UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

BSI Sistemas de Informação UFRPE

Arthur Henrique Martins Santos

Orquestração de Pipelines de Dados Educacionais: uma Arquitetura Automatizada com Apache Airflow e MinIO para Integração SIGAA-ENADE

Recife

2026

Arthur Henrique Martins Santos

Orquestração de Pipelines de Dados Educacionais: uma Arquitetura Automatizada com Apache Airflow e MinIO para Integração SIGAA-ENADE

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

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Estatística e Informática

Curso de Bacharelado em Sistemas de Informação

Orientador: Gabriel Alves

Recife

2026

Dedico este trabalho aos meus pais, cujo apoio e sacrifício pavimentaram o meu caminho.
E a mim mesmo, por ter tido a coragem de começar e a força inabalável para terminar.

Agradecimentos

Agradeço, primeiramente, à minha família, alicerce do meu percurso. Obrigado por sempre me apoiarem incondicionalmente e por construírem um lar que me proporcionou a paz e a estrutura necessárias para me dedicar aos estudos com serenidade.

À minha namorada, Karine, que cruzou o meu caminho durante a graduação e se tornou um dos meus maiores pilares. Seu apoio constante, sua parceria e suas inestimáveis dicas de escrita foram fundamentais para que este trabalho ganhasse forma.

Ao meu orientador, Professor Gabriel, por acreditar no potencial desta pesquisa e por me guiar com maestria. Agradeço imensamente pela confiança depositada em mim ao me dar a oportunidade de atuar como monitor do Observatório de Dados da Graduação (ODG), uma experiência que definiu meu percurso de carreira.

À Universidade Federal Rural de Pernambuco, pela estrutura, pelo corpo docente qualificado e pelas oportunidades que tornaram possível minha formação como Bacharel em Sistemas de Informação.

Aos meus amigos, que trouxeram leveza, alegria e companheirismo aos dias mais desafiadores, ajudando a moldar o profissional e a pessoa que sou hoje.

Por fim, expresso minha gratidão à própria vida, pela resiliência e pelas oportunidades que me permitiram chegar até aqui e celebrar o encerramento deste ciclo.

“O sonho é que leva a gente para a frente. Se a gente for seguir a razão, fica aquietado, acomodado.”

(Ariano Suassuna)

Resumo

A fragmentação e a dependência de processos manuais na integração de bases educacionais representam um desafio significativo para a governança de dados no ensino superior, aumentando o risco de falhas operacionais e inconsistências analíticas. Este trabalho propõe uma arquitetura de orquestração de pipelines de dados para automatizar a extração, transformação e carga (ETL) das bases consumidas pelo System of Academic Business Intelligence and Analytics (SABIA), plataforma institucional da Universidade Federal Rural de Pernambuco (UFRPE). A solução foi desenvolvida utilizando Apache Airflow para o gerenciamento de Grafos Acíclicos Direcionados (DAGs), Python e Pandas para o processamento, e o MinIO como repositório de objetos, estruturado em um modelo enxuto de camadas para separar dados brutos de dados processados em formato Parquet. O pipeline integra registros acadêmicos internos (sistemas SIGAA e SIGA) e o arquivo oficial de vagas da instituição com microdados públicos de avaliação (ENADE/INEP). A execução prática do projeto validou a conversão de 14 rotinas analíticas, outrora executadas manualmente, em um fluxo interdependente e concluído de forma automatizada em aproximadamente 7 minutos. Os resultados demonstram que a nova infraestrutura mitigou problemas de qualidade de dados como divergências de nomenclatura, valores ausentes e duplicidades, além de fornecer a rastreabilidade e a capacidade de recuperação de falhas exigidas por uma governança madura. Conclui-se que a padronização orquestrada não apenas tornou o abastecimento do SABIA confiável e independente de intervenção humana, como também consolidou um modelo arquitetural escalável e replicável para outras instituições que utilizam as mesmas infraestruturas de gestão acadêmica.

Palavras-chave: Engenharia de Dados. Governança de Dados. ETL. Apache Airflow. SIGAA. ENADE.

Abstract

The fragmentation and reliance on manual processes for integrating educational databases pose a significant challenge to data governance in higher education, increasing the risk of operational failures and analytical inconsistencies. This work proposes a data pipeline orchestration architecture to automate the extraction, transformation, and loading (ETL) of the databases consumed by the System of Academic Business Intelligence and Analytics (SABIA), an institutional platform of the Federal Rural University of Pernambuco (UFRPE). The solution was developed using Apache Airflow for managing Directed Acyclic Graphs (DAGs), Python and Pandas for processing, and MinIO as an object storage repository, structured in a lean layered model to separate raw data from processed data in Parquet format. The pipeline integrates internal academic records (SIGAA and SIGA systems) and the institution's official enrollment-capacity file with public assessment microdata (ENADE/INEP). The practical execution of the project validated the conversion of 14 previously manual analytical routines into an interdependent workflow completed automatically in approximately 7 minutes. The results show that the new infrastructure mitigated data quality issues such as naming discrepancies, missing values, and duplicates, while providing the traceability and failure-recovery capability required by mature data governance. It is concluded that orchestrated standardization not only made SABIA's data supply reliable and independent of human intervention, but also consolidated a scalable and replicable architectural model for other institutions that rely on the same academic management infrastructures.

Keywords: Data Engineering. Data Governance. ETL. Apache Airflow. SIGAA. ENADE.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

API	Application Programming Interface
BI	Business Intelligence
CPC	Conceito Preliminar de Curso
CSV	Comma-Separated Values
DAG	Directed Acyclic Graph (Grafo Acíclico Dirigido)
ELT	Extract, Load, Transform
ENADE	Exame Nacional de Desempenho dos Estudantes
ETL	Extract, Transform, Load
EWS	Early Warning Systems
IES	Instituições de Ensino Superior
IGC	Índice Geral de Cursos
INEP	Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira
LGPD	Lei Geral de Proteção de Dados Pessoais
MDE	Mineração de Dados Educacionais
S3	Simple Storage Service
SABIA	System of Academic Business Intelligence and Analytics
SGBD	Sistema de Gerenciamento de Banco de Dados
SIGAA	Sistema Integrado de Gestão de Atividades Acadêmicas
SINAES	Sistema Nacional de Avaliação da Educação Superior
SIS	Sistemas de Informação de Estudantes
SQL	Structured Query Language

Sumário

1 Introdução

A integração de diferentes fontes de dados representa um dos desafios na gestão de dados educacionais, tanto no cenário brasileiro quanto internacional. Grande parte dos dados produzidos por instituições de ensino permanece dispersa em sistemas legados e formatos pouco estruturados, o que compromete a interoperabilidade entre bases e eleva o custo de reutilização da informação (ALCANTARA et al., 2015). Esse cenário se intensifica à medida que cresce o volume de fontes a serem processadas, tornando os processos manuais de extração e consolidação insustentáveis a médio prazo (AMARE; SIMONOVA, 2020).

Além disso, estudos recentes em engenharia de software confirmam que pipelines de dados são propensos a falhas e de difícil manutenção. Em um estudo de caso qualitativo com profissionais de três empresas distintas, Munappy et al. (2020) identificaram problemas de qualidade de dados, manutenção de infraestrutura e barreiras organizacionais como os principais desafios enfrentados na implementação de pipelines, destacando que tais estruturas são adotadas justamente para viabilizar rastreabilidade, tolerância a falhas e redução de erros humanos. Ao investigar as causas-raiz de problemas relacionados a dados em projetos reais, Foidl et al. (2023) identificaram que a causa mais frequente está associada à tipagem incorreta dos dados (33% dos casos), concentrada majoritariamente nas etapas de limpeza (35%) e ingestão (34%) do pipeline. Além disso, tarefas de integração e ingestão de dados representam quase metade (47%) de todas as dúvidas técnicas levantadas por desenvolvedores, reforçando a integração de fontes como um dos pontos mais críticos na construção de pipelines confiáveis (FOIDL et al., 2023).

Quando esses problemas não são reconhecidos na etapa de engenharia, tendem a se manifestar sob a forma de silos de dados: bases relevantes que permanecem isoladas entre si, apesar do potencial analítico conjunto que poderiam oferecer. Bases públicas como os microdados do ENADE operam de forma desconectada dos Sistemas de Informação de Estudantes (SIS) internos das Universidades públicas, a exemplo do SIGAA (FINAMOR et al., 2022). Essa fragmentação limita severamente a utilização eficaz dos dados para a tomada de decisão institucional, refletindo o atraso de maturidade em governança de dados enfrentado pelas instituições de ensino superior (CERNEA; KIERYLO, 2024).

Para superar as limitações das extrações manuais, integrar esses silos de dados, processar e cruzar fontes distintas é necessário o uso de fluxos automatizados de Extração, Transformação e Carga (ETL/ELT) e armazenamentos centralizados (BRITO

et al., 2023; RIBEIRO et al., 2022). A estruturação de uma infraestrutura robusta, a exemplo de ambientes *Data Lakehouse* orquestrados por ferramentas modernas, possibilita não apenas a integração dessas fontes, mas a criação de uma fundação essencial para a aplicação de algoritmos preditivos e painéis de Business Intelligence (BI) em escala (ARMBRUST et al., 2021).

A substituição de rotinas manuais por pipelines orquestrados tem se consolidado como a abordagem padrão para garantir rastreabilidade. Em domínios como saúde (CEJUDO et al., 2025) e finanças (VANAM, 2025), a adoção de ferramentas como o Apache Airflow e arquiteturas em camadas (como a medallion) demonstrou reduzir substancialmente a latência de processamento e as falhas operacionais. Contudo, enquanto esses setores avançam na automação de seus fluxos de dados, a governança de dados no ensino superior fica para trás: Cernea e Kierylo (2024) apontam que instituições de ensino superior ainda enfrentam um atraso significativo na maturidade de governança de dados em relação a outros setores, como o financeiro e o de saúde, o que reforça a necessidade de "frameworks"e infraestruturas mais robustas de gestão de dados também no contexto acadêmico.

Esse cenário de defasagem reflete diretamente os desafios enfrentados pela Universidade Federal Rural de Pernambuco (UFRPE), essa limitação arquitetural afeta diretamente o System of Academic Business Intelligence and Analytics (SABIA). Embora a plataforma cumpra o papel de visualizar indicadores acadêmicos, sua camada de alimentação não é automatizada. O fluxo atual de integração de fontes para o SABIA (como o SIGAA e os microdados do ENADE) é composto por rotinas de Extração, Transformação e Carga (ETL) desenvolvidas na forma de *scripts* isolados. Na prática, cada etapa do processamento reside em um arquivo de código separado. Como há uma relação de dependência lógica e temporal entre eles (por exemplo, a unificação das bases só pode ocorrer após a limpeza individual de cada fonte), um desenvolvedor precisa executar um *script* manualmente, aguardar sua finalização, conferir a ausência de erros e, só então, acionar o próximo arquivo. Essa necessidade de intervenção humana sequencial demanda tempo excessivo, dificulta a manutenção e aumenta significativamente o risco de falhas operacionais.

Diante desse cenário, este trabalho tem como objetivo aprimorar a camada de engenharia de dados do SABIA. Propõe-se a estruturação de um pipeline orquestrado com Apache Airflow que integre e automatize as rotinas modulares já existentes para o SIGAA e os microdados do ENADE, com armazenamento segregado entre dados brutos e processados no MinIO. Ao converter *scripts* isolados em Grafos Acíclicos Direcionados (DAGs), o projeto visa eliminar a dependência do acionamento humano, gerenciando automaticamente as relações de precedência temporal e garantindo a atualização recorrente, íntegra e rastreável das bases analíticas (em formato Parquet)

consumidas pelo sistema.

1.1 Problema e Questões de Pesquisa

Para nortear o desenvolvimento da orquestração estruturada neste trabalho, foram elaboradas as seguintes Questões de Pesquisa (RQs):

- **RQ1:** Como a transição de *scripts* ETL de execução manual para uma arquitetura baseada em Grafos Acíclicos Direcionados (DAGs) impacta o tempo de atualização e a confiabilidade das bases do sistema SABIA (medida pela redução na taxa de falhas de execução e indisponibilidade)?
- **RQ2:** Quais adaptações técnicas são necessárias nas rotinas isoladas existentes (fontes SIGAA e ENADE) para viabilizar sua ingestão e tratamento de forma automatizada, respeitando a sua interdependência (onde a execução de uma tarefa depende do sucesso de outra)?
- **RQ3:** De que forma a integração do Apache Airflow provê mecanismos de rastreabilidade, auditoria e recuperação de falhas operacionais que não existiam no modelo anterior?

1.2 Objetivo Geral

Desenvolver uma arquitetura de orquestração de dados utilizando Apache Airflow para automatizar a ingestão, transformação e carga das bases educacionais da UFRPE, eliminando a execução manual de *scripts* e com o intuito de promover maior escalabilidade e facilitar a manutenibilidade da camada de dados do sistema SABIA.

1.2.1 Objetivos Específicos

- Mapear e refatorar as rotinas ETL modulares atuais, estruturando-as como DAGs para gerenciar corretamente as relações de precedência entre as extrações do SIGAA e do ENADE.
- Efetivar a integração das fontes de dados, desenvolvendo etapas no pipeline que cruzem e unifiquem as bases do SIGAA e do ENADE em um repositório centralizado, viabilizando análises conjuntas.
- Implementar o pipeline automatizado que padroniza o tratamento de qualidade de dados (divergências de nomenclatura, valores ausentes e duplicidades) gerando os artefatos finais integrados em formato Parquet.

- Configurar alertas, rotinas de *logs* e mecanismos de monitoramento das execuções, garantindo feedback técnico ágil em caso de quebra do pipeline.
- Projetar a arquitetura de DAGs de forma desacoplada, de modo a permitir a incorporação de novas fontes de dados institucionais sem a necessidade de reestruturação do fluxo já construído.

1.3 Justificativa

Do ponto de vista acadêmico, este trabalho se justifica pelo atraso de maturidade em governança de dados identificado no ensino superior frente a setores mais avançados nessa temática, como a área da saúde e o mercado financeiro (CERNEA; KIERYLO, 2024). Ao aplicar ao SABIA da UFRPE um padrão arquitetural já consolidado em domínios como saúde (CEJUDO et al., 2025) e finanças (VANAM, 2025), este trabalho contribui com um caso concreto de aplicação de orquestração de dados no contexto de instituições públicas de ensino superior brasileiras.

Do ponto de vista institucional, a infraestrutura proposta elimina a dependência de intervenção humana (acionamento manual repetitivo) para a atualização dos dados do sistema. Ao orquestrar o processo, o pipeline garante que a limpeza, a validação e a integração dos dados sigam regras de negócio rigorosas e padronizadas de forma contínua. Isso assegura que os painéis do SABIA reflitam métricas acadêmicas atualizadas e íntegras, oferecendo suporte confiável à tomada de decisão pela gestão da universidade.

1.4 Organização do Trabalho

O restante deste documento está organizado da seguinte forma: o Capítulo 2 apresenta os Trabalhos Relacionados em integração e orquestração de dados educacionais e domínios correlatos. O Capítulo 3, Fundamentação Teórica, aborda as bases da Engenharia de Dados, processos de ETL e ferramentas de orquestração. O Capítulo 4 descreve os Métodos empregados na conversão das rotinas para DAGs e a estruturação da arquitetura. O Capítulo 5 apresenta os Resultados obtidos com a implementação do pipeline e traz a Discussão dos Resultados, cruzando os dados técnicos operacionais (como tempo de execução e redução de intervenção) para responder às Questões de Pesquisa (RQs). Por fim, o Capítulo 6 apresenta as Conclusões e caminhos para trabalhos futuros.

2 Trabalhos Relacionados

A literatura recente demonstra um esforço crescente na aplicação de Mineração de Dados Educacionais (MDE) e *Learning Analytics* para mitigar a evasão e analisar o desempenho de estudantes no ensino superior. Em um mapeamento sistemático sobre o tema, conduzido a partir da produção do principal congresso brasileiro de Informática na Educação, Colpo et al. (2020) evidenciam os desafios metodológicos e as lacunas na aplicação de técnicas de MDE voltadas à previsão de evasão no Brasil. Alinhados a essa visão, estudos como o de Andrade et al. (2023) realizaram um mapeamento sistemático evidenciando que a manipulação de dados de desempenho acadêmico (como notas e aprovações em disciplinas) e dados demográficos é fundamental para a criação de painéis analíticos e modelos preditivos. Mais recentemente, Andrade et al. (2025) demonstraram a eficácia de sistemas alimentados por bases de dados acadêmicos para embasar a tomada de decisão de gestores e professores, integrando a análise de dados a metodologias pedagógicas para evitar a retenção. Reforçando essa premissa, Silva (2025) desenvolveu um pipeline otimizado de *Machine Learning* para a detecção de evasão escolar e demonstrou que a precisão dessas análises depende intrinsecamente de uma rigorosa engenharia de atributos e do pré-processamento de dados acadêmicos dinâmicos, como a evolução de notas e o ritmo de aprovação dos alunos ao longo dos semestres.

Avançando para as infraestruturas capazes de suportar essa modelagem preditiva em larga escala, Kaphle e Shrestha (2026) propuseram uma arquitetura de *Big Data* orientada a *Lakehouse* desenhada especificamente para *Educational Analytics*. Para modernizar esses fluxos, Ramos et al. (2022) destacam a necessidade de transicionar de sistemas administrativos tradicionais e fragmentados para arquiteturas de *Data Lake* lógico, capazes de integrar múltiplas fontes governamentais heterogêneas sob uma camada única de armazenamento, garantindo maior governança e viabilizando consultas analíticas que seriam inviáveis nos sistemas de origem isolados. O estudo reforça que uma camada de dados estruturada e centralizada é o que viabiliza a criação de sistemas de alerta e monitoramento mais responsivos.

Para que todas essas análises sejam possíveis, a etapa de integração e pré-processamento dos dados é um fator crítico. No trabalho de Correia et al. (2024), focado na identificação de padrões de evasão a partir do cruzamento de dados demográficos com históricos de disciplinas e avaliações nacionais como ENEM e ENADE, os autores relatam que a extração dessas informações ocorreu de forma manual, através de consultas SQL diretas no banco de dados de produção da instituição, resultando em planilhas não otimizadas (arquivos XLS e CSV) que precisaram passar por limpeza,

tratamento de *outliers* e padronização manual. O caso ilustra como a ausência de automação nessa etapa não compromete apenas a agilidade, mas também a qualidade das bases usadas em análises subsequentes.

Para superar esse gargalo das extrações manuais, a implementação de sistemas de *Business Intelligence* (BI) e *Data Warehouses* educacionais torna-se imprescindível. Laksitowening et al. (2025), por exemplo, detalharam a construção de um modelo dimensional através de uma abordagem *Bottom-Up*, integrando dados dispersos de Sistemas de Gestão de Aprendizagem (LMS) e sistemas acadêmicos administrativos. Os autores ilustram como a execução prática de um processo automatizado de Extração, Transformação e Carga (ETL) resolve conflitos de granularidade e consolida indicadores educacionais confiáveis, substituindo com sucesso a manipulação manual de planilhas.

A evolução teórica e tecnológica dessas abordagens de integração de dados é profundamente analisada por Mahmud e Ikbal (2022). Em sua revisão sistemática sobre o papel dos pipelines ETL na escalabilidade de sistemas de BI, os autores mapeiam a transição dos processamentos em lote tradicionais para paradigmas modernos, como ELT e processamento em nuvem. O estudo evidencia a necessidade de adotar ferramentas de código aberto flexíveis para lidar com a alta volumetria e heterogeneidade das fontes modernas, corroborando as escolhas de arquitetura em cenários de dados complexos.

No contexto brasileiro, a necessidade de enriquecer bases acadêmicas institucionais com avaliações nacionais já foi discutida por Correia et al. (2024). Buscando resolver os gargalos de processamento dessas bases governamentais volumosas, Gomes et al. (2017) evidenciaram a necessidade de utilizar ferramentas robustas de ETL para extrair, transformar e limpar os dados do ENADE. O processo de mineração destas bases específicas tem sido amplamente documentado; Cunha, Sales e Santos (2021), por exemplo, detalham um processo prático e automatizado de extração e análise dos microdados do ENADE para os cursos de Ciência da Computação da UFPA, fornecendo subsídios diretos para a melhoria pedagógica dos cursos. A relevância analítica dessa integração é reforçada por Kantorski et al. (2023), que combinaram técnicas de aprendizagem de máquina com uma ferramenta de *Business Intelligence* para prever a evasão em 106 cursos de graduação presenciais, evidenciando o valor de bases de desempenho acadêmico consolidadas para a tomada de decisão institucional. Contudo, ao cruzar bases públicas com dados internos, surge o desafio da governança: Almeida, Braganholo e Oliveira (2025) apontam que, em Sistemas-de-Sistemas que integram fontes independentes, a ausência de mecanismos de controle do ciclo de vida dos dados compromete a rastreabilidade e a integridade das informações desde a coleta até o armazenamento, exigindo um tratamento cuidadoso na etapa de ingestão. Para enfrentar essas mesmas dificuldades de escala, Santos e Fuini (2024) propuseram uma

arquitetura em nuvem (utilizando serviços AWS e scripts em Python) para automatizar a rotina de tratamento dos microdados do ENADE, resolvendo problemas complexos de tipagem para a alimentação de *dashboards*.

Visando escalar e orquestrar de forma sustentável essas automações, trabalhos atuais têm consolidado o uso do Apache Airflow no setor público, industrial e institucional brasileiro. Na prática, Sousa (2025) propôs a arquitetura *AutoMicroETL*, utilizando o Airflow integrado a um repositório estruturado em camadas (arquitetura medalhão: bronze, prata e ouro) para orquestrar a extração e a padronização dos microdados brutos do ENEM. Em um contexto análogo, Salles (2025) evidenciou a eficácia do Airflow ao orquestrar fluxos complexos de ETL para dados de compras da Prefeitura de Florianópolis, enquanto Bezerra et al. (2025) propuseram uma arquitetura de *Data Lake* distribuída e de baixo acoplamento, orquestrada com Airflow, Hadoop e Spark, para a coleta, o armazenamento e o processamento de dados legados heterogêneos em um ecossistema de dados industrial — cenário próximo, em heterogeneidade e volume de fontes, ao enfrentado pelo SABIA com o SIGAA, o SIGA e o INEP. Avançando na governança dessas arquiteturas, Silva et al. (2025) propõem um modelo de linhagem de dados baseado em metadados padronizados (Dublin Core) para rastrear operações de ETL entre fontes heterogêneas, reforçando a importância da rastreabilidade também discutida por Almeida, Braganholo e Oliveira (2025). Em linha semelhante, Linhares (2025) apresentou uma arquitetura modular de engenharia de dados fundamentada em *Data Lakehouse* e em princípios de *Domain-Driven Design*, aplicada à consolidação automatizada de dados públicos heterogêneos para alimentar modelos preditivos, reforçando que o padrão arquitetural adotado neste trabalho — camadas de dados brutos e processados orquestradas por pipelines automatizados — vem sendo replicado em TCCs brasileiros recentes mesmo fora do domínio educacional.

Neste cenário, o presente trabalho se diferencia e complementa os trabalhos aqui mencionados ao propor a modernização e a integração de ponta a ponta dessa arquitetura de dados educacionais. Enquanto os trabalhos sobre evasão focam nos modelos analíticos dependentes de extrações manuais, e os estudos de engenharia de dados (Gomes et al. (2017), Santos e Fuini (2024), Sousa (2025)) processam bases públicas de forma isolada, este trabalho preenche a lacuna da orquestração unificada entre fontes institucionais internas (SIGAA, SIGA) e públicas (ENADE), contribuindo para reduzir o atraso de maturidade em governança de dados no ensino superior discutido na Introdução deste trabalho. A utilização do Apache Airflow para orquestrar o processo de ETL dos dados internos do SIGAA (UFRPE) automatiza a limpeza descrita como custosa na literatura. Ademais, a agregação proposta com os dados do ENADE atende à demanda acadêmica por cruzamento de métricas nacionais e institucionais, consolidando essas informações estruturadas em arquivos Parquet (base para um ambiente *Lakehouse*). Essa escolha arquitetural corrobora os achados

empíricos de Gouvêa (2025) e Linhares (2025), garantindo a alta performance, a resiliência e a escalabilidade necessárias para que a área analítica da instituição possa focar exclusivamente na geração de valor, como a análise de sobrevivência e o monitoramento preciso do risco discente.

3 Fundamentação Teórica

Este capítulo apresenta os conceitos que fundamentam o trabalho, partindo da definição de Engenharia de Dados e dos paradigmas ETL e ELT. Em seguida, abordam-se as arquiteturas de Data Lake e Medallion, bem como o uso do MinIO e do formato Parquet para armazenamento. Por fim, discutem-se as premissas de governança de dados e o contexto das fontes institucionais e governamentais que alimentam a plataforma SABIA.

3.1 Engenharia de Dados

A Engenharia de Dados é a área responsável por projetar, construir e manter os sistemas que viabilizam a coleta, o armazenamento e a disponibilização dos dados. Diferentemente da Ciência de Dados, cujo foco está na extração de *insights* a partir das informações expostas, a Engenharia de Dados foca na infraestrutura e nos processos que tornam os dados íntegros e prontos para serem consumidos por um sistema ou tabelas.

Como dito anteriormente, a implementação incorreta dessa infraestrutura pode acarretar gargalos diretos. Como apontam Munappy et al. (2020), a manutenção da infraestrutura e a qualidade dos dados são desafios centrais na gestão de pipelines corporativos, enquanto Foidl et al. (2023) discorrem que problemas de tipagem e falhas nas etapas de ingestão e limpeza respondem pela maioria dos incidentes em projetos de dados. Essas fontes demonstram como a Engenharia de Dados, bem feita, é imprescindível para a confiabilidade de qualquer sistema analítico, como é o caso do SABIA.

O principal mecanismo dessa disciplina é o pipeline de dados, uma sequência automatizada que extrai, transforma e entrega a informação, geralmente implementada sob o paradigma ETL ou ELT.

3.2 Processos ETL (*Extract, Transform, Load*)

A integração entre as bases públicas do INEP e os dados do SIGAA exige um fluxo estruturado, como por exemplo, o ETL (*Extract, Transform, Load*). Segundo Singu (2022), o ETL é um processo padrão para consolidar as informações de maneira adequada. Estratégias semelhantes são aplicadas na organização de dados públicos brasileiros, como no pipeline desenvolvido por Neto, Neres e Carvalho (2024) para as

bases do IBGE.

Conforme a literatura, o processo de ETL é composto por três etapas:

- **Extração (Extract):** Coleta dos dados a partir de fontes heterogêneas (bancos relacionais, APIs, planilhas) e sua alocação em uma área de transição temporária (*staging area*).
- **Transformação (Transform):** Etapa de limpeza e padronização. É onde divergências de nomenclatura são corrigidas, valores nulos são tratados e regras de negócio são aplicadas para garantir a qualidade do dado bruto.
- **Carga (Load):** Escrita dos dados já tratados no repositório final (como um *Data Lake* ou *Data Warehouse*), disponibilizando-os para o consumo por ferramentas de Business Intelligence (BI).

3.2.1 ELT (*Extract, Load, Transform*) e a escolha pelo ETL

Uma variação do ETL é o ELT (*Extract, Load, Transform*), no qual os dados são carregados em sua forma bruta no destino de armazenamento, para só então serem transformados dentro desse próprio ambiente, tipicamente aproveitando o poder computacional de um *Data Warehouse* ou *Data Lake* moderno (SINGHAL; AGGARWAL, 2022). Enquanto o ETL realiza a transformação antes do carregamento, restringindo o volume e a granularidade dos dados armazenados, o ELT preserva os dados brutos junto aos dados transformados, favorecendo cenários de reprocessamento e auditoria.

Apesar dessa vantagem analítica do ELT, o presente trabalho optou por manter a abordagem tradicional do ETL. A decisão se justifica por uma exigência técnica e legal: as transformações aplicadas ao SIGAA precisam ocorrer antes que a informação seja persistida no servidor. Essa manipulação prévia garante a conformidade do projeto com a Lei Geral de Proteção de Dados (LGPD), tema aprofundado na Seção 3.8.

3.3 *Data Lake*, *Data Lakehouse* e Arquitetura *Medallion*

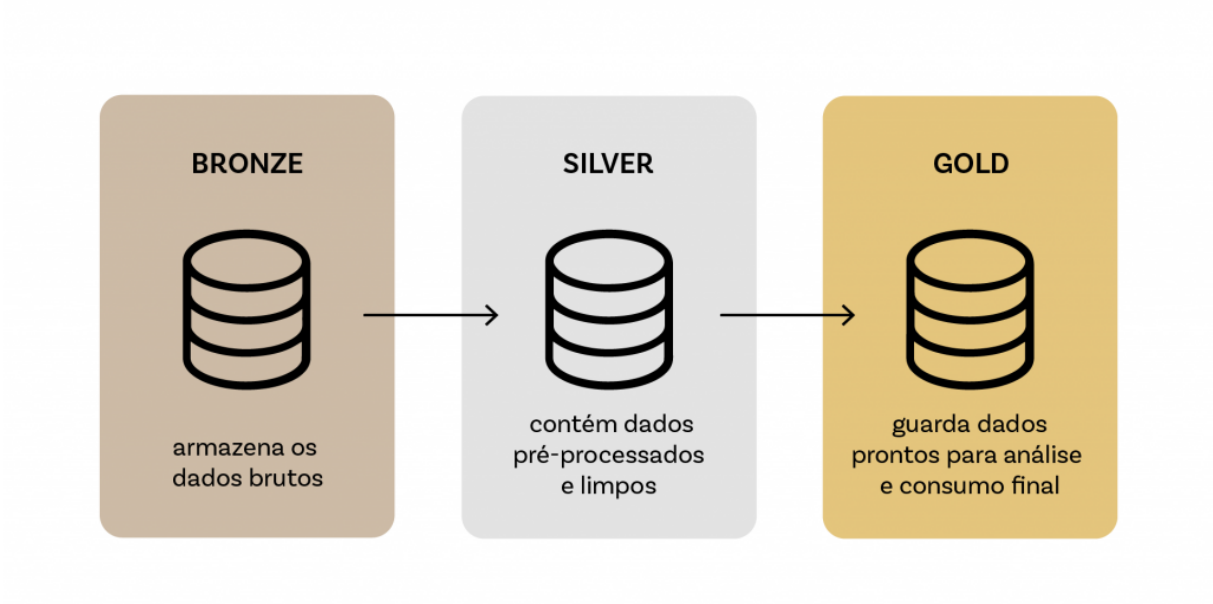
Um *Data Lake* é um repositório centralizado para armazenamento de dados em formato bruto e nativo. Ao dispensar a definição prévia de esquema (*schema-on-read*), ele oferece mais flexibilidade do que um *Data Warehouse* tradicional. No entanto, sem uma governança adequada, essa mesma flexibilidade pode transformar o repositório em um "*data swamp*"(pântano de dados), marcado pela desorganização e perda de confiabilidade da informação.

Buscando mitigar esse risco e manter o baixo custo do *Data Lake*, o mercado adotou a arquitetura *Data Lakehouse*. Ela traz funcionalidades de gerenciamento

típicas de um *Data Warehouse* — como transações ACID, versionamento e controle de esquema — operando diretamente sobre formatos de arquivo abertos, como o Parquet (ARMBRUST et al., 2021). No Brasil, modelos semelhantes já estruturam dados abertos complexos, a exemplo da consolidação de bases semânticas da Receita Federal (ROLIM et al., 2024).

Dentro do *Lakehouse*, a organização mais comum é a arquitetura *Medallion*. Ela estrutura o fluxo em camadas lógicas de refinamento progressivo em *bronze*, *prata* (*silver*) e *ouro* (*gold*) como pode ser visto na Figura 1, onde o dado avança cada vez mais limpo e pronto para consumo (SALAMI, 2025). Este projeto adota a premissa principal dessa arquitetura (a separação estrita entre dados brutos e processados), porém em um modelo simplificado e adaptado para as necessidades do pipeline, conforme detalhado na Seção 3.4.

Figura 1 – Diagrama da Arquitetura de Medalhão e suas camadas



Fonte: Statplace (2024).

3.4 *Object Storage* e MinIO

Os métodos tradicionais de armazenamento, como sistemas de arquivos em disco (*File System*) e tabelas relacionais, costumam apresentar gargalos de escalabilidade e custo computacional elevado em cenários analíticos de grande volume (ARMBRUST et al., 2021). Como alternativa, o modelo de *Object Storage* (Armazenamento de Objetos) guarda os arquivos de forma independente dentro de contêineres lógicos (*Buckets*). Cada objeto possui um identificador único e metadados customizá-

veis, o que facilita o particionamento horizontal e o processamento distribuído (Amazon Web Services, 2024).

O MinIO é um servidor *open-source* de *Object Storage* de alto desempenho, compatível nativamente com a API S3 da AWS (MinIO, 2024). Escolhido como o repositório principal deste projeto, o MinIO permite organizar o *Data Lake* por meio de diretórios segregados e gerenciar políticas de acesso baseadas em chaves de segurança. Sua aplicação técnica em pipelines acadêmicos e arquiteturas de Big Data já foi validada em iniciativas nacionais, como no arcabouço ORBITER (LOUREIRO; OLIVEIRA, 2022) e no sistema oceanográfico SADD (MONTEIRO et al., 2025).

Essa estrutura aplica a lógica de refinamento da arquitetura *Medallion* (SALAMI, 2025) ao separar fisicamente o dado cru (equivalente à camada *bronze*) do dado processado. No entanto, trata-se de uma adaptação mais enxuta: não há segregação em pastas distintas para os estágios intermediário (*silver*) e final (*gold*). Ambos coexistem no diretório `output`, sendo identificados exclusivamente por convenção de nomenclatura. Essa decisão arquitetural evitou a adição de complexidade desnecessária, mantendo a operação do pipeline aderente à escala dos dados da universidade.

3.5 Formato Parquet

O Apache Parquet é um formato de armazenamento de dados de código aberto, orientado a colunas (*columnar*), que se consolidou como padrão em arquiteturas analíticas de larga escala (VOHRA, 2016). Diferentemente de formatos orientados a linha, como CSV, nos quais uma consulta que utiliza poucas colunas ainda exige a leitura de todos os registros da tabela, o Parquet armazena os valores de cada coluna de forma contígua em disco. Essa organização permite duas otimizações centrais para cargas de trabalho analíticas: a leitura seletiva de colunas (*column pruning*), que reduz a quantidade de dados lidos do disco, e a aplicação de esquemas de codificação e compressão mais eficientes, uma vez que valores de uma mesma coluna tendem a apresentar maior similaridade entre si.

Além do ganho de desempenho, o Parquet incorpora o próprio esquema dos dados dentro do arquivo, o que garante a tipagem correta das colunas no momento da leitura — mitigando diretamente um dos problemas mais recorrentes identificados na literatura de pipelines de dados, a saber, a tipagem incorreta de dados na etapa de ingestão (FOIDL et al., 2023). Por esses motivos, o formato foi adotado como padrão de saída para os artefatos finais gerados pelo pipeline proposto neste trabalho, sendo consumido diretamente pelos painéis do SABIA.

3.6 Qualidade e Governança de Dados

A literatura define a qualidade de dados como "adequação ao uso"(*fitness for use*), ou seja, a capacidade de uma base de dados atender às necessidades práticas de quem a consome (WANG; STRONG, 1996). Mais do que ausência de erros técnicos, a qualidade envolve dimensões relacionais como acurácia, completude e consistência. A falta dessas características compromete diretamente a confiabilidade operacional de plataformas institucionais, fenômeno já documentado por Espíndola et al. (2018) na análise de sistemas de saúde — um cenário com desafios análogos aos enfrentados pelas bases acadêmicas do SABIA.

Por outro lado, a Governança de Dados engloba as políticas e processos que asseguram o ciclo de vida da informação. Se a qualidade avalia o estado do dado em si, a governança estabelece as regras de acesso, a padronização e a responsabilização técnica. O atraso na maturidade dessas políticas estruturais é, segundo Cernea e Kierylo (2024), um dos principais gargalos das instituições de ensino superior quando comparadas a setores corporativos mais orientados a dados.

No escopo desta pesquisa, esses dois conceitos são aplicados na prática em duas frentes do pipeline: (i) na fase de transformação, através da limpeza de divergências, tratamento de valores ausentes e remoção de duplicidades nas fontes do SIGAA e do ENADE; e (ii) na orquestração pelo Apache Airflow, que introduz rotinas de *logging*, monitoramento e recuperação de falhas. Essa camada de controle garante a rastreabilidade e a auditabilidade exigidas por uma governança madura, respondendo diretamente à RQ3 do presente trabalho.

3.7 Dados Abertos e o Exame Nacional de Desempenho dos Estudantes (ENADE)

O Exame Nacional de Desempenho dos Estudantes (ENADE) constitui-se como um dos principais instrumentos de avaliação do Sistema Nacional de Avaliação da Educação Superior (SINAES), instituído pela Lei nº 10.861, de 14 de abril de 2004 (BRASIL, 2004). O exame tem como premissa avaliar o rendimento dos estudantes concluintes dos cursos de graduação, mensurando a absorção de conteúdos programáticos das diretrizes curriculares, bem como o desenvolvimento de habilidades e competências necessárias para a compreensão de temas ligados à realidade brasileira e mundial (Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira – INEP, 2024).

Nesse contexto, os resultados gerados pelo exame, juntamente com o Conceito Preliminar de Curso (CPC) e o Índice Geral de Cursos (IGC), são disponibilizados publicamente pelo Instituto Nacional de Estudos e Pesquisas Educacionais Anísio

Teixeira (INEP) sob a política de Dados Abertos Governamentais. Do ponto de vista da Engenharia de Dados, o acesso a esses microdados representa uma oportunidade para as Instituições de Ensino Superior (IES) construírem diagnósticos sobre sua eficiência.

3.8 Os Sistemas Acadêmicos (SIGAA e SIGA) e a Lei Geral de Proteção de Dados Pessoais (LGPD)

Para complementar as bases agregadas do INEP, a análise acadêmica institucional exige o consumo de dados de sistemas transacionais. Na Universidade Federal Rural de Pernambuco (UFRPE), essa base interna é formada tanto pelo atual Sistema Integrado de Gestão de Atividades Acadêmicas (SIGAA) quanto pelo sistema legado anterior, conhecido como SIGA. Ambos armazenam informações detalhadas sobre o ciclo de vida do estudante, incluindo cadastros, matrículas, frequências e histórico escolar (Universidade Federal do Rio Grande do Norte – UFRN, 2024). O uso conjunto das duas bases no projeto garante que a trajetória histórica da instituição não seja perdida por causa da transição tecnológica de sistemas.

Como as tabelas dessas plataformas contêm dados pessoais identificáveis, a extração e o cruzamento dessas informações estão diretamente submetidos à Lei Geral de Proteção de Dados Pessoais (Lei nº 13.709/2018) (BRASIL, 2018). No contexto do pipeline desenvolvido neste trabalho, tanto os dados mais recentes do SIGAA quanto os históricos mais antigos do SIGA passam pelo mesmo processo de tratamento. A conformidade legal é garantida por meio de técnicas de anonimização aplicadas ainda na etapa de transformação. Essa trava de segurança assegura que os arquivos Parquet finais entreguem ao SABIA exclusivamente agregações estatísticas e identificadores ofuscados, impossibilitando tecnicamente a reidentificação dos discentes nos *dashboards* (BRASIL, 2018).

3.9 O SABIA: *System of Academic Business Intelligence and Analytics*

O SABIA é uma plataforma de Business Intelligence (BI) e Analytics desenvolvida na UFRPE com o objetivo de subsidiar a gestão baseada em evidências nas Instituições de Ensino Superior (IES) (MARQUES et al., 2023). A ferramenta integra conceitos de *Learning Analytics* (LA), voltado ao desempenho de estudantes em nível de sala de aula, e *Academic Analytics* (AA), que amplia esse escopo para a esfera institucional. Através dessa abordagem, a plataforma disponibiliza painéis dinâmicos que cobrem desde o detalhamento de cursos e disciplinas específicas até métricas globais da instituição (MARQUES et al., 2023).

Para fornecer essa visão institucional holística, os painéis são alimentados por indicadores educacionais de referência (Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira – INEP, 2024). Entre eles, destacam-se o **Índice Geral de Cursos (IGC)**, que sintetiza a qualidade global da instituição de ensino; o **Conceito Preliminar de Curso (CPC)**, indicador de qualidade que avalia os cursos de graduação considerando o desempenho dos alunos e a infraestrutura; e o **Indicador de Diferença entre Desempenhos (IDD)**, que mensura o valor agregado pelo curso ao comparar os resultados dos alunos ingressantes e concluintes. Além dos indicadores do INEP, a plataforma incorpora métricas de gestão interna como a **Taxa de Sucesso da Graduação (TSG)**, que calcula o percentual de discentes que concluem seus cursos no tempo regular esperado.

Do ponto de vista de engenharia, a plataforma foi construída seguindo boas práticas de projetos de BI, com cada painel consumindo dados de um *data mart* próprio, o que simplifica tanto o carregamento quanto a manutenção individual de cada visualização. A alimentação desses *data marts* segue os princípios de pipelines ETL, contemplando etapas de ingestão (a partir de fontes como o Censo da Educação Superior, os resultados do ENADE e os sistemas internos de gestão acadêmica), limpeza, transformação e carregamento (MARQUES et al., 2023). Por se tratar de indicadores calculados na granularidade de desempenho de disciplinas por semestre, a atualização dos dados não demanda processamento em tempo real, sendo executada em lote — hoje, de forma manual, ao menos uma vez por semestre.

É justamente essa característica manual da camada de alimentação que constitui o objeto deste trabalho: embora o SABIA já esteja em pleno funcionamento em uma IES pública federal, auxiliando gestores no acompanhamento de indicadores institucionais e na tomada de decisões como o cálculo de vagas em processos seletivos, a ausência de automação na atualização de seus *data marts* compromete a agilidade e a confiabilidade do processo, motivando a proposta de orquestração descrita neste trabalho.

4 Métodos

Este capítulo descreve os procedimentos técnicos adotados no desenvolvimento do pipeline, aplicando os conceitos discutidos no Capítulo anterior à automatização da extração, tratamento e disponibilização dos registros acadêmicos do SABIA. O projeto implementa o fluxo ETL apresentado na Seção 3.2, orquestrado pelo Airflow, integrando os dados do SIGAA/SIGA da UFRPE com os microdados públicos do INEP (ENADE). A seguir, são detalhadas as fontes utilizadas, as tecnologias escolhidas e o funcionamento de cada etapa do pipeline.

4.1 Visão Geral da Arquitetura do Sistema

O desenho geral do projeto consolida a separação física entre dados brutos e tratados, premissa arquitetural discutida na Seção 3.3. O fluxo operacional funciona de forma linear e centralizada. A Figura 2 apresenta a visão consolidada da arquitetura, destacando o fluxo entre SABIA, Airflow, MinIO e as etapas de transformação em Python.

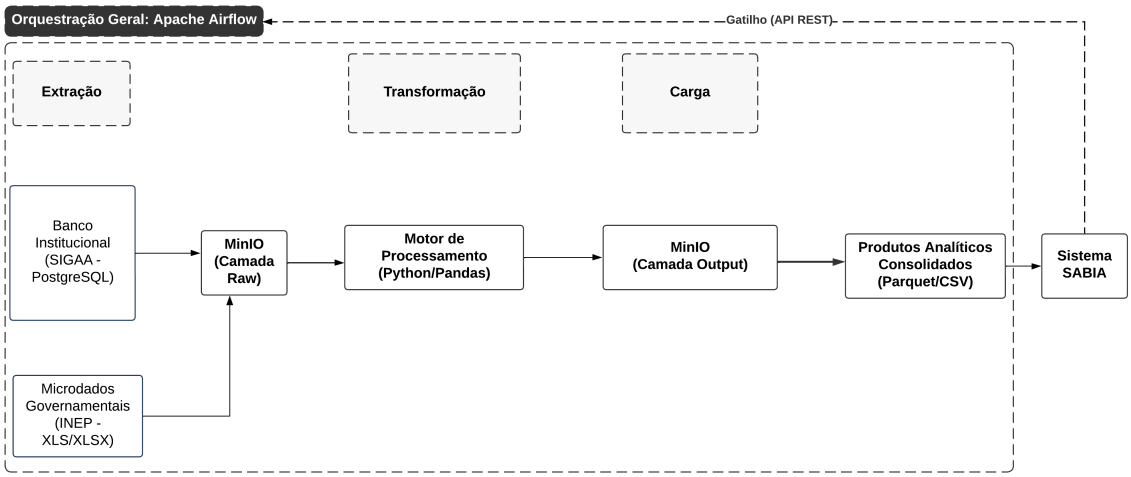
A execução do pipeline é iniciada externamente: o sistema SABIA envia uma requisição via API REST para o Airflow, que assume o controle do ETL a partir desse momento. Na etapa inicial, os dados extraídos das planilhas do governo e dos sistemas acadêmicos (SIGAA e SIGA) — ou supridos por dados simulados (*mocks*) para fins de teste, caso indisponíveis — são armazenados em sua forma original na camada *Raw* do MinIO. Em seguida, scripts em Python utilizando a biblioteca Pandas aplicam as regras de limpeza e executam o cruzamento das informações. Por fim, os arquivos consolidados são salvos na camada *Output* do MinIO, onde ficam disponíveis para consumo direto pelos painéis da plataforma.

4.2 Fontes de Dados

O pipeline trabalha com duas origens principais, unindo o contexto interno da universidade com a avaliação nacional apresentada na Seção 3.7 e seguintes:

SIGAA: os dados sobre perfil dos discentes, estrutura dos cursos, disciplinas, notas e matrículas têm origem no banco de dados relacional (PostgreSQL) da instituição. Por restrições de segurança de rede e princípios de governança, o Airflow não realiza conexão direta com os bancos de produção. A extração inicial e a aplicação de rotinas de anonimização dessas bases são realizadas exclusivamente pelo Professor

Figura 2 – Visão geral da arquitetura do sistema



Fonte: O autor (2026).

responsável, constituindo um estágio prévio que não integra o escopo prático deste trabalho. O pipeline orquestrado proposto inicia sua atuação consumindo diretamente os arquivos estruturados e já anonimizados, a partir do momento em que são depositados no *bucket* do MinIO.

SIGA: sistema legado da universidade, utilizado para resgatar os dados anteriores à adoção do SIGAA e garantir a preservação da série histórica da instituição. Seguindo os mesmos protocolos de segurança aplicados ao SIGAA, a extração e a anonimização são procedimentos prévios realizados pelo Professor responsável. Uma vez ingeridos pelo pipeline no ambiente do MinIO, os registros passam por um tratamento de padronização e são renomeados internamente para a taxonomia unificada "sigaa", harmonizando a origem para viabilizar as etapas de transformação conjunta.

INEP: os arquivos abertos (XLS/XLSX) são baixados diretamente do portal de Dados Abertos do órgão e, posteriormente, cruzados com as bases de desempenho interno da universidade. O recorte temporal adotado neste trabalho compreende o período de 2010 a 2023. Essa delimitação justifica-se por se tratar da série histórica integralmente consolidada e publicamente disponibilizada em formato de microdados até o momento da condução desta pesquisa. É característico do ciclo avaliativo do ENADE haver uma latência natural — frequentemente superior a um ano — entre a aplicação da prova e a liberação dos microdados tratados, o que explica a indisponibilidade dos dados referentes a 2024. Além disso, as diretrizes das edições mais recentes (2024 e 2025) adotaram formatos direcionados, a exemplo do Enade das Licenciaturas, focando a avaliação em áreas e cursos específicos em vez de abranger o perfil universitário global. Por comprometerem a comparabilidade longitudinal homogênea exigida pelos painéis do SABIA, esses anos foram deixados fora do escopo atual."

4.3 Ferramentas e Tecnologias Utilizadas

Para dar suporte ao fluxo de dados descrito na Fundamentação Teórica, a infraestrutura foi montada com as seguintes tecnologias:

Airflow: orquestrador do projeto. Os fluxos de trabalho são escritos em formato de código, compondo Grafos Acíclicos Direcionados (DAGs). O Airflow garante que uma tarefa só comece quando suas dependências forem atendidas e lida com retentativas automáticas caso ocorra falha na execução (SALLES, 2025).

Python e Pandas: toda a lógica de transformação foi desenvolvida em Python, utilizando o Pandas para as operações de limpeza, filtros e junções entre as bases.

MinIO: configurado com *buckets* separados para os arquivos originais (*Raw*) e os arquivos finais processados (*Output*), seguindo a adaptação da arquitetura Medallion descrita na Seção 3.4.

Formato Parquet: adotado como padrão de saída das tabelas geradas pelo pipeline, pelos motivos de desempenho e tipagem já discutidos na Seção 3.5.

4.4 O Processo de ETL

A implementação do ETL partiu diretamente das rotinas modulares já existentes no projeto, reestruturando-as em torno das etapas de extração, transformação e carga discutidas na Seção 3.2, porém adaptadas às particularidades de cada fonte de dados da UFRPE. Na extração, conforme justificado pelas restrições de segurança e conformidade com a Lei Geral de Proteção de Dados (LGPD), o pipeline não se conecta diretamente ao banco de produção do SIGAA. Em vez disso, os arquivos estruturados e previamente anonimizados pelo professor responsável são depositados no *bucket* do MinIO, de onde *scripts* dedicados, como o `etlExtractSigaa.py`, realizam sua leitura.

Para o ambiente de desenvolvimento e homologação, quando esses arquivos não estão disponíveis, a DAG `dag_populate_minio` gera dados simulados (*mocks*) das entidades acadêmicas. A utilização desses *mocks* não representa uma fragilidade do sistema, mas uma decisão metodológica voltada à segurança da informação: eles foram gerados sob supervisão, possuindo o esquema e a tipagem exatos das tabelas reais. Isso evita o trânsito contínuo de dados sensíveis de discentes fora do ambiente isolado da instituição, ao mesmo tempo em que garante que o pipeline seja validado de forma contínua, tornando-o agnóstico à origem física da tabela e plenamente apto a processar a carga real. Já os dados do INEP são obtidos de forma automatizada, por meio do download direto das planilhas a partir do portal de Dados Abertos do órgão. Em todos os casos, os arquivos são gravados sem qualquer alteração na camada *Raw* do MinIO, preservando uma cópia fiel da fonte recebida e permitindo o reprocessamento

em caso de falha nas etapas seguintes.

A transformação concentra a maior parte da lógica de negócio do pipeline e é executada com Pandas dentro de cada DAG específica, detalhadas na Seção 4.5. De forma geral, essa etapa cobre três frentes. A limpeza corrige tipos numéricos e textuais, remove caracteres inválidos e padroniza nomes de colunas herdados de fontes distintas (SIGAA, SIGA e INEP). O tratamento propriamente dito resolve divergências de nomenclatura entre *campi* e turnos; por exemplo, denominações como CAMPUS RECIFE, UNIDADE ACADÊMICA DE SERRA TALHADA e UEADT são normalizadas para formas padronizadas como SEDE, UAST e UAEADTEC, enquanto registros sem informação de turno são preenchidos como TURNO INDEFINIDO. Também corrige problemas recorrentes nas operações de *merge* entre SIGAA e SIGA, que geram colunas duplicadas com sufixos `_x` e `_y`, como `campus_x`/`campus_y` ou `nome_x`/`nome_y`, mantendo-se apenas a versão considerada consistente. Já o enriquecimento cria colunas derivadas, como o tempo de vínculo do aluno e os indicadores de sucesso por disciplina, além de realizar o cruzamento entre os dados internos da UFRPE e os microdados do INEP, agregando ao banco institucional atributos como área do curso, município e indicadores oficiais de desempenho, a exemplo de ENADE, IDD, CPC e IGC.

Por fim, a carga consolida as tabelas finais de alunos, cursos, disciplinas e indicadores e as exporta para a camada *Output* do MinIO em formato Parquet, disponibilizando-as para leitura direta pelo SABIA. Diferentemente do processo manual anterior, essa etapa passa a ser iniciada automaticamente ao término de cada DAG, eliminando a necessidade de qualquer ação humana entre o tratamento dos dados e sua disponibilização final.

4.5 Orquestração no Airflow

inicia

O fluxo completo é gerenciado por uma DAG principal chamada `dag_etl_sabia`. Em vez de processar dados diretamente, o papel dela é funcionar como um controlador: ela lê um arquivo de configuração (`CONFIG_SUB_DAGS`) e inicia as outras DAGs do projeto na ordem correta, usando o `TriggerDagRunOperator`.

Isso permite que rotinas independentes rodem ao mesmo tempo, mas garante que etapas que dependem de tabelas anteriores fiquem aguardando. Vale destacar que, sempre que uma DAG consolida pela primeira vez os dados brutos vindos do SIGAA e do SIGA, aplica-se como padrão a remoção de duplicidades resultantes da união das duas origens; esse tratamento não é repetido nas etapas seguintes, que já consomem bases previamente deduplicadas. Nas subseções a seguir, cada uma

dessas rotinas é explicada na ordem em que aparecem no fluxo, acompanhadas de suas representações visuais na interface do Airflow.

4.5.1 Ingestão Base: `dag_populate_minio`

A `dag_populate_minio` (Figura 3) é responsável pela carga inicial de arquivos externos no projeto. A principal função dela é baixar as planilhas de indicadores de qualidade da educação superior direto do site oficial do INEP e salvá-las no MinIO, organizando por ano.

O download é feito pela função `_download_e_salva_indicador_xlsx`. Ela faz a requisição do arquivo, salva provisoriamente na máquina, envia para o MinIO e depois apaga o arquivo local para não acumular arquivos extras no servidor. Esse processo se repete para os conceitos de Enade, IDD, CPC e IGC de vários anos. O ano de 2020 foi intencionalmente desconsiderado no código, pois o INEP não divulgou esses indicadores naquele ano.

Além da ingestão dos arquivos do INEP, a DAG possui uma rotina auxiliar chamada `populate_mock_data`, voltada exclusivamente para testes e desenvolvimento. Essa rotina gera dados simulados que espelham perfeitamente a estrutura e a tipagem das entidades acadêmicas do SIGAA — especificamente: cadastros de discentes, estrutura de cursos, matrizes curriculares (componentes), turmas e registros de matrículas com histórico de notas. A execução dessa validação ocorre sob demanda (*on-demand*) durante o ciclo de desenvolvimento, ou seja, sempre que o pipeline passa por uma manutenção, adição de uma nova regra de transformação ou auditoria de código. A necessidade dessa simulação justifica-se pelo isolamento do ambiente de produção: como a extração dos dados reais exige a intervenção manual do professor orientador para garantir a anonimização e a adequação à LGPD, o uso de *mocks* permite que a integridade do fluxo de ponta a ponta seja validada com alta frequência pelo desenvolvedor, sem onerar a equipe ou expor informações sensíveis. Após a geração, esses dados são marcados com origem SIGAA e exportados para os diretórios de saída do projeto, servindo como base para cenários de demonstração e testes lógicos. Embora essa funcionalidade esteja integralmente implementada, ela permanece desativada por padrão no código de produção, sendo acionada estritamente no ambiente de homologação. Durante os testes de execução, essa DAG levou cerca de 41 segundos para rodar.

4.5.2 Preparo de Fontes Externas: `dag_etl_indicadores_inep`

A `dag_etl_indicadores_inep` (Figura 4) cuida da padronização dos microdados do INEP. O maior problema dessa base é que os nomes das colunas e a estrutura

Figura 3 – DAG dag_populate_minio no Airflow

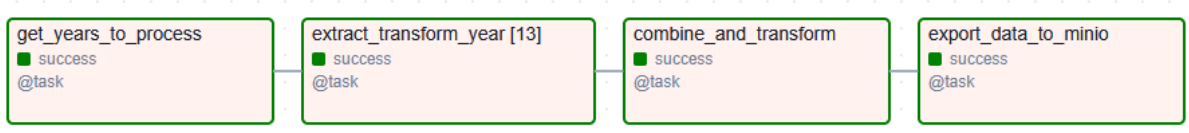
Fonte: O autor (2026).

dos arquivos mudam de um ano para o outro. Para resolver isso, o script usa um dicionário (`COLUMNS_MAP`) que mapeia os nomes antigos e novos para um padrão único, abrangendo os dados de 2010 a 2023.

Após renomear as colunas, o código filtra a base para manter apenas os dados da UFRPE (através do código da IES 587). Depois, os diferentes indicadores de um mesmo ano (IGC, CPC, Enade e IDD) são mesclados em uma única tabela.

Por fim, a rotina faz uma limpeza final: padroniza textos para caixa alta, ajusta os nomes das modalidades de ensino e corrige códigos de cursos que mudaram ao longo do tempo. O resultado é salvo no arquivo `output/df_indicadores_inep.parquet` dentro da pasta output. O tempo médio de execução dessa etapa foi de 1 minuto e 7 segundos.

Figura 4 – DAG dag_etl_indicadores_inep no Airflow



Fonte: O autor (2026).

4.5.3 Dimensão de Cursos: `dag_etl_curso`

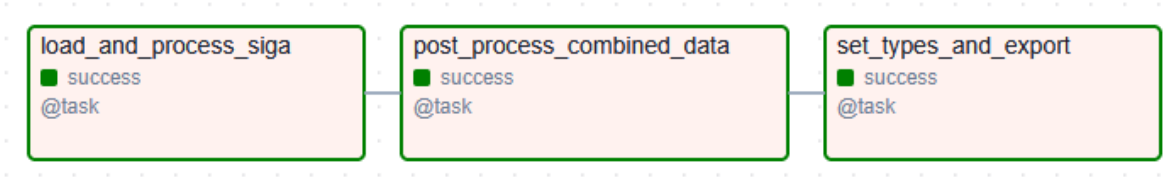
A `dag_etl_curso` (Figura 5) padroniza os dados referentes aos cursos da universidade. A primeira tarefa carrega e unifica os arquivos gerados pelas extrações do SIGAA e do sistema legado SIGA. Em seguida, aplica correções básicas, como preencher os valores nulos de códigos do INEP com zero e padronizar a nomenclatura dos *campi* da instituição. Embora todos os registros pertençam exclusivamente à UFRPE, a transição histórica entre sistemas legados e o preenchimento manual ao longo dos anos geraram divergências na grafia das unidades acadêmicas. Para resolver

isso, denominações dispersas como "CAMPUS RECIFE", "UNIDADE ACADÊMICA DE SERRA TALHADA"e "UEADT"são normalizadas para as nomenclaturas e siglas oficiais adotadas pelos painéis do SABIA, como "SEDE", "UAST"e "UAEADTEC", garantindo que os cálculos de métricas agrupem os discentes na unidade correta.

Para deixar o código do Airflow mais limpo, a maior parte das regras de transformação dos cursos foi movida para uma função externa dentro do arquivo `utils_etl_curso.py`.

A última tarefa garante que colunas numéricas de identificação estejam no tipo correto de dados (inteiros que aceitam valores nulos). Com tudo ajustado, a base é exportada para `output/df_curso.parquet`, centralizando as informações dos cursos para o restante do projeto. Essa DAG rodou em aproximadamente 19 segundos.

Figura 5 – DAG dag_etl_curso no Airflow



Fonte: O autor (2026).

4.5.4 Dimensão de Disciplinas: `dag_etl_componente`

Assim como a base de cursos, a `dag_etl_componente` (Figura 6) junta os componentes curriculares vindos dos dois sistemas acadêmicos (SIGAA e SIGA).

O código inicia carregando os dois arquivos em paralelo. No código atual, há uma padronização temporária em que as duas origens recebem o valor 'sigaa' na coluna de marcação. Em seguida, os dois conjuntos de dados são concatenados e os registros repetidos são eliminados. O fluxo foi feito de forma que, se um dos arquivos de origem faltar, a DAG continua a execução apenas com o arquivo disponível, evitando que o pipeline pare de executar por conta de um erro.

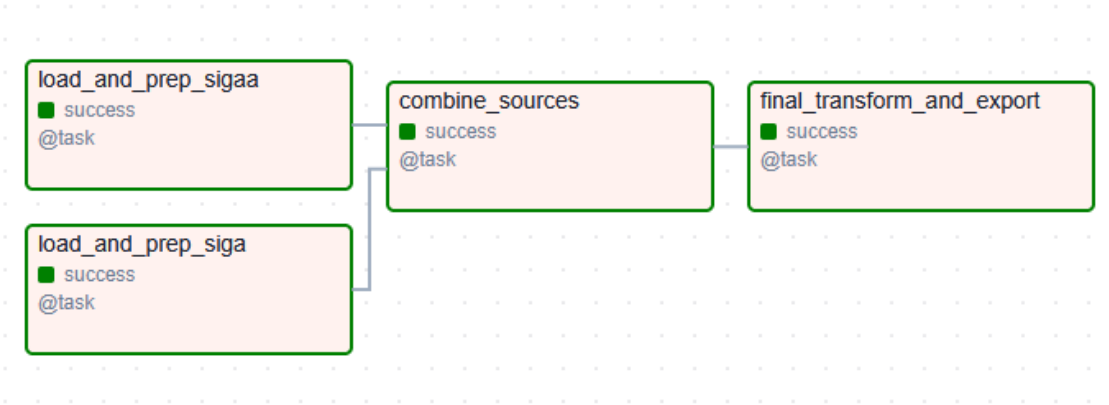
Na etapa final, a DAG aplica uma padronização textual abrangente, convertendo todos os campos do tipo texto para letras maiúsculas. Essa transformação justifica-se pela necessidade de garantir a integridade nas operações relacionais de cruzamento de dados (*joins*) e agrupamentos (*group by*) que ocorrem tanto no pipeline quanto nos *dashboards* do SABIA. Como as ferramentas analíticas e os motores de busca costumam ser sensíveis à capitalização (*case-sensitive*), variações simples de digitação nas fontes de origem (como "Recife", "RECIFE"ou "recife") criariam ramificações falsas e duplicidades nas visualizações. A conversão em maiúsculas atua como uma camada final de governança, uniformizando a base antes de sua persistência no formato Parquet.

4.5.5 Convergência Discente: `dag_discente`

A `dag_discente` (Figura 7) é responsável pelo tratamento dos dados cadastrais dos alunos. Para isso, ela carrega de forma centralizada três bases principais: os registros de discentes advindos do SIGAA, os registros legados do SIGA e a base de cursos já processada, necessária para validar e complementar as informações acadêmicas de cada aluno.

Após a ingestão, a DAG executa uma etapa de transformação separada para cada origem de dados. No caso do SIGAA, são aplicadas rotinas como preenchimento

Figura 6 – DAG dag_etl_componente no Airflow

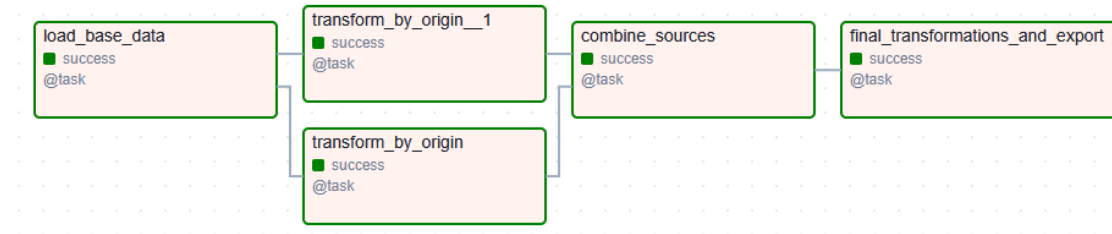


Fonte: O autor (2026).

de turnos ausentes com o valor "TURNO INDEFINIDO", padronização textual em caixa alta, ajuste do campo `forma_ingresso` e identificação das entradas regulares. Já para os dados legados do SIGA, o tratamento inclui a padronização de atributos cadastrais, como o campo de gênero, além da aplicação de regras específicas para classificar o tipo de ingresso. Em ambos os casos, os registros são cruzados com a base de cursos previamente tratada, o que permite complementar e validar os vínculos acadêmicos dos discentes. Em seguida, os dados processados são consolidados em uma única base, com remoção de duplicidades e descarte de colunas totalmente nulas, o que contribui para padronizar a estrutura antes da etapa final.

Durante a consolidação, a rotina também trata um problema recorrente em operações de *merge* com Pandas: a duplicação de colunas com os sufixos `_x` e `_y`. Nessa situação, a implementação adota por convenção a versão `_y` como referência final, remove a correspondente `_x` e renomeia a coluna preservada para seu nome original. Em seguida, executa uma etapa adicional de pós-processamento para refinar o resultado final. Ao término, a base consolidada de discentes é exportada para o MinIO em formato Parquet. Esse processamento levou cerca de 18 segundos.

Figura 7 – DAG dag_discente no Airflow



Fonte: O autor (2026).

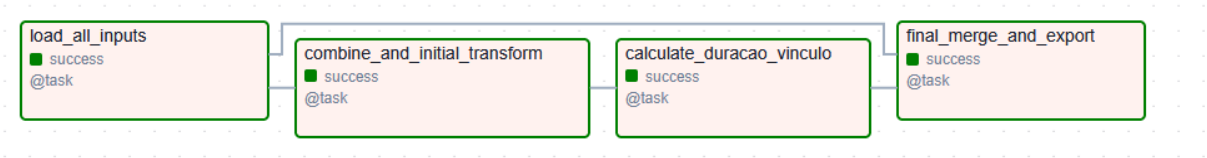
4.5.6 Granularidade de Período: `dag_discente_curso_turno_periodo`

A `dag_discente_curso_turno_periodo` (Figura 8) foca no histórico do aluno ao longo do tempo. Em vez de olhar só para o cadastro geral, ela acompanha semestre a semestre o vínculo do estudante.

Depois de carregar as bases de período, a DAG unifica os registros das duas origens e realiza uma padronização inicial. Nessa etapa, também converte campos binários, como ingresso, situação final e vínculo, para valores booleanos, facilitando a leitura. Em seguida, ela calcula a `duracao_vinculo` do aluno, que é quanto tempo períodos ele passou na universidade. Para isso, o script conta apenas os períodos regulares e preenche os espaços em branco para frente, garantindo que mesmo os semestres excepcionais consigam trazer os dados o histórico de duração do aluno.

No final, a DAG cruza esse histórico de semestres com os dados cadastrais do aluno e cria a coluna `retido_duracao_vinculo`, que indica se o estudante já ultrapassou o tempo previsto de duração do curso. Antes da exportação, o processo ainda remove colunas redundantes resultantes da consolidação, deixando a base final mais limpa. O arquivo gerado fica sendo a base para estudos de evasão, sendo processado em cerca de 20 segundos.

Figura 8 – DAG dag_discente_curso_turno_periodo no Airflow



Fonte: O autor (2026).

4.5.7 Perfil de Turmas: `dag_etl_curso_turno_periodo`

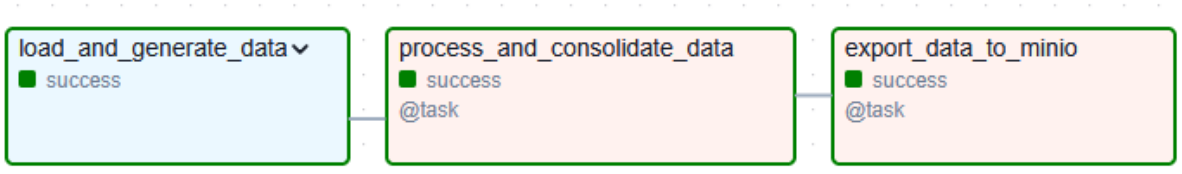
A `dag_etl_curso_turno_periodo` (Figura 9) é responsável por transformar o histórico individual dos estudantes em uma base agregada por curso, turno e período letivo. Em vez de trabalhar no nível do aluno, essa etapa reorganiza os dados em nível institucional, produzindo uma visão consolidada da oferta e entrada em cada curso ao longo do tempo.

Para isso, a DAG usa como fonte a base `df_discente_curso_turno_periodo`, tratada na etapa anterior, considerada no código como a referência central para reconstruir a estrutura temporal dos cursos. A partir dela, o processo identifica combinações únicas de período letivo, curso e turno, formando a malha básica sobre a qual os demais atributos são adicionados.

Um ponto importante dessa DAG é que ela foi adaptada para funcionar mesmo sem o arquivo oficial de vagas da universidade. Como estava temporariamente indisponível, o pipeline passou a gerar em memória uma base simulada de vagas, tomando como molde os próprios cursos e períodos existentes na base de discentes. Nessa simulação, os valores de vagas são sorteados dentro de um intervalo plausível de 30 a 61 vagas, enquanto os indicadores de ingressantes e ingressantes regulares também são preenchidos artificialmente para manter a consistência, com números em torno de 38 a 44 ingressantes.

Depois disso, a DAG cruza a base estrutural dos cursos com esses valores simulados e produz um conjunto consolidado com informações como vagas, ingressantes, ingressantes regulares e população ideal. Mesmo sendo uma solução temporária, essa abordagem permite validar o fluxo completo da pipeline e sustentar testes das etapas analíticas seguintes até que os dados institucionais reais sejam disponibilizados no MinIO. O resultado final é exportado como `output/df_curso_turno_periodo`, em uma execução de aproximadamente 19 segundos.

Figura 9 – DAG dag_etl_curso_turno_periodo no Airflow



Fonte: O autor (2026).

4.5.8 Desempenho por Componente: `dag_discente_disciplina_periodo`

A `dag_etl_curso_turno_periodo` (Figura 9) é responsável por transformar o histórico individual dos estudantes em uma base agregada por curso, turno e período letivo. Em vez de trabalhar no nível do aluno, essa etapa reorganiza os dados em nível institucional, produzindo uma visão consolidada da oferta e entrada em cada curso ao longo do tempo.

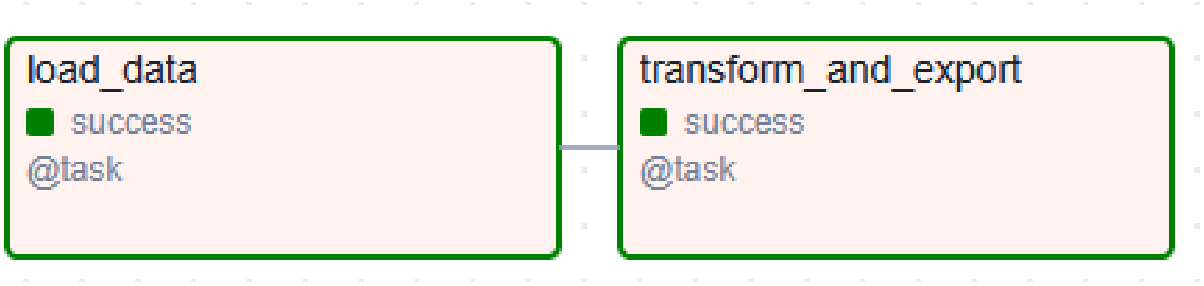
Para isso, a DAG utiliza como fonte principal a base `df_discente_curso_turno_periodo`, tratada na etapa anterior, adotada como referência central para reconstruir a linha do tempo dos cursos. Nessa etapa, é construída uma estrutura base formada pelas combinações distintas de período letivo, curso e turno observadas nos dados acadêmicos, sobre a qual os demais atributos são posteriormente agregados.

Um ponto metodológico importante dessa DAG diz respeito aos quantitativos de vagas e ingressos. Na versão experimental desta rotina, os campos referentes a vagas e ingressantes foram preenchidos com valores simulados (gerados logicamente pelo

código) apenas para viabilizar testes de integração e estruturação da base, mitigando a indisponibilidade temporária de algumas fontes no ambiente de desenvolvimento. Essa parametrização artificial garante a consistência das etapas analíticas subsequentes e valida o funcionamento de ponta a ponta do pipeline, devendo tais variáveis ser posteriormente substituídas pelos quantitativos oficiais de oferta da UFRPE para que reflitam a capacidade real de ingresso de cada curso e turno.

Após estabelecer a estrutura base, a DAG realiza o cruzamento com esses valores e calcula indicadores derivados, destacando-se a população ideal. A coluna `populacao_ideal` representa a quantidade estimada de estudantes que um curso deveria manter vinculados em um dado período, considerando as vagas ofertadas nas turmas de ingresso ainda compreendidas dentro do tempo ideal de conclusão. Em outras palavras, trata-se de uma medida de referência teórica derivada da oferta institucional da UFRPE e da duração ideal do curso. Ao final do processamento, a base estruturada é exportada para o MinIO em formato Parquet, armazenada sob o caminho `output/df_curso_turno_periodo`. A execução dessa rotina de agregação leva aproximadamente 19 segundos.

Figura 10 – DAG dag_discente_disciplina_periodo no Airflow



Fonte: O autor (2026).

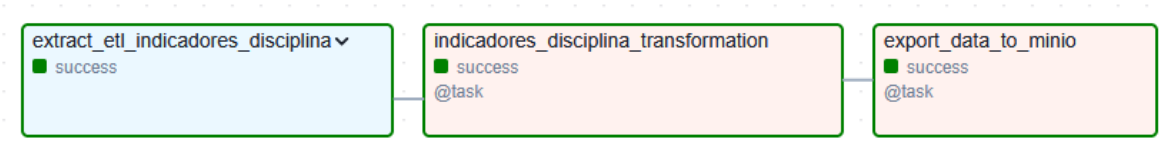
4.5.9 Agregação de Disciplina: `dag_etl_indicadores_disciplina`

A `dag_etl_indicadores_disciplina` (Figura 11) resume os dados detalhados das matrículas em indicadores estatísticos por disciplina. Partindo da base `df_discente_disciplina_periodo`, a DAG cria marcações lógicas para categorizar a situação final do aluno (aprovado, reprovado ou cancelado). Um ponto metodológico central dessa transformação é a segregação das notas: o cálculo de desempenho médio é realizado isolando-se exclusivamente as notas dos discentes aprovados. Essa separação justifica-se para evitar severas distorções estatísticas nos *dashboards* finais. Alunos reprovados (especialmente por falta) ou que cancelaram a matrícula frequentemente apresentam notas zeradas ou atipicamente baixas, que refletem o abandono da disciplina, e não o nível real de assimilação do conteúdo. Ao separar os desempenhos de situações diferentes, o pipeline garante que o SABIA exiba uma métrica confiável sobre

o rendimento acadêmico da turma, impedindo que o viés da evasão ou reprovação mascare a média real de aprendizado.

Em seguida, os dados são agrupados por período letivo, componente curricular, curso, turno, tipo de período e unidade responsável. Sobre esse agrupamento, são calculados indicadores como quantidade de matrículas, média e desvio-padrão das notas (geral e só dos aprovados), média de faltas, e os totais de aprovados, reprovados e cancelados.

Figura 11 – DAG dag_etl_indicadores_disciplina no Airflow



Fonte: O autor (2026).

4.5.10 Métricas Institucionais: `dag_etl_indicadores_curso`

A `dag_etl_indicadores_curso` (Figura 12) consolida uma das bases mais completas do projeto. Ela junta os históricos dos alunos, as características dos cursos, os dados de vagas e os indicadores do INEP.

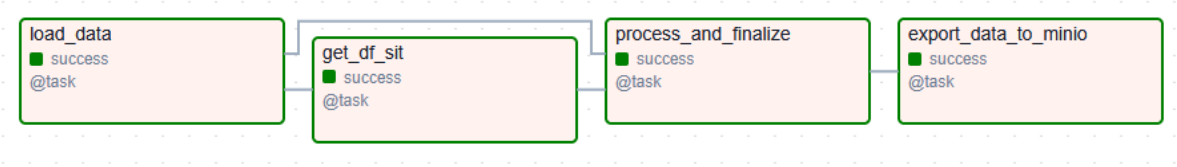
Primeiro, a dag parte da base de discentes por curso, turno e período e conta quantos estudantes estão cursando, formados, cancelados, evadidos ou suspensos em cada oferta. Quando a informação de assistência estudantil está disponível, essa mesma contagem é feita separadamente para os alunos assistidos.

Em seguida, a base recebe a separação em ano e semestre e é cruzada com os dados cadastrais dos cursos e com os indicadores do INEP (ENADE, IDD, CPC e IGC). Como o ciclo avaliativo do Sistema Nacional de Avaliação da Educação Superior (SINAES) ocorre, via de regra, a cada três anos para uma mesma área de conhecimento, os cursos não recebem novas notas anualmente. Para evitar lacunas na série temporal que comprometeriam as métricas nos painéis do SABIA, a DAG emprega uma técnica de preenchimento progressivo (*forward fill*). Essa rotina propaga o indicador mais recente divulgado para preencher os anos subsequentes sem avaliação. Essa estratégia justifica-se metodologicamente pois o conceito obtido por um curso permanece oficialmente válido perante o Ministério da Educação (MEC) até que o próximo ciclo avaliativo seja concluído, garantindo assim a integridade e a continuidade das correlações analíticas ao longo do tempo.

Por fim, o fluxo incorpora os dados de vagas, ingressantes, população ideal e duração de curso, renomeia as colunas para um padrão único e calcula indicadores

como `vagas_ociosas` e `taxa_preenchimento_vagas`. Por envolver muitos cruzamentos, essa etapa levou cerca de 1 minuto e 5 segundos.

Figura 12 – DAG dag_etl_indicadores_curso no Airflow



Fonte: O autor (2026).

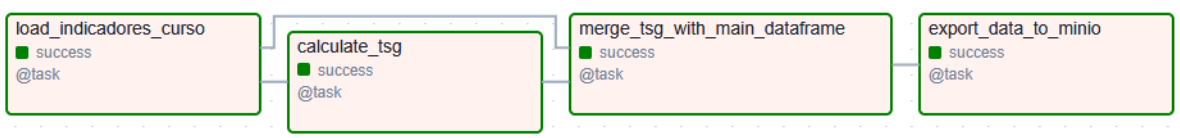
4.5.11 Taxa de Sucesso: `dag_tsg`

A `dag_tsg` (Figura 13) calcula a Taxa de Sucesso na Graduação (TSG) a partir da base `df_indicadores_curso`, relacionando a quantidade de ingressantes regulares com o número de alunos diplomados dentro do prazo esperado.

Quando a coluna `conclusao_ideal_turma` está disponível, ela é usada como referência de prazo. Caso contrário, a DAG estima essa data somando 9 semestres ao período de ingresso, aproximação compatível com a duração de uma graduação de quatro anos e meio.

Depois disso, a rotina agrupa os dados por curso e ano de conclusão, soma os ingressantes e os diplomados de cada grupo e calcula a TSG pela razão entre esses dois valores, tratando os casos de divisão por zero. O resultado é anexado de volta à base de indicadores de curso. Essa etapa levou cerca de 19 segundos.

Figura 13 – DAG dag_tsg no Airflow



Fonte: O autor (2026).

4.5.12 Matriz de Reoferta: `dag_etl_reoferta`

A `dag_etl_reoferta` (Figura 14) tenta sugerir quantas vagas um curso deveria abrir, comparando a capacidade ideal de alunos com a ocupação real de quem ainda está matriculado.

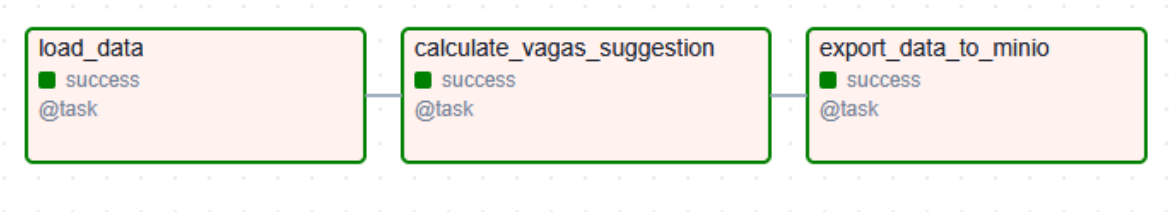
Para isso, o fluxo cruza a base de curso por período com a base de discentes por curso, turno e período, filtrando apenas os alunos vinculados em períodos regulares. A

partir disso, calcula-se a população real e o saldo entre a capacidade ideal e a ocupação observada. A DAG também cria uma segunda medida, `saldo_r2`, que considera os alunos retidos dentro de uma janela de tolerância de dois semestres.

Esse cálculo depende da variável `conclusao_ideal_turma`. Quando essa coluna não está disponível na base de entrada, a implementação verifica a existência de `periodo_letivo_conclusao_ideal`, tratando-a como coluna alternativa por representar a mesma referência temporal de conclusão ideal da turma. Caso nenhuma das duas esteja presente, o pipeline estima esse valor a partir do período letivo de ingresso, projetando a conclusão ideal em semestres futuros.

Por fim, os saldos são convertidos em sugestões de vagas por uma regra de arredondamento em blocos de cinco, resultando em faixas como “5 a 10 vagas”. A execução demorou 20 segundos.

Figura 14 – DAG dag_etl_reoferta no Airflow



Fonte: O autor (2026).

4.5.13 Preparação para Modelagem: `dag_etl_models`

A `dag_etl_models` (Figura 15) monta a tabela que pode ser usada futuramente para treinar modelos preditivos. Ela reorganiza os dados em uma estrutura por aluno e período letivo, juntando o histórico de vínculo acadêmico com o desempenho em disciplinas.

A DAG parte de duas bases: a trajetória do aluno por curso, turno e período, que traz atributos como curso, turno, forma de ingresso, área de conhecimento e situação do vínculo; e o histórico de desempenho por disciplina, com médias, carga horária cursada, aprovações, reprovações e cancelamentos.

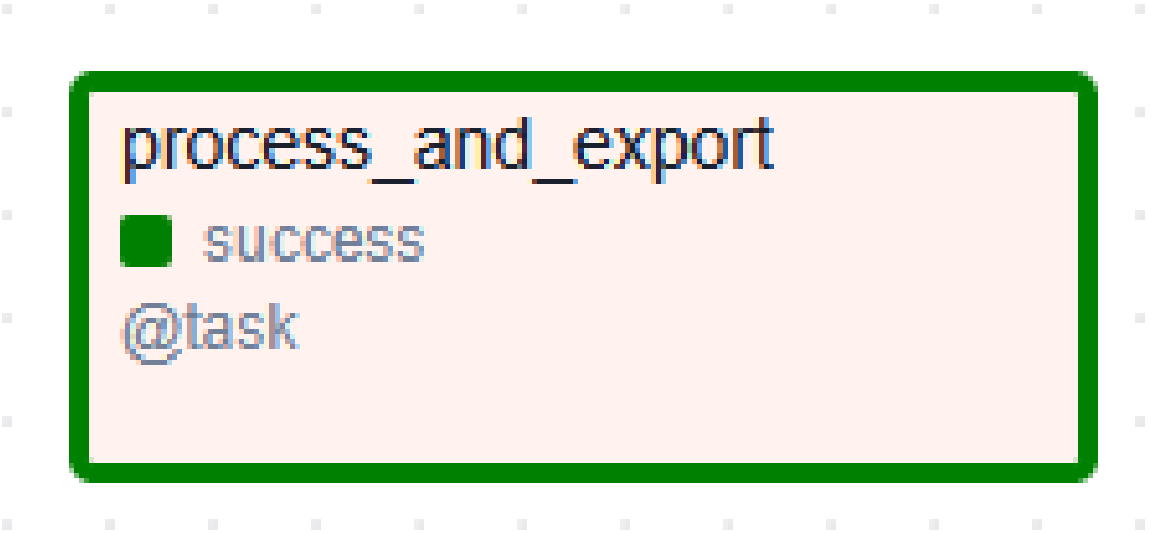
No bloco de vínculo, o código identifica o último período letivo registrado para cada discente e utiliza essa informação para inferir sua condição final no curso. A partir da situação observada nesse último período, são criadas variáveis indicadoras que distinguem alunos formados, evadidos e ainda vinculados. Em seguida, para os casos de evasão, a rotina aplica uma classificação baseada no tempo total de permanência do estudante. Quando a saída ocorre em até dois períodos letivos, a evasão é tratada como mais precoce; quando ocorre até quatro períodos, como evasão intermediária;

quando o aluno abandona o curso após atingir ou ultrapassar a duração prevista de integralização, a evasão é classificada como tardia. Os casos restantes são mantidos como evasão em faixa intermediária de percurso.

No bloco de desempenho, a rotina consolida as disciplinas cursadas em cada semestre e calcula médias, carga horária total e quantidade de resultados por situação, acumulando esses valores ao longo do tempo para acompanhar a evolução do aluno semestre a semestre.

Por fim, as duas bases são unidas e a DAG cria variáveis de retenção, como excesso de reprovações e de trancamentos acumulados: se o aluno passar de cinco, a linha recebe um alerta. Essa etapa levou 18 segundos.

Figura 15 – DAG dag_etl_models no Airflow



Fonte: O autor (2026).

4.5.14 Estudo de Correlação: `dag_etl_correlacao`

A `dag_etl_correlacao` (Figura 16) calcula se a nota de uma disciplina tende a acompanhar a nota de outra, indicando matérias que têm comportamento semelhante no histórico dos alunos.

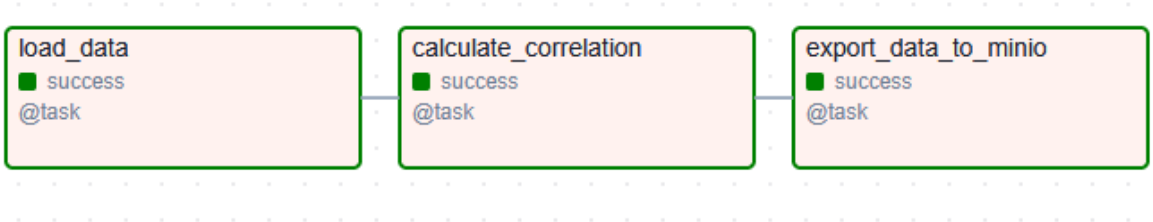
Para focar em dados mais relevantes, a DAG limita o processamento aos últimos dez anos letivos. Quando a base de entrada não traz o ano de forma explícita, o código reconstrói essa informação a partir do campo `periodo_letivo`.

Em seguida, os dados são reorganizados em formato matricial, com os alunos nas linhas, as disciplinas nas colunas e as médias finais como valores. Sobre essa matriz é aplicado o cálculo de correlação de postos de Spearman (DANCEY; REIDY, 2018),

uma medida estatística não paramétrica que capta associações entre desempenhos sem exigir que as notas sigam uma distribuição normal.

O código exige que a comparação seja feita apenas se houver um pareamento de pelo menos 40 alunos em comum entre as disciplinas analisadas. Essa restrição metodológica justifica-se pela necessidade de garantir a robustez e a significância estatística do coeficiente gerado. Em amostras pequenas, o cálculo de correlação torna-se altamente sensível a valores atípicos (*outliers*), podendo indicar correlações espúrias motivadas puramente pelo acaso. Ao estabelecer um piso de 40 observações — patamar que oferece uma margem de segurança superior ao limiar tradicional de 30 amostras amplamente referenciado na literatura estatística —, o pipeline evita falsos positivos gerados por turmas com baixa adesão. Isso assegura que o SABIA exiba apenas padrões de similaridade sólidos e historicamente consolidados no percurso acadêmico. O processamento durou 19 segundos.

Figura 16 – DAG dag_etl_correlacao no Airflow



Fonte: O autor (2026).

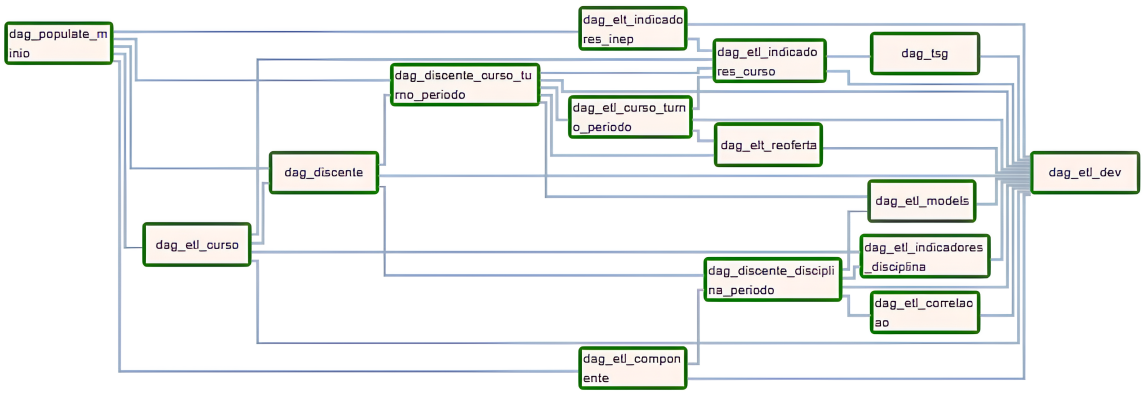
4.5.15 Orquestração Final: `dag_etl_sabia`

Por fim, a `dag_etl_sabia` (Figura 17) é a responsável por juntar todas as outras peças. Conforme explicado no início, ela não modifica tabelas, mas cria a ordem de execução com o `TriggerDagRunOperator`, forçando cada etapa do pipeline a esperar sua vez quando necessário.

O fluxo de dependências faz com que os dados brutos sejam carregados primeiro. Depois vêm as tabelas mais gerais de cursos e disciplinas, seguidas pelos históricos dos alunos. As etapas analíticas de correlação e sugestão de vagas ficam por último, porque precisam que todas as tabelas anteriores já estejam prontas.

Somando o tempo de execução de todas as DAGs individuais citadas nas seções anteriores, chega-se a aproximadamente 7 minutos e 04 segundos. Contudo, como a `dag_etl_sabia` permite que DAGs que não dependem uma da outra sejam executadas em paralelo pelo Airflow, o tempo real percebido do pipeline costuma ser menor do que a simples soma dos tempos individuais.

Figura 17 – DAG dag_etl_sabia no Airflow



Fonte: O autor (2026).

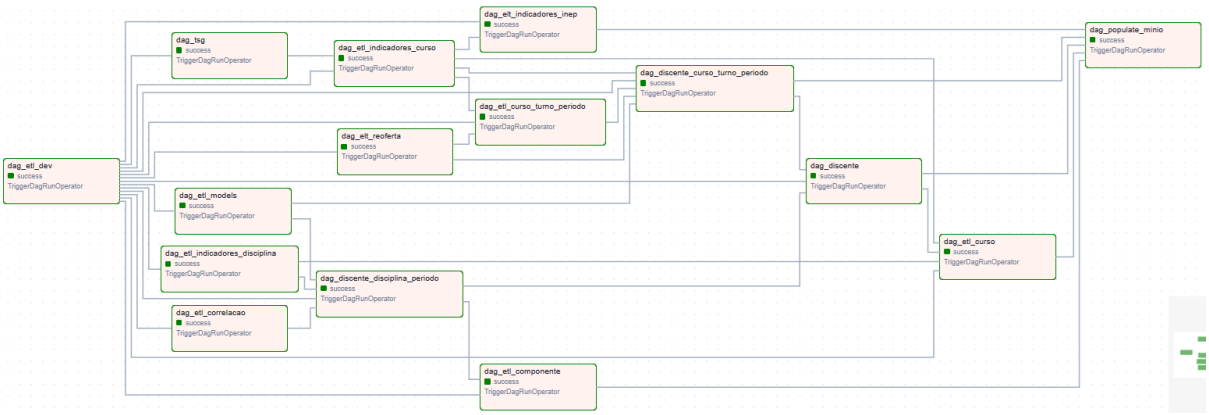
5 Resultados e Discussão

A arquitetura de dados proposta neste trabalho foi validada por meio da execução prática do pipeline completo, contemplando as 14 DAGs descritas no Capítulo 4 e a DAG controladora `dag_etl_sabia`. Este capítulo apresenta os resultados em cinco frentes: execução técnica do pipeline, tratamento de problemas de qualidade, produtos analíticos gerados, disponibilização das bases no MinIO e impactos institucionais e práticos da solução.

5.1 Execução Orquestrada do Pipeline

A Figura 18 apresenta a interface do Apache Airflow com o grafo de execução (*Graph View*) da rotina principal, evidenciando o padrão *Controller DAG*: a `dag_etl_sabia` não processa dados diretamente, apenas aciona as demais DAGs na ordem correta por meio do `TriggerDagRunOperator`, respeitando as dependências entre elas — a extração inicial (`dag_populate_minio`) precisa terminar antes das rotinas de tratamento primário (`dag_discente`, `dag_etl_curso`), que por sua vez precisam terminar antes das etapas analíticas finais (correlação, reoferta, modelagem).

Figura 18 – Grafo de dependências e execução das tarefas orquestradas pelo Apache Airflow



Fonte: O autor (2026).

A Tabela 1 detalha a execução de todas as DAGs do pipeline, todas concluídas com sucesso.

Tabela 1 – Execução de todas as DAGs do pipeline

DAG	Entrada	Saída	Tempo
dag_populate_minio	INEP	INEP + SIGAA (mock)	00:41
dag_elt_indicadores_inep	Planilhas INEP (2010-2023)	df_indicadores_inep	01:07
dag_etl_curso	SIGAA/SIGA	df_curso	00:19
dag_etl_componente	SIGAA/SIGA	df_componente	00:20
dag_discente	SIGAA/SIGA + df_curso	df_discente	00:18
dag_discente_curso_turno_periodo	df_discente + df_populate_minio	df_discente_curso_turno_periodo	00:20
dag_etl_curso_turno_periodo	df_discente_curso_turno_periodo	df_curso_turno_periodo	00:19
dag_discente_disciplina_periodo	df_discente + df_componente	df_discente_disciplina_periodo	00:18
dag_etl_indicadores_disciplina	df_curso + df_discente_disciplina_periodo	df_indicadores_disciplina	01:00
dag_etl_indicadores_curso	df_indicadores_inep + df_curso + df_curso_turno_periodo	df_indicadores_curso	01:05
dag_tsg	df_indicadores_curso	df_indicadores_curso	00:19
dag_elt_reoferta	df_curso_turno_periodo + df_discente_curso_turno_periodo	df_reoferta	00:20
dag_etl_models	Vínculo + desempenho	df_vinculo_periodo	00:18
dag_etl_correlacao	df_discente_disciplina_periodo	df_correlacao_componentes	00:20

Fonte: O autor (2026).

Somando o tempo individual de todas as DAGs, o pipeline completo processa as bases em aproximadamente 7 minutos e 04 segundos. Como o Airflow executa em paralelo as DAGs que não dependem umas das outras (por exemplo, `dag_etl_curso` e `dag_etl_componente` podem rodar ao mesmo tempo), o tempo real percebido tende a ser menor que essa soma.

5.2 Tratamento de Problemas de Qualidade e Integração

A execução do pipeline evidenciou uma série de problemas de qualidade típicos de ambientes com múltiplas fontes, cada um resolvido em uma etapa específica do processo, conforme resume a Tabela 2.

Tabela 2 – Problemas de qualidade tratados e a DAG responsável pela solução

Problema identificado	**Onde ocorre**	**Solução aplicada**
Nomes de colunas do ENADE mudam a cada ano	`dag_elt_indicadores_inep`	Dicionário `COLUMNS_MAP` unificando 2010-2023 em um padrão único
Indicador de 2020 ausente na fonte oficial	`dag_elt_indicadores_inep`	Ano ausente intencionalmente no código, sem gerar erro no pipeline
Colunas duplicadas após *merge* (sufixos _x/_y)	`dag_discente`	Remoção das versões redundantes, mantendo a coluna mais confiável
Nomes divergentes de *campi* e cursos	`dag_etl_curso`	Normalização textual e padronização de nomenclatura
Arquivo oficial de vagas indisponível	`dag_etl_curso_turno_periodo`	Geração temporária de valores simulados (30 a 61 vagas por curso/período) até a disponibilização do arquivo real
Ausência da data de conclusão ideal da turma	`dag_tsg`, `dag_elt_reoferta`	Estimativa por soma de 9 semestres ao período de ingresso quando o dado não está disponível
Matrículas sem correspondência válida	`dag_discente_disciplina_periodo`	Filtragem dos registros sem vínculo confiável a um componente

Fonte: O autor (2026).

Vale destacar que nem todas essas soluções são definitivas. O caso das vagas simuladas, em particular, é um contorno temporário: os indicadores derivados dela (como `vagas_ociosas` e as sugestões da `dag_elt_reoferta`) devem ser tratados como validação do fluxo, não como números institucionais reais, enquanto o arquivo oficial da universidade não for integrado ao MinIO.

5.3 Produtos Analíticos Automatizados

O pipeline não introduziu novos cálculos: as rotinas que geram `taxa_sucesso`, `taxa_sucesso_ampla`, a Taxa de Sucesso na Graduação (`dag_tsg`), as sugestões de reoferta (`dag_elt_reoferta`) e a matriz de correlação entre disciplinas (`dag_etl_correlacao`) já existiam como scripts corretos antes deste trabalho. O que não existia era a garantia

de que todos seriam executados, na ordem certa, sempre que uma atualização fosse necessária.

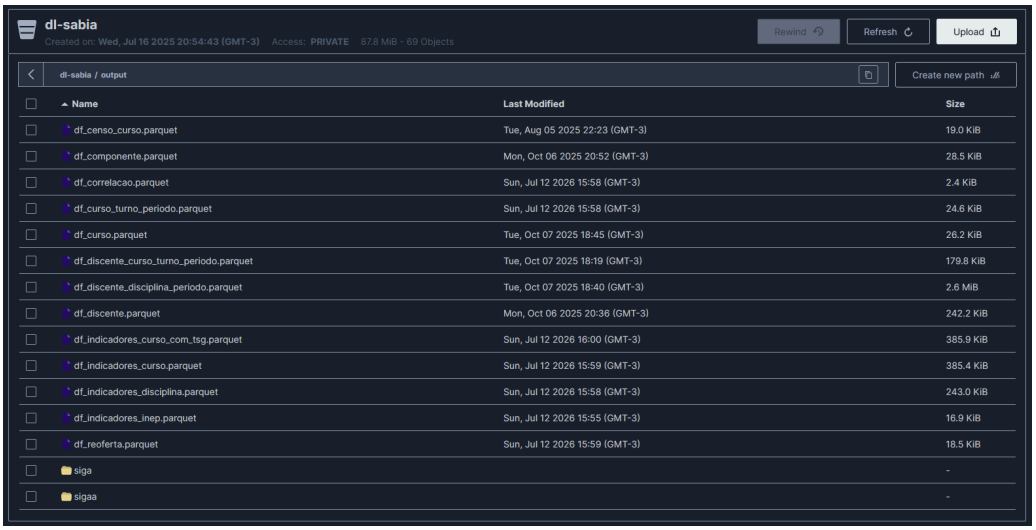
No processo manual anterior, cada um desses scripts precisava ser iniciado individualmente por uma pessoa, respeitando as dependências entre eles — a `dag_tsg`, por exemplo, só produz um resultado correto se `dag_etl_indicadores_curso` já tiver rodado antes. Se essa ordem não fosse seguida à risca, ou se alguma etapa fosse esquecida, uma ou mais bases ficavam desatualizadas sem que isso ficasse evidente para quem consultava o SABIA depois.

Com o Airflow, a `dag_etl_sabia` passou a acionar essas 14 rotinas na ordem correta automaticamente, bastando uma execução para atualizar toda a cadeia. Além da velocidade, o ganho está na visibilidade: se alguma etapa falha, a interface do Airflow aponta exatamente qual DAG parou e o log da execução registra o motivo, algo que o processo manual não oferecia — antes, uma base desatualizada só era percebida quando alguém notava um número estranho no próprio painel do SABIA.

5.4 Disponibilização das Bases no MinIO

A Figura 19 mostra o resultado final do pipeline: as bases listadas na Tabela 1 já materializadas na camada *Output* do MinIO, prontas para consumo pelos painéis do SABIA. Com o pipeline em produção, a atualização integra-se diretamente à interface do usuário, em que basta um administrador do sistema acionar o botão de atualização de dados no próprio SABIA na página de Configurações. Essa ação inicia a requisição para o Airflow e desencadeia toda a cadeia de DAGs.

Figura 19 – Bases analíticas disponibilizadas automaticamente no MinIO



Fonte: O autor (2026).

5.5 Resultados Institucionais e Práticos

A centralização do fluxo em uma arquitetura automatizada reduz o retrabalho nas etapas de coleta e preparação, e favorece a atualização recorrente das bases usadas pela gestão da universidade. Ao automatizar a ingestão, a padronização e a carga dos dados, a arquitetura permite que as bases sejam atualizadas sempre que necessário, em vez de depender da rotina manual usada anteriormente.

A organização modular do pipeline também simplifica sua evolução: uma vez estabelecido o padrão das rotinas de ETL e sua integração com a `dag_etl_sabia`, novas tabelas podem ser incorporadas por meio da criação de uma DAG específica, testada de forma independente antes de entrar no fluxo principal.

5.6 Discussão dos Resultados

Retomando as questões de pesquisa (RQs) estabelecidas na Introdução deste trabalho, a implementação do pipeline orquestrado pelo Apache Airflow demonstrou resultados que respondem diretamente aos problemas levantados.

RQ1 — Confiabilidade e tempo de atualização. A Tabela 1 demonstra que as 14 DAGs foram concluídas com sucesso, processando o pipeline completo de ponta a ponta em aproximadamente 7 minutos e 04 segundos. Mais do que a otimização do tempo, o maior ganho desse modelo reside na elevação da confiabilidade operacional. No modelo anterior (manual), a consistência das bases do SABIA dependia do acionamento sequencial correto por um operador humano, abrindo margem para esquecimentos e erros de dependência temporal. Com a nova arquitetura, a orquestração foi unificada: um único acionamento da `dag_etl_sabia` garante a execução íntegra de toda a cadeia, respeitando estritamente a relação de precedência temporal entre os dados brutos e as tabelas agregadas.

RQ2 — Adaptações técnicas nas rotinas modulares. A transição de arquivos isolados para Grafos Acíclicos Direcionados exigiu muito mais do que o simples encapsulamento de *scripts*. O principal trabalho foi reestruturar a lógica original para viabilizar o tratamento autônomo e interdependente das bases. Isso envolveu a criação de regras de negócio fundamentais para a integridade do *Data Lakehouse*, tais como: o tratamento de junções para resolver colunas duplicadas (eliminando sufixos `_x` e mantendo a referência `_y`); a padronização textual (*case-sensitive*) e de nomenclatura de *campi* para evitar falsas ramificações nos *dashboards*; a aplicação de heurísticas para suprir lacunas (como a estimativa de 9 semestres para conclusão ideal); e a validação de integridade referencial, descartando "disciplinas fantasmas"no cruzamento de dados. A arquitetura também foi ajustada para modelar quantitativos de vagas institucionais

fundamentados diretamente no arquivo oficial da UFRPE, garantindo que as tabelas reflitam a capacidade real de oferta.

RQ3 — Rastreabilidade e auditoria (Governança de Dados). A Figura 18 ilustra como cada tarefa passou a ser registrada individualmente, com *status*, *logs* detalhados e duração visíveis na interface do Airflow. Diferentemente da execução manual, que não deixava rastros auditáveis, essa centralização responde de forma direta à lacuna de governança apontada por Cernea e Kierylo (2024) para as instituições de ensino superior. Com a ferramenta, é possível saber exatamente qual DAG gerou cada base, em qual momento e sob quais condições. Ademais, a modularidade do Airflow provê uma recuperação ágil de falhas: se uma etapa de transformação quebra, o operador é notificado e pode reexecutar apenas a DAG afetada a partir do ponto de falha, sem a necessidade de reprocessar o pipeline inteiro.

Para alinhar os resultados ao escopo da pesquisa, vale delimitar o alcance da infraestrutura desenvolvida. Em conformidade irrestrita com a Lei Geral de Proteção de Dados (LGPD), a etapa de anonimização dos microdados do SIGAA e do SIGA continua sendo executada previamente pelo professor responsável em ambiente isolado. O pipeline atua de maneira agnóstica a partir do momento em que esses arquivos seguros são disponibilizados no MinIO, o que reforça a segurança da arquitetura. Quanto ao INEP, a ausência dos indicadores consolidados de 2024 e o escopo restrito de 2025 (focado em licenciaturas) representam uma limitação estrita da latência do ciclo avaliativo do SINAES, tratando-se de um gargalo inerente à fonte governamental, e não de uma restrição arquitetural do sistema proposto.

De modo geral, os resultados indicam que a contribuição primordial deste trabalho não está em alterar as análises pedagógicas realizadas pelo SABIA, mas sim em modernizar a sua fundação técnica. Ao substituir processos frágeis por um fluxo íntegro, automatizado e consolidado em formato Parquet, o projeto demonstra que o atraso de maturidade em governança de dados no ensino superior (CERNEA; KIERYLO, 2024) pode ser efetivamente reduzido por meio da aplicação de padrões arquiteturais já validados pelo mercado, sustentando a gestão acadêmica com informações altamente confiáveis.

6 Conclusões

A automação do processo de ingestão, transformação e disponibilização das bases do SABIA, que antes dependia de extrações e consolidações manuais, mostrou-se tecnicamente viável e resolveu o principal gargalo identificado na introdução deste trabalho. Ao aplicar uma arquitetura sustentada por Apache Airflow, MinIO e bibliotecas de processamento em Python, foi possível transformar 14 rotinas isoladas em um pipeline orquestrado, acionado por um único ponto e capaz de processar toda a cadeia em poucos minutos.

Ao cruzar o que foi desenvolvido com as questões de pesquisa, fica claro que a transição para as DAGs resolveu o gargalo da dependência humana na execução sequencial (RQ1). Esse processo não se resumiu a agendar scripts; exigiu uma reestruturação lógica profunda dos ETLs e a inclusão de regras de qualidade de dados que antes dependiam da verificação manual de quem operava o fluxo (RQ2). Como resultado direto dessa mudança, o novo modelo passou a registrar cada execução individualmente e a permitir que, em caso de falha, apenas a etapa afetada seja reexecutada, algo que o ambiente anterior não oferecia (RQ3). A principal entrega do trabalho, portanto, é a garantia de que a infraestrutura que alimenta o SABIA agora é robusta, auditável e menos dependente.

Naturalmente, há limites práticos no que foi implementado. A anonimização dos dados do SIGAA continua sendo feita manualmente pelo professor responsável, antes de os dados chegarem ao Airflow. Essa etapa está fora do escopo da automação proposta neste trabalho, e sua eventual incorporação ao pipeline orquestrado é uma possibilidade de continuidade, condicionada às definições institucionais de segurança e privacidade sobre quem pode acionar esse processo.

Outro ponto de atenção envolve o arquivo de vagas da universidade. Os indicadores atuais derivados dele operam com valores simulados e devem ser vistos como uma validação técnica do pipeline, e não como dados institucionais definitivos, cenário que mudará assim que a base oficial for integrada ao MinIO. Há também restrições que fogem ao controle da arquitetura, como a indisponibilidade temporal dos dados do INEP referentes a 2024 e parte de 2025, o que limita temporariamente o horizonte das análises do sistema.

Um último desdobramento que merece destaque é o potencial de escalabilidade da solução. Como toda a lógica de extração e transformação foi moldada em cima da estrutura de dados do SIGAA, existe um caminho claro para exportar essa arquitetura. Considerando que dezenas de outras Instituições de Ensino Superior utilizam o SIGAA,

o pipeline consolidado neste trabalho pode perfeitamente servir como um modelo base para modernizar a governança de dados em outros ambientes fora do escopo local.

Referências

ALCANTARA, W. et al. Desafios no uso de dados abertos conectados na educação brasileira. In: *Anais do Workshop de Desafios da Computação Aplicada à Educação (DESAFIE)*. [S.l.: s.n.], 2015. p. 1–10. Citado na página 12.

ALMEIDA, J. M. d.; BRAGANHOLO, V.; OLIVEIRA, D. de. Governança de dados em sistemas-de-sistemas: Uma abordagem orientada à dados de proveniência. In: *Anais do Simpósio Brasileiro de Banco de Dados (SBBD)*. Porto Alegre, RS, Brasil: SBC, 2025. p. 182–195. Disponível em: <https://sol.sbc.org.br/index.php/sbbd/article/view/37237>. Citado 2 vezes nas páginas 17 e 18.

AMARE, M. Y.; SIMONOVA, S. Learning analytics for higher education: proposal of big data ingestion architecture. *SHS Web of Conferences*, EDP Sciences, v. 89, p. 01001, 2020. Citado na página 12.

Amazon Web Services. *O que é armazenamento de objetos?* 2024. Disponível em: <https://aws.amazon.com/pt/what-is/object-storage/>. Citado na página 23.

ANDRADE, T. L. d. et al. A utilização de metodologias ativas com suporte de mineração de dados educacionais e *Learning Analytics* para a mitigação da evasão em ead: um mapeamento sistemático da literatura. *Revista Brasileira de Informática na Educação*, v. 31, p. 1057–1088, 2023. Citado na página 16.

ANDRADE, T. L. d. et al. Mineração de dados educacionais e metodologias ativas integrados a um sistema de recomendação para prevenção da evasão na educação a distância. *Revista Brasileira de Informática na Educação*, v. 33, p. 748–772, 2025. Citado na página 16.

ARMBRUST, M. et al. Lakehouse: A new generation of open platforms that unify data warehousing and advanced analytics. In: *Proceedings of the 11th Conference on Innovative Data Systems Research (CIDR)*. Virtual Event: [s.n.], 2021. Disponível em: <https://www.cidrdb.org/cidr2021/papers/cidr2021_paper17.pdf>. Citado 2 vezes nas páginas 13 e 22.

BEZERRA, A. L. d. S. et al. Rpas e data lakes para a indústria 4.0: Um estudo de caso de ecossistema de dados integrados. In: *Anais do Simpósio Brasileiro de Banco de Dados (SBBD)*. Porto Alegre, RS, Brasil: SBC, 2025. p. 753–759. Disponível em: <https://sol.sbc.org.br/index.php/sbbd/article/view/37280>. Citado na página 18.

BRASIL. *Lei nº 10.861, de 14 de abril de 2004. Institui o Sistema Nacional de Avaliação da Educação Superior (Sinaes) e dá outras providências*. Brasília, DF: [s.n.], 2004. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2004-2006/2004/lei/l10.861.htm>. Citado na página 24.

BRASIL. *Lei nº 13.709, de 14 de agosto de 2018. Lei Geral de Proteção de Dados Pessoais (LGPD)*. Brasília, DF: [s.n.], 2018. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm>. Citado na página 25.

BRITO, D. et al. Data orchestration and infrastructure modernization for educational data mining in latin america. *IEEE Latin America Transactions*, 2023. Citado na página 13.

CEJUDO, A. et al. Scalable big data platform with end-to-end traceability for health data monitoring in older adults: Development and performance evaluation. *JMIR Medical Informatics*, v. 13, p. e81701, 2025. Citado 2 vezes nas páginas 13 e 15.

CERNEA, P.; KIERYLO, M. Implementing data governance: Insights and strategies from the higher education sector. *Journal of Financial Transformation*, v. 59, p. 60–69, 2024. Citado 5 vezes nas páginas 12, 13, 15, 24 e 50.

COLPO, M. P. et al. Mineração de dados educacionais na previsão de evasão: uma rsl sob a perspectiva do congresso brasileiro de informática na educação. In: *Anais do Simpósio Brasileiro de Informática na Educação (SBIE)*. Porto Alegre, RS, Brasil: SBC, 2020. Disponível em: <https://sol.sbc.org.br/index.php/sbie/article/view/12866>. Citado na página 16.

CORREIA, R. C. M. et al. Análise dos principais fatores que influenciam a evasão no ensino superior utilizando técnicas de mineração de dados educacionais. In: *Anais...* Presidente Prudente, SP: [s.n.], 2024. Universidade Estadual Paulista (UNESP). Citado 2 vezes nas páginas 16 e 17.

CUNHA, R.; SALES, C.; SANTOS, R. Análise automática com os microdados do enade para melhoria do ensino dos cursos de ciência da computação. In: *Anais do Workshop sobre Educação em Computação (WEI)*. Porto Alegre, RS, Brasil: SBC, 2021. p. 208–217. Disponível em: <https://sol.sbc.org.br/index.php/wei/article/view/15912>. Citado na página 17.

DANCEY, C. P.; REIDY, J. *Estatística Sem Matemática para Psicologia*. 7. ed. Porto Alegre: Penso Editora, 2018. Citado na página 42.

ESPíNDOLA, P. L. et al. Governança de dados aplicada à ciência da informação: análise de um sistema de dados científicos para a área da saúde. *RDBCI: Revista Digital de Biblioteconomia e Ciência da Informação*, Campinas, SP, v. 16, n. 3, p. 574–598, 2018. Disponível em: <https://periodicos.sbu.unicamp.br/ojs/index.php/rdbci/article/view/8651080>. Citado na página 24.

FINAMOR, L. et al. Integration of administrative educational data and standardized national exams in brazil: challenges and perspectives. *Revista Brasileira de Avaliação Educacional*, 2022. Citado na página 12.

FOIDL, H. et al. Data pipeline quality: Influencing factors, root causes of data-related issues, and processing problem areas for developers. *arXiv preprint arXiv:2309.07067*, 2023. Citado 3 vezes nas páginas 12, 20 e 23.

GOMES, F. P. et al. Desenvolvimento de um ambiente Data Warehouse para organizar e analisar os dados do Exame Nacional de Desempenho dos Estudantes (ENADE). In: CENTRO UNIVERSITÁRIO LUTERANO DE PALMAS (CEULP/ULBRA). *Anais...* Palmas, TO, 2017. Citado 2 vezes nas páginas 17 e 18.

GOUVêA, V. R. *Estudo comparativo de ferramentas de orquestração: Dagster e Airflow*. 101 p. Monografia (Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação)) — Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2025. Citado na página 19.

Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira – INEP. *Exame Nacional de Desempenho dos Estudantes (Enade)*. Brasília, DF: [s.n.], 2024. Disponível em: <https://www.gov.br/inep/pt-br/areas-de-atuacao/avaliacao-e-exames-educacionais/enade>. Citado 2 vezes nas páginas 24 e 26.

KANTORSKI, G. Z. et al. Mineração de dados educacionais para predição da evasão em cursos de graduação presenciais no ensino superior. In: *Anais do Simpósio Brasileiro de Informática na Educação (SBIE)*. Porto Alegre, RS, Brasil: SBC, 2023. p. 1133–1142. Disponível em: <https://sol.sbc.org.br/index.php/sbie/article/view/26741>. Citado na página 17.

KAPHLE, B.; SHRESTHA, B. A Lakehouse-Oriented Big Data Infrastructure for Educational Analytics: Integrating Administrative and Assessment Data for Early Student Risk Prediction. *International Journal of Information Technology and Computer Science Applications (IJITCSA)*, v. 4, n. 1, p. 47–58, 2026. ISSN 2985-5330. Citado na página 16.

LAKSITOWENING, K. A. et al. Incorporating Learning Analytics and Business Intelligence into Higher Education E-Learning. *Khazanah Informatika: Jurnal Ilmu Komputer dan Informatika*, v. 11, n. 1, p. 127–134, 2025. Citado na página 17.

LINHARES, L. R. *Uma Arquitetura de Dados Modular para Predição de Curtailment no Setor Elétrico Brasileiro*. Dissertação (Mestrado) — Universidade Federal de Lavras, Lavras, MG, 2025. Trabalho de Conclusão de Curso. Citado 2 vezes nas páginas 18 e 19.

LOUREIRO, J.; OLIVEIRA, D. de. Orbiter: um arcabouço para implantação automática de aplicações big data em arquiteturas serverless. In: *Proceedings of the 37th Brazilian Symposium on Data Bases (SBBD)*. Búzios, RJ: SBC, 2022. p. 379–384. Disponível em: <https://sol.sbc.org.br/index.php/sbbd/article/view/21823>. Citado na página 23.

MAHMUD, D.; IKBAL, M. Z. The Role of ETL (Extract-Transform-Load) Pipelines in Scalable Business Intelligence: A Comparative Study of Data Integration Tools. In: ASRC PROCEDIA: GLOBAL PERSPECTIVES IN SCIENCE AND SCHOLARSHIP. *World Summit on Scientific Research and Innovation (WSSRI)*. Florida, USA, 2022. v. 2, n. 1, p. 89–121. Citado na página 17.

MARQUES, E. et al. Sabia: Uma plataforma para auxiliar a gestão baseada em evidências nas instituições de ensino superior. In: *Anais do Workshop de Aplicações Práticas de Learning Analytics e Inteligência Artificial no Brasil (WAPLA), CBIE 2023*. SBC, 2023. Disponível em: <https://sol.sbc.org.br/index.php/wapla/article/view/26130>. Citado 2 vezes nas páginas 25 e 26.

MinIO. *MinIO Object Storage Documentation*. 2024. Disponível em: <https://min.io/docs/minio/linux/index.html>. Citado na página 23.

MONTEIRO, A. et al. Sadd - sistema de armazenamento e disponibilização de dados oceanográficos. In: *Companion Proceedings of the 40th Brazilian Symposium on Data Bases (SBBD)*. Fortaleza, CE: SBC, 2025. p. 117–122. Disponível em: <https://sol.sbc.org.br/index.php/sbbd_estendido/article/download/37578/37360>. Citado na página 23.

MUNAPPY, A. R. et al. Data pipeline management in practice: Challenges and opportunities. In: *Product-Focused Software Process Improvement (PROFES 2020)*. [S.l.: s.n.], 2020. Citado 2 vezes nas páginas 12 e 20.

NETO, A.; NERES, M.; CARVALHO, M. *Uma Arquitetura de Data Lake de Dados Ambientais e Socioeconômicos para Fomentar Pesquisa e Inovação*. Porto Alegre, RS: SBC, 2024. Disponível em: <https://sol.sbc.org.br/index.php/wcama/article/view/36107>. Citado na página 20.

RAMOS, G. S. et al. Data lakes lógicos como plataformas para dados governamentais em sociedades e cidades inteligentes. In: *Anais do Latin American Symposium on Digital Government (LASDiGov / WCGE)*. Porto Alegre, RS, Brasil: SBC, 2022. p. 215–226. Disponível em: <https://sol.sbc.org.br/index.php/wcge/article/view/20724>. Citado na página 16.

RIBEIRO, M. et al. Automated etl pipelines for large scale academic records processing. *Brazilian Symposium on Databases (SBBD)*, 2022. Citado na página 13.

ROLIM, T. V. et al. Construção do dataset semântico de pessoas jurídicas. In: *Anais do Dataset Showcase Workshop (DSW)*. Florianópolis, SC: SBC, 2024. Disponível em: <https://sol.sbc.org.br/index.php/dsw/article/view/30614>. Citado na página 22.

SALAMI, S. *Hub Star Modeling 2.0 for Medallion Architecture*. 2025. Disponível em: <https://arxiv.org/abs/2504.08788>. Citado 2 vezes nas páginas 22 e 23.

SALLES, G. F. de. *Extração, transformação e carga de dados sobre compras da Prefeitura de Florianópolis usando Airflow*. 64 p. Monografia (Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação)) — Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2025. Citado 2 vezes nas páginas 18 e 29.

SANTOS, J. A. d.; FUINI, M. G. Desvendando o ENADE: uma análise abrangente dos dados e resultados dos cursos superiores. *Prospectus*, 2024. ISSN 2674-8576. Citado 2 vezes nas páginas 17 e 18.

SILVA, H. A. B. d. et al. Uma abordagem para a gestão da linhagem de dados heterogêneos. In: *Anais do Simpósio Brasileiro de Banco de Dados (SBBD)*. Porto Alegre, RS, Brasil: SBC, 2025. p. 630–643. Disponível em: <https://sol.sbc.org.br/index.php/sbbd/article/view/37270>. Citado na página 18.

SILVA, M. da. *Detecção de evasão escolar no ensino superior: um pipeline otimizado com stacking XGBoost-LightGBM, KMeansSMOTE e otimização bayesiana*. Monografia (Trabalho de Conclusão de Curso (Especialização em Inteligência Artificial Aplicada)) — Setor de Educação Profissional e Tecnológica, Universidade Federal do Paraná, Curitiba, 2025. 2025. Citado na página 16.

SINGHAL, B.; AGGARWAL, A. Etl, elt and reverse etl: A business case study. In: *2022 Second International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE)*. Bangalore, India: IEEE, 2022. p. 1–4. Citado na página 21.

SINGU, S. K. Etl process automation: Tools and techniques. *ESP Journal of Engineering & Technology Advancements*, v. 2, n. 1, p. 74–85, mar 2022. Disponível em: <https://www.researchgate.net/profile/Santosh-Kumar-Singu/publication/386874870_ETL_Process_Automation_Tools_and_Techniques/links/675a3011951ca355613eac3b/ETL-Process-Automation-Tools-and-Techniques.pdf>. Citado na página 20.

SOUSA, I. C. de. *Arquitetura conceitual para automação de ETL de microdados públicos: um estudo de caso com o ENEM*. 25 p. Monografia (Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Computação)) — Instituto Federal da Paraíba, Campina Grande, 2025. Citado na página 18.

Universidade Federal do Rio Grande do Norte – UFRN. *SIGAA - Sistema Integrado de Gestão de Atividades Acadêmicas*. Natal, RN: [s.n.], 2024. Superintendência de Informática (SINFO). Disponível em: <https://docs.info.ufrn.br/>. Citado na página 25.

VANAM, L. Etl optimization for scalable bi in financial enterprises. *International Journal of Computational and Experimental Science and Engineering (IJCESEN)*, v. 11, n. 3, p. 6526–6535, 2025. Citado 2 vezes nas páginas 13 e 15.

VOHRA, D. Apache parquet. In: *Practical Hadoop Ecosystem*. Berkeley, CA: Apress, 2016. Citado na página 23.

WANG, R. Y.; STRONG, D. M. Beyond accuracy: what data quality means to data consumers. *Journal of Management Information Systems*, v. 12, n. 4, p. 5–33, 1996. Citado na página 24.