'volume': safe_get('volume', 0),
                'volume_medio': safe_get('averageVolume', 0),
                'beta': safe_get('beta', 0.0)
            })

            # Preencher dados da tabela de dividendos
            dividendos.append({
                'ticker': ticker,
                'taxa_dividendo': safe_get('dividendYield', 0.0),
                'data_dividendo': safe_get('dividendDate', None),
                'indice_distribuido': safe_get('dividendPayoutRatio', 0.0)
            })

            # Preencher dados da tabela de valuation
            valuation.append({
                'ticker': ticker,
                'pl_futuro': safe_get('priceToBook', 0.0),
                'pl_retroativo': safe_get('priceToEarnings', 0.0),
                'pvc_bookings': safe_get('priceToSales', 0.0),
                'indice_preco_lucro_cresc': safe_get('priceToEarningsGrowth', 0.0)
            })

        except Exception as e:
            print(f"Erro ao processar o ticker {ticker}: {e}")

```

Criação dos dataframes:

```

# Preencher dados da tabela de dividendos
dividendos.append({
    'ticker': ticker,
    'taxa_dividendo': safe_get('dividendRate', 0.0),
    'data_exdividendo': safe_get('exDividendDate'), # Unix timestamp
    'indice_distribuido': safe_get('payoutRatio', 0.0)
})

# Preencher dados da tabela de valuation
valuation.append({
    'ticker': ticker,
    'pl_futuro': safe_get('forwardPE', 0.0),
    'pl_retroativo': safe_get('trailingPE', 0.0),
    'preco_booking': safe_get('priceToBook', 0.0),
    'indice_preco_lucro_cresc': safe_get('pegRatio', 0.0)
})

except Exception as e:
    print(f"Erro ao processar o ticker {ticker}: {e}")

# Criar DataFrames Spark com esquema definido
df_geral = spark.createDataFrame(geral, schema=schema_geral)
df_financeira = spark.createDataFrame(financeira, schema=schema_financeira)
df_mercado = spark.createDataFrame(mercado, schema=schema_mercado)
df_dividendos = spark.createDataFrame(dividendos, schema=schema_dividendos)
df_valuation = spark.createDataFrame(valuation, schema=schema_valuation)

# Converter Unix timestamp para data legível
df_dividendos = df_dividendos.withColumn('data_exdividendo', from_unixtime(col('data_exdividendo').cast('bigint')))

# Adicionar colunas de timestamp
df_geral = df_geral.withColumn('dt_ptcao', date_format(current_timestamp(), 'yyyy-MM-dd'))

```

Resultados

Tabela de cadastro das empresas:

```

df_geral, df_financeira, df_mercado, df_dividendos, df_valuation = criar_tabelas_spark(df_tickers)
# Exibir os DataFrames
df_geral.show()
df_financeira.show()
df_mercado.show()
df_dividendos.show()
df_valuation.show()

```

ticker	setor	industria	funcionarios	cidade	estado	pais	website	resumo_negocios	exchange	dt_ptcao	dthr_igtao
AAPL	Technology	Consumer Electronics	161000	Cupertino	CA	United States	https://www.apple.com	Apple Inc. design...	NMS	2024-10-13	2024-10-13 15:38:...
WDAI	Technology	Semiconductors	296000	Santa Clara	CA	United States	https://www.nvidia.com	NVIDIA Corporatio...	NMS	2024-10-13	2024-10-13 15:38:...
MSFT	Technology	Software - Infras...	228000	Redmond	WA	United States	https://www.micro...	Microsoft Corpora...	NMS	2024-10-13	2024-10-13 15:38:...
GOOG	Communication Ser...	Internet Content ...	179582	Mountain View	CA	United States	https://abc.xyz	Alphabet Inc. off...	NMS	2024-10-13	2024-10-13 15:38:...
AMZN	Consumer Cyclical	Internet Retail	1525000	Seattle	WA	United States	https://www.about...	Amazon.com, Inc. ...	NMS	2024-10-13	2024-10-13 15:38:...
META	Communication Ser...	Internet Content ...	70799	Menlo Park	CA	United States	https://investor...	Meta Platforms, I...	NMS	2024-10-13	2024-10-13 15:38:...
TSM	Technology	Semiconductors	208000	Hsinchu City	NULL	Taiwan	https://www.tsmc.com	Taiwan Semiconduc...	NVQ	2024-10-13	2024-10-13 15:38:...
AVGO	Technology	Semiconductors	208000	Palo Alto	CA	United States	https://www.broad...	Broadcom Inc. des...	NMS	2024-10-13	2024-10-13 15:38:...
TSLA	Consumer Cyclical	Auto Manufacturers	140473	Austin	TX	United States	https://www.tesla.com	Tesla, Inc. desig...	NMS	2024-10-13	2024-10-13 15:38:...
WMT	Consumer Defensive	Discount Stores	2100000	Bentonville	AR	United States	https://corporate...	Walmart Inc. enga...	NVQ	2024-10-13	2024-10-13 15:38:...
JPM	Financial Services	Banks - Diversified	316043	New York	NY	United States	https://www.jpmorg...	JPMorgan Chase & ...	NVQ	2024-10-13	2024-10-13 15:38:...
XOM	Energy	Oil & Gas Integrated	62000	Spring	TX	United States	https://corporate...	Exxon Mobil Corpo...	NVQ	2024-10-13	2024-10-13 15:38:...
ORCL	Technology	Software - Infras...	159000	Austin	TX	United States	https://www.oracle.com	Oracle Corporatio...	NVQ	2024-10-13	2024-10-13 15:38:...
PG	Consumer Defensive	Household & Perso...	108000	Cincinnati	OH	United States	https://www.pg.com	The Procter & Gam...	NVQ	2024-10-13	2024-10-13 15:38:...
JNJ	Healthcare	Drug Manufacturer...	131000	New Brunswick	NJ	United States	https://www.jnj.com	Johnson & Johnson...	NVQ	2024-10-13	2024-10-13 15:38:...
BAC	Financial Services	Banks - Diversified	212000	Charlotte	NC	United States	https://www.bankof...	Bank of America C...	NVQ	2024-10-13	2024-10-13 15:38:...
KO	Consumer Defensive	Beverages - Non-Al...	79100	Atlanta	GA	United States	https://www.coca...	The Coca-Cola Com...	NVQ	2024-10-13	2024-10-13 15:38:...
MRK	Healthcare	Drug Manufacturer...	70000	Rahway	NJ	United States	https://www.merck.com	Merck & Co., Inc...	NVQ	2024-10-13	2024-10-13 15:38:...
AMD	Technology	Semiconductors	26000	Santa Clara	CA	United States	https://www.amd.com	Advanced Micro De...	NMS	2024-10-13	2024-10-13 15:38:...

only showing top 20 rows

Tabela de finanças:

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output settings.

ticker	preco_atual	fechamento_anterior	abertura	minimo_dia	maximo_dia	minimo_52_semanas	maximo_52_semanas	volume	volume_medio	beta	dt_ptcao	dthr_igtao
AAPL	227.55	229.04	229.32	227.34	229.41	164.08	237.23	31759188	52131056	1.239	2024-10-13	2024-10-13 15:38:...
WDAI	134.8	134.81	134.02	133.66	135.78	39.23	140.76	170209474	324770165	1.669	2024-10-13	2024-10-13 15:38:...
MSFT	416.32	415.84	415.91	413.27	417.1099	324.39	468.35	141449444	19930348	0.896	2024-10-13	2024-10-13 15:38:...
GOOG	164.52	163.18	163.33	162.5	165.27	121.46	193.31	10945971	17891218	1.038	2024-10-13	2024-10-13 15:38:...
GOOGL	163.24	162.08	162.13	161.235	163.9	120.21	191.75	15344251	25478684	1.038	2024-10-13	2024-10-13 15:38:...
AMZN	188.82	186.65	186.63	186.3	189.9284	118.35	201.2	25751557	39839331	1.147	2024-10-13	2024-10-13 15:38:...
META	589.95	583.83	584.54	582.71	591.21	279.4	602.95	8587051	13605935	1.216	2024-10-13	2024-10-13 15:38:...
TSM	198.81	185.78	188.17	188.01	191.4899	84.95	193.47	12130677	15634181	1.268	2024-10-13	2024-10-13 15:38:...
AVGO	181.48	185.69	181.14	177.4	182.55	81.834	186.42	23070023	27799742	1.184	2024-10-13	2024-10-13 15:38:...
TSLA	217.8	238.77	228.12	214.4	223.34	138.8	271.0	142628874	85236089	2.297	2024-10-13	2024-10-13 15:38:...
WMT	80.1	79.61	79.7	79.385	80.13	49.84667	81.6	7176423	16498103	0.516	2024-10-13	2024-10-13 15:38:...
JPM	222.29	212.84	215.67	215.16	224.6299	135.19	225.48	16243847	8911495	1.102	2024-10-13	2024-10-13 15:38:...
XOM	123.61	123.14	123.1	122.9504	124.0399	95.77	126.34	8162025	14287645	0.877	2024-10-13	2024-10-13 15:38:...
ORCL	175.77	175.41	175.405	174.4	177.0	99.26	178.61	4834668	8383640	1.008	2024-10-13	2024-10-13 15:38:...
PG	171.09	168.95	169.46	168.97	171.41	142.5	177.04	4365893	7056606	0.407	2024-10-13	2024-10-13 15:38:...
JNJ	161.46	160.51	162.18	161.2	162.66	143.13	168.85	5252195	6547184	0.518	2024-10-13	2024-10-13 15:38:...
BAC	41.95	39.97	40.5	40.4	42.17	24.96	44.44	50569983	40139662	1.338	2024-10-13	2024-10-13 15:38:...
KO	69.57	69.25	69.46	68.995	69.59	52.84	73.53	5790999	13763570	0.608	2024-10-13	2024-10-13 15:38:...
MRK	109.73	109.4	109.1	107.9	109.915	99.14	134.63	8572149	9523098	0.399	2024-10-13	2024-10-13 15:38:...
AMD	167.89	164.18	164.185	163.0101	169.35	93.12	227.3	42136175	44674215	1.695	2024-10-13	2024-10-13 15:38:...

Tabela de valuation:

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#).

ticker	pl_futuro	pl_retroativo	preco_booking	indice_preco_lucro_cresc	dt_ptcao	dthr_igtao
AAPL	30.421124	34.58287	51.928345	3.09	2024-10-13	2024-10-13 15:38:...
NVDA	33.44913	63.286385	56.925674	0.9	2024-10-13	2024-10-13 15:38:...
MSFT	27.353483	35.34126	11.527619	2.16	2024-10-13	2024-10-13 15:38:...
GOOG	18.888634	23.578282	6.7404127	1.04	2024-10-13	2024-10-13 15:38:...
GOOGL	18.741676	23.428374	6.687971	1.03	2024-10-13	2024-10-13 15:38:...
AMZN	32.49914	45.06444	8.377108	1.24	2024-10-13	2024-10-13 15:38:...
META	24.287773	38.207373	9.532543	2.01	2024-10-13	2024-10-13 15:38:...
TSM	23.016888	34.134167	1.3049246	1.31	2024-10-13	2024-10-13 15:38:...
AVGO	29.461039	147.54471	3.0644534	1.94	2024-10-13	2024-10-13 15:38:...
TSLA	70.944626	61.352116	10.466123	73.56	2024-10-13	2024-10-13 15:38:...
WMT	29.448528	41.71875	7.623489	3.19	2024-10-13	2024-10-13 15:38:...
JPM	13.399035	12.363181	1.9304217	7.0	2024-10-13	2024-10-13 15:38:...
XOM	14.750597	14.785886	2.046083	4.46	2024-10-13	2024-10-13 15:38:...
ORCL	24.548883	45.301548	45.034588	2.42	2024-10-13	2024-10-13 15:38:...
PG	23.026918	28.420265	8.148695	3.75	2024-10-13	2024-10-13 15:38:...
JNJ	15.362513	24.463638	5.432157	2.44	2024-10-13	2024-10-13 15:38:...
BAC	11.588398	14.719299	1.2199732	1.25	2024-10-13	2024-10-13 15:38:...
KO	22.809835	28.280487	11.595	4.03	2024-10-13	2024-10-13 15:38:...
MRK	11.312371	20.32037	6.3841057	0.16	2024-10-13	2024-10-13 15:38:...
AMD	30.91897	199.86905	4.804682	1.46	2024-10-13	2024-10-13 15:38:...

only showing top 28 rows

2.2.2 Lições Aprendidas

Destaca-se que na criação das tabelas do postgres foi necessário criar constraints, pois o python estava identificando erros ao tentar comparar as tabelas em staging com as tabelas de destino.

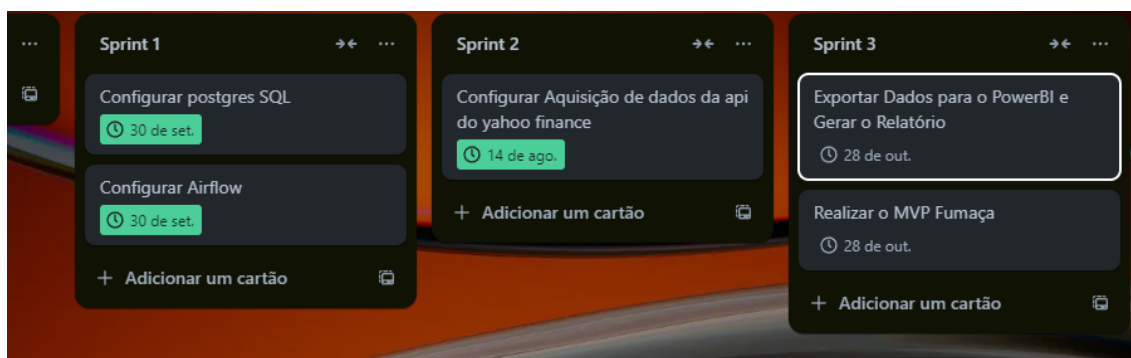
Adicionalmente o limite de 200 registros da api do yahoo finance pode ser considerado um limitante em alguns cenários.

2.3 Sprint 3

2.3.1 Solução

- Evidência do planejamento:

Para sprint 3, a ser entregue no dia 28/10/2024 temos as seguintes atividades:



Fonte: [Trello](https://trello.com/).

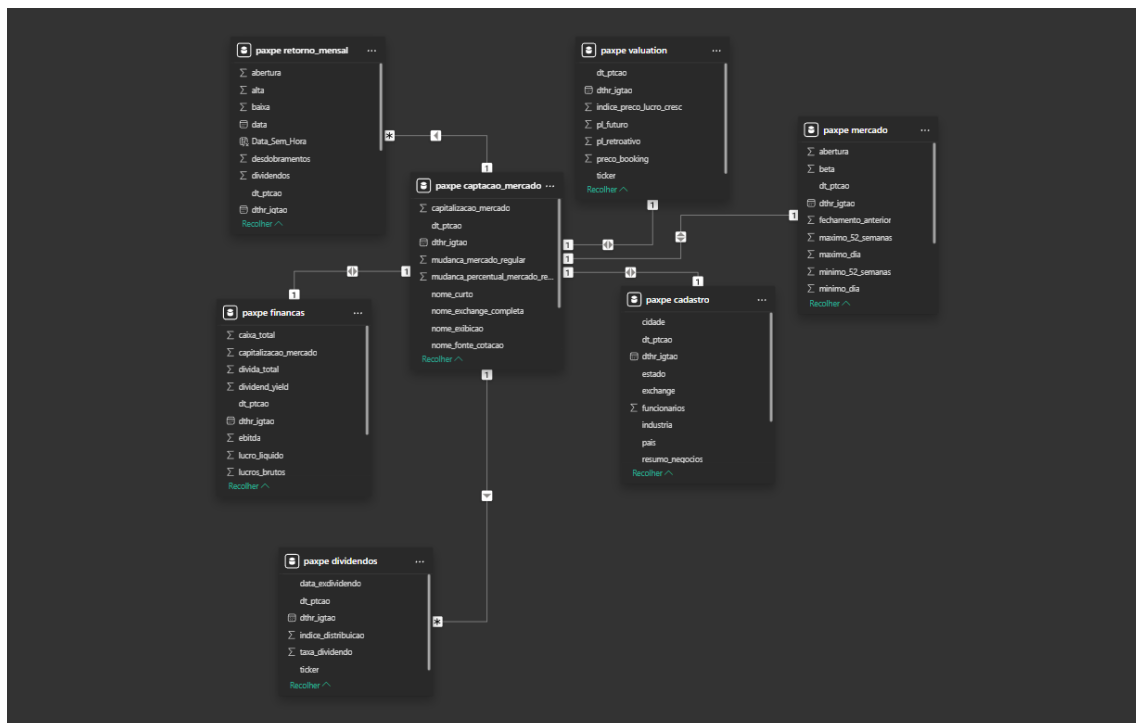
- Evidência da execução de cada requisito:

Exportar Dados para o PowerBI e Gerar o Relatório: Essa tarefa tem como objetivo realizar a primeira integração, de forma visual, a partir dos dados anteriormente aquisitados na sprint 2.

Para realizar esse desenvolvimento, realizaremos a aquisição dos dados do PostgreSQL para o powerbi via conexão direta, conforme demonstrado na disciplina de IDT.

Entretanto, por ser um trabalho de fins de engenharia e arquitetura os dados apresentados não possuem a finalidade de ser um relatório robusto e sim com índices-chave das empresas. Para demonstrar que o trabalho de engenharia proposto de integração de dados entre diferentes soluções foi realizado.

1. criação do vínculo com o PostgreSQL e relacionamento de tabelas:



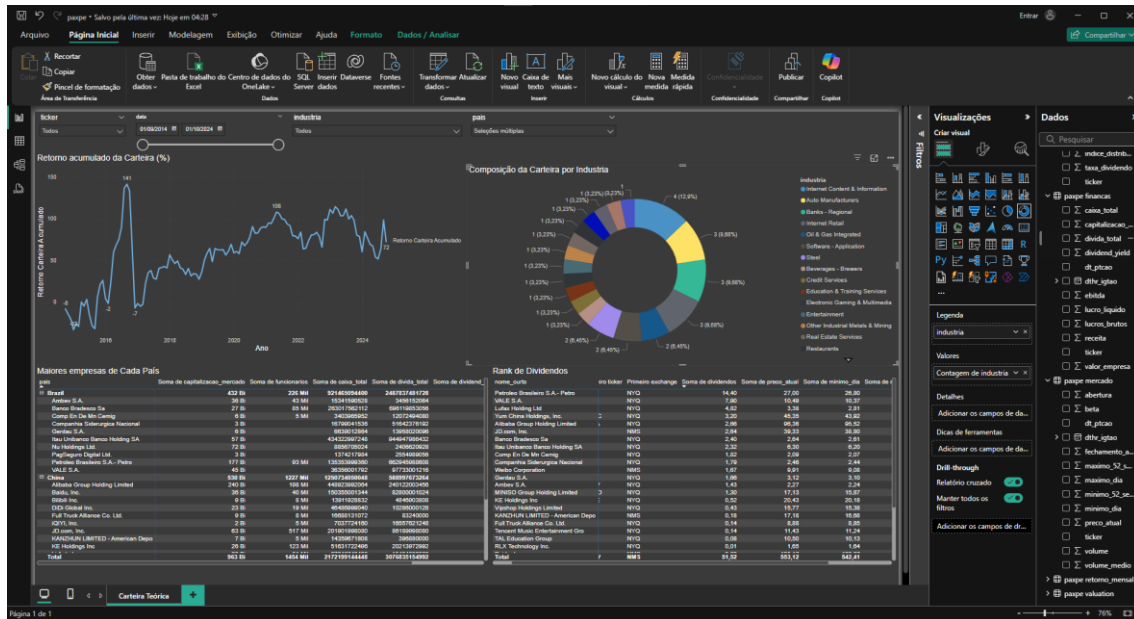
Cada tabela foi paralelizada para relacionamento pela chave da empresa, o ticker. O mesmo modelo de entidade relacional anteriormente apresentado na sprint 2.

Realizar o MVP Fumaça

Essa tarefa tem como objetivo testar a esteira de processo de forma efetiva. Com objetivo de executar o pipeline construído na sprint 2 e atualizar os dados no powerbi, para apresentação final do projeto.

Evidência dos resultados:

Relatório do powerbi construído:



Evidência de atualização dos dados (mvp fumaça):

Acionamento do airflow:



Log de sucesso do airflow



Consulta de atualização no banco de dados

temp_dividendos																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																				
-----------------	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Atualização de dados no powerbi

Consultas (7)

Fonte([Schema="pape",Item="retorno_mensal"])[Data]

		1.2 volume	1.2 dividendos	1.2 desdobramentos	1.2 fechamento_mes_anterior	1.2 valor_retorno	1.2 percentagem_retorno	1.2 dt_pisao	1.2 dt_liqua
10	454	3117990800	0	0	22,76490402	-0,638449478	-3,683079346	2024-08-28	25/10/2024 21:41:42
11	078	2743118400	0	0	21,92945454	1,974756241	9,006272477	2024-08-28	25/10/2024 21:41:42
12	965	2520514000	0,1425	0	23,90121078	0,433467865	1,813581198	2024-08-28	25/10/2024 21:41:42
13	763	3872062400	0	0	24,33467865	1,734498978	7,127683922	2024-08-28	25/10/2024 21:41:42
14	087	2747657200	0	0	26,08917763	6,11199324	0,433436152	2024-08-28	25/10/2024 21:41:42
15	139	440304200	0	0	101,5070859	0,619740498	0,402623395	2024-08-28	13/10/2024 18:32:23
16	743	77245600	0	0	105,288992	9,157875061	8,668082628	2024-10-24	25/10/2024 00:23:25
17	1284	87516000	0	0	114,8967243	-2,84545808	-1,98834642	2024-10-24	25/10/2024 00:23:25
18	1215	79446700	0	0	112,6122284	-4,716796875	-4,188529916	2024-10-24	25/10/2024 00:23:25
19	1432	742613800	0	0	113,7574844	3,469558716	3,049608829	2024-08-28	25/10/2024 03:06:17
20	1463	460544200	0	0	117,2270432	1,226303101	1,046092324	2024-08-28	25/10/2024 03:06:17
21	1507	450876100	0	0	118,4533463	-4,516395569	-8,812805389	2024-08-28	25/10/2024 03:06:17
22	1164	2886220000	0,1425	0	26,18217087	-0,696409225	-2,659880517	2024-08-28	25/10/2024 21:41:42
23	4035	2435088800	0	0	25,48576164	1,353238706	5,333561245	2024-08-28	25/10/2024 21:41:42
24	138	2257408800	0	0	26,84926035	1,817151633	4,774627496	2024-08-28	25/10/2024 21:41:42
25	1255	2329874400	0,1425	0	28,12681138	3,620061567	12,80839783	2024-08-28	25/10/2024 21:41:42
26	1693	2248313600	0	0	31,75190735	1,690349579	5,328611871	2024-08-28	25/10/2024 21:41:42
27	1852	1493216400	0	0	31,44225683	-0,002138439	-0,006992379	2024-08-28	25/10/2024 21:41:42
28	1554	2615927200	0,1575	0	31,43991852	2,120697021	6,343812766	2024-08-28	25/10/2024 21:41:42
29	1016	2736712400	0	0	35,56861554	-1,896175385	-5,332234438	2024-08-28	25/10/2024 21:41:42
30	9993	1688047600	0	0	33,66444016	1,100959778	3,703938842	2024-08-28	25/10/2024 21:41:42
31	1019	2077165200	0	0	36,16675186	3,501228333	9,68079286	2024-08-28	25/10/2024 21:41:42
32	1258	2402553600	0,1575	0	39,68798019	0,659412384	1,662329115	2024-08-28	25/10/2024 21:41:42
33	1761	2124755200	0	0	40,51739258	-0,472064972	-1,170581438	2024-08-28	25/10/2024 21:41:42
34	1403	232476200	0,75	0	60,16573324	0,421551254	0,701114689	2024-08-28	25/10/2024 21:41:42
35	1908	536462200	0	0	51,74449118	-2,5380625	-4,724121369	2024-08-28	25/10/2024 21:41:42
36	1228	128160200	0	0	114,2074738	-7,69055004	-6,733279652	2024-10-24	25/10/2024 00:23:25
37	1404	220315000	0	0	51,79669556	-0,632801514	-1,262508669	2024-08-28	25/10/2024 21:41:42
38	1381	334494700	0,42	0	51,05389494	-3,481555939	-6,819773926	2024-08-28	25/10/2024 21:41:42
39	012	286683400	0,42	0	51,49402237	-3,580921173	-6,954052157	2024-08-28	25/10/2024 21:41:42
40	1607	260737100	0	0	60,29085159	-2,170505524	-5,60005783	2024-08-28	25/10/2024 21:41:42

2.3.2 Lições Aprendidas

Por não ter muita vivência com a ferramenta, destaco a dificuldade inicial em trabalhar com o powerbi sem degradar a performance da distribuição dos dados no painel.

3. Considerações Finais

3.1 Resultados

Acredito que o projeto foi contemplado seguindo o desenho proposto na disciplina de inovação e design thinking, gostaria de destacar, que a iniciativa possuiu apenas um caminho pois não obtive ideias alternativas por ora.

Do lado positivo, destaco que pude atender o requisito construindo uma integração simples, usando apenas duas linguagens de programação comumente utilizadas como o python e o SQL. Como trabalho de engenharia entendo que os relacionamentos entre ferramentas e dados foram realizados conforme a proposta inicial.

Do lado negativo, destaco que estamos em soluções dentro da empresa, no mercado conhecido como “on-premisses”. Quer dizer que se reverte para a empresa a responsabilidade de sustentação/manutenção de ambientes em níveis de hardware e software, como patches de manutenção, atualização de OS e demais temas de administração de redes/bancos de dados/segurança corporativa. Cenário que facilmente poderia ser diminuído, porém não cessado, com o uso da computação em nuvem.

Por fim, finalizo acreditando que o projeto foi desenvolvido dentro dos requisitos acordados na proposta inicial e entendo ter sido uma experiência validadora para próximos projetos reais.

3.2 Contribuições

Acredito que a simplicidade de construção da integração entre as diferentes ferramentas para a finalidade foi atendida, diferentemente de um cenário com vários pipelines e integrações temos apenas 2 ou 3 ferramentas na arquitetura e com um nível de trabalho podendo ser menor do que em cenários mais complexos.

Sob um aspecto de dados, temos a possibilidade de consultar índices chaves de empresas do mundo inteiro, e não só dos estados unidos. Apesar de estarmos em custodiantes norte-americanos.



3.3 Próximos passos

Acredito que para os próximos passos o projeto poderia vir a ser enriquecido com a contratação de um analista de negócios para melhor insight nos relatórios construídos em dashboard, em um cenário de crescimento da empresa muitas outras iniciativas poderão surgir ao longo da jornada. Destaco o horizonte de cloud que pode vir a ser uma possibilidade de evolução técnica futuramente.

