INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO CÂMPUS CAMPINAS

DEIVE AUDIERES LEAL

**AVALIAÇÃO DE ESTRATÉGIAS PARA A CRIAÇÃO DE PIPELINE DE DADOS VOLTADO PARA CIÊNCIAS HUMANAS**

CAMPINAS

2021

DEIVE AUDIERES LEAL

**AVALIAÇÃO DE ESTRATÉGIAS PARA A CRIAÇÃO DE PIPELINE DE DADOS VOLTADO PARA CIÊNCIAS HUMANAS**

Trabalho de Conclusão de Curso apresentado como exigência parcial para obtenção do diploma do Curso de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia Câmpus Campinas.

Orientador: Prof. Dr. Samuel Botter Martins.

CAMPINAS

2021

**Dados Internacionais de Catalogação na Publicação**

Solicitar a ficha catalográfica pelo sistema Pergamum, (Meu Pergamum, solicitações, ficha catalográfica) após as correções sugeridas pela banca. Apresenta-se no verso da página de rosto

Deive Audieres Leal

**Avaliação de estratégias para a criação de pipeline de dados voltado para ciências humanas**

Trabalho de Conclusão de Curso apresentado como exigência parcial para obtenção do diploma do Curso de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo Câmpus Campinas.

Aprovado pela banca examinadora em: \_\_\_\_\_ de \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ de \_\_\_\_\_.

BANCA EXAMINADORA

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Dr. Samuel Botter Martins (orientador)

IFSP Câmpus Campinas

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof.a Me xxxxxxxxxxxxxxxx

Universidade Estadual de Campinas

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Me xxxxxxxxxxxxxxxx

IFSP Câmpus XXXX

*Dedico este trabalho aos meus familiares,*

*colegas de classe e àquelas pessoas que,*

*de uma forma ou de outra,*

*são inspirações para sempre ir em frente.*

**AGRADECIMENTOS**

Agradeço primeiramente a Deus, pelo dom da vida e pela oportunidade de concluir mais uma etapa de minha experiência acadêmica. Agradeço a todos os professores e servidores do IFSP Campus Campinas, que contribuíram direta e indiretamente para a conclusão desse trabalho. Agradeço também à minha família, que deu todo o apoio necessário para que eu chegasse até aqui. Agradeço ao meu orientador que me auxiliou a solucionar as dificuldades encontradas no caminho.

*"A força não provém da capacidade física.*

*Provém de uma vontade indomável."*

*Mahatma Gandhi*

**RESUMO**

O presente trabalho visa o desenvolvimento de pipeline de dados tendo como foco as ciências humanas. Assim, há análise de estratégias ferramentais que possibilitem a implementação desses fluxos de dados. Parte-se da premissa de que todo o ambiente e ferramentais devem ser de código aberto ou voltado para código aberto. Da mesma forma, espera-se que o ambiente final de disponibilização dos dados seja de Big Data. Assim, emprega-se o uso do Hadoop e Hive para armazenamento e interface para a manipulação dos dados. Utiliza-se o modelo de Extração, Carregamento e Transformação, ELT, para caracterizar as tarefas a serem realizadas e o sistema de camadas para armazenamento.

Os pipelines foram construídos utilizando Pentaho Data Integration e Python com Airflow, utilizando dados públicos brasileiros. O projeto visa diminuir a distância entre as áreas de conhecimento possibilitando que boas perguntas sejam feitas e que contenham dados disponíveis para que hipóteses de ação ou resposta possam ser formuladas.

**Palavras-chave**: ELT, Big Data, Engenharia de Dados, Software Livre.

**ABSTRACT**

Versão do resumo em língua inglesa.

**Keywords**: Xxxxxxx. Xxxxxxx. Xxxxxxx. Xxxxxxx. Xxxxxxx.

**LISTA DE FIGURAS**

[Figura 1: Carreiras em dados 18](#Figura!0|sequence)

[Figura 2: Processo de ETL 19](#Figura!1|sequence)

[Figura 3: Processo de ETL - Destaque da extração 20](#Figura!2|sequence)

[Figura 4: Processo de ETL - Destaque da transformação 20](#Figura!3|sequence)

[Figura 5: Processo de ETL - Destaque da carga 21](#Figura!4|sequence)

[Figura 6: Processo de ELT 22](#Figura!5|sequence)

[Figura 7: Arquitetura Hive 25](#Figura!6|sequence)

[Figura 8: Versões do Pentaho. Fonte: Hitachi Vantara 26](#Figura!7|sequence)

[Figura 9: 10 Linguagens de programação mais utilizadas 2021. Fonte IEEE 27](#Figura!8|sequence)

[Figura 10: Arquitetura do Airflow. Fonte: Apache Airflow 28](#Figura!9|sequence)

[Figura 11: Arquitetura geral do pipeline 30](#Figura!10|sequence)

[Figura 12: Pentaho - tela inicial 32](#Figura!11|sequence)

[Figura 13: Pentaho - Arquivos de download 32](#Figura!12|sequence)

[Figura 14: Migração de dados na camada bronze 34](#Figura!13|sequence)

[Figura 15: Sugestão do Pentaho para a criação de tabela 35](#Figura!14|sequence)

[Figura 16: Camada silver 36](#Figura!15|sequence)

[Figura 17: Camada gold 36](#Figura!16|sequence)

[Figura 18: Airflow tela inicial 37](#Figura!17|sequence)

[Figura 19: Tipos de instalação do Airflow 37](#Figura!18|sequence)

[Figura 20: Página de documentação - detalhe 38](#Figura!19|sequence)

[Figura 21: Visual Studio Code 39](#Figura!20|sequence)

[Figura 22: Etapas em uma DAG de carregamento de arquivo csv 40](#Figura!21|sequence)

[Figura 23: Detalhe da tarefa de migração 40](#Figura!22|sequence)

[Figura 24: Detalhe da conversão de json para dataframe 40](#Figura!23|sequence)

[Figura 25: Transição para silver - detalhe 41](#Figura!24|sequence)

[Figura 26: Junção das tabelas de educação e localização geográfica 41](#Figura!25|sequence)

**LISTA DE TABELAS**

Tabela 1 – Softwares utilizados...................................................................................... 15

Tabela 2 – Hardware utilizado........................................................................................ 26

**LISTA DE SIGLAS**

API *Application Programming Interface*

BI *Business Intelligence*

CSV *Comma-separated values*

DAG *Directed Acyclic Graph*

DW *Data Warehouse*

ELT *Extract, Load, Transform*

ETL *Extract, Transform, Load*

HDFS *Hadoop Distributed File System*

HQL *Hive Query Language*

OLAP *Online Analytical Processing*

PDI *Pentaho Data Integration*

SQL *Structured Query Language*

**SUMÁRIO**

[1 INTRODUÇÃO 19](#__RefHeading___Toc3718_1625992954)

[2 OBJETIVOS 20](#__RefHeading___Toc3720_1625992954)

[2.1 Objetivo Geral 20](#__RefHeading___Toc3722_1625992954)

[2.2 Objetivos Específicos 20](#__RefHeading___Toc3724_1625992954)

[3 FUNDAMENTAÇÃO TEÓRICA 21](#__RefHeading___Toc3726_1625992954)

[3.1 Engenharia de Dados 21](#__RefHeading___Toc4329_1625992954)

[3.2. ETL e ELT 22](#__RefHeading___Toc4331_1625992954)

[3.2.1. ETL 22](#__RefHeading___Toc1550_3470351174)

[3.2.2. Extração 22](#__RefHeading___Toc1552_3470351174)

[3.2.3. Transformação 23](#__RefHeading___Toc1554_3470351174)

[3.2.4. Carga 24](#__RefHeading___Toc1556_3470351174)

[3.2.5. ELT 24](#__RefHeading___Toc1558_3470351174)

[3.3. Data Warehouse 25](#__RefHeading___Toc4338_1625992954)

[3.4. Hadoop 26](#__RefHeading___Toc1482_3470351174)

[3.5. Hive 27](#__RefHeading___Toc1484_3470351174)

[3.6. Pentaho Data Integration 28](#__RefHeading___Toc12656_3616532468)

[3.7. Python e Airflow 30](#__RefHeading___Toc3730_2763659088)

[3.7.1. Python 30](#__RefHeading___Toc3732_2763659088)

[3.7.2. Airflow 31](#__RefHeading___Toc3734_2763659088)

[4 Metodologia 33](#__RefHeading___Toc3736_2763659088)

[5 DESENVOLVIMENTO 34](#__RefHeading___Toc3738_2763659088)

[5.1 Pentaho 35](#__RefHeading___Toc3732_1625992954)

[5.1.1 Instalação 35](#__RefHeading___Toc3740_2763659088)

[5.1.2 Documentação Oficial 36](#__RefHeading___Toc3742_2763659088)

[5.1.3 Aprendizagem 36](#__RefHeading___Toc3744_2763659088)

[5.1.4 Execução 36](#__RefHeading___Toc3746_2763659088)

[5.1.4.1 Camada bronze 37](#__RefHeading___Toc3748_2763659088)

[5.1.4.2. Camada silver 38](#__RefHeading___Toc3750_2763659088)

[5.1.4.3. Camada gold 39](#__RefHeading___Toc3752_2763659088)

[5.2 Airflow e Python 39](#__RefHeading___Toc4991_2763659088)

[5.2.1 Instalação 40](#__RefHeading___Toc4993_2763659088)

[5.2.2 Documentação oficial 40](#__RefHeading___Toc4995_2763659088)

[5.2.3 Aprendizagem 41](#__RefHeading___Toc4997_2763659088)

[5.2.4 Execução 41](#__RefHeading___Toc4999_2763659088)

[5.2.4.1 Camada bronze 42](#__RefHeading___Toc5001_2763659088)

[5.2.4.2 Camada silver 44](#__RefHeading___Toc5003_2763659088)

[5.2.4.3 Camada gold 44](#__RefHeading___Toc5005_2763659088)

[6 CONSIDERAÇÕES FINAIS 45](#__RefHeading___Toc3734_1625992954)

[REFERÊNCIAS 46](#__RefHeading___Toc5007_2763659088)

# 1 INTRODUÇÃO

Com o aumento acelerado de produção de dados e o amplo uso dentro dos mais diversificados cenários e objetivos, a engenharia de dados se destaca no sentido de manter e criar uma base sólida de dados, que então poderão ser consumidos, seja por cientistas de dados ou engenheiros de aprendizado de máquina ou analistas de dados. O engenheiro de dados tem entre suas atribuições a construção e manutenção de arquiteturas direcionadas a dados, tais como banco de dados e sistemas de processamento em grande escala. Dentro da gama de possibilidades, a ETL 一 do inglês, *Extract, Transform, Load* 一 ou extração, transformação e carregamento é uma das etapas que merece destaque.

Nessa etapa, a extração corresponde ao agrupamento das mais diversas fontes de dados. Nela os dados são recuperados em sua forma crua, seja em bancos relacionais, planilhas, arquivos, ou outras formas de armazenamento de dados. Por sua vez, a transformação visa tornar os dados aptos para a utilização. Assim, são utilizadas técnicas de limpeza, classificação 一 categóricos e contínuos 一, e normalização sobre os dados. Por fim, o carregamento dos dados dá um local de acesso ao consumidor dos dados (por exemplo, o cientista de dados). Exemplos de locais de acesso são API – *Application Programming Interface* – e *data warehouse*.

*Data warehouse* é um banco de dados estruturado voltado para consultas avançadas, com grande quantidade de dados históricos e direcionado para suportar análises. Nele são centralizados uma grande quantidade de dados advindos de diferentes fontes que passaram pelo processo de ETL.

Sendo uma etapa tão importante e que tem uma massa de dados que pode chegar à casa de terabytes, foram surgindo diferentes ferramentas que auxiliassem no processo de ETL. Assim, temos desde ferramentas open source a ferramentas proprietárias, mas todas voltadas para maximizar e melhorar as informações que são angariadas. Visando o uso de ferramentas open source para a execução do ETL, há uma ausência de estudos que comparem as diferentes soluções existentes, seus prós e contras, bem como quais cenários cada ferramenta é indicada.

# 2 OBJETIVOS

### 2.1 Objetivo Geral

Desenvolver modelo de pipeline de dados via processo de ELT para Big Data com ferramentas de integração de dados para criação de base com enfoque em ciências humanas.

### 2.2 Objetivos Específicos

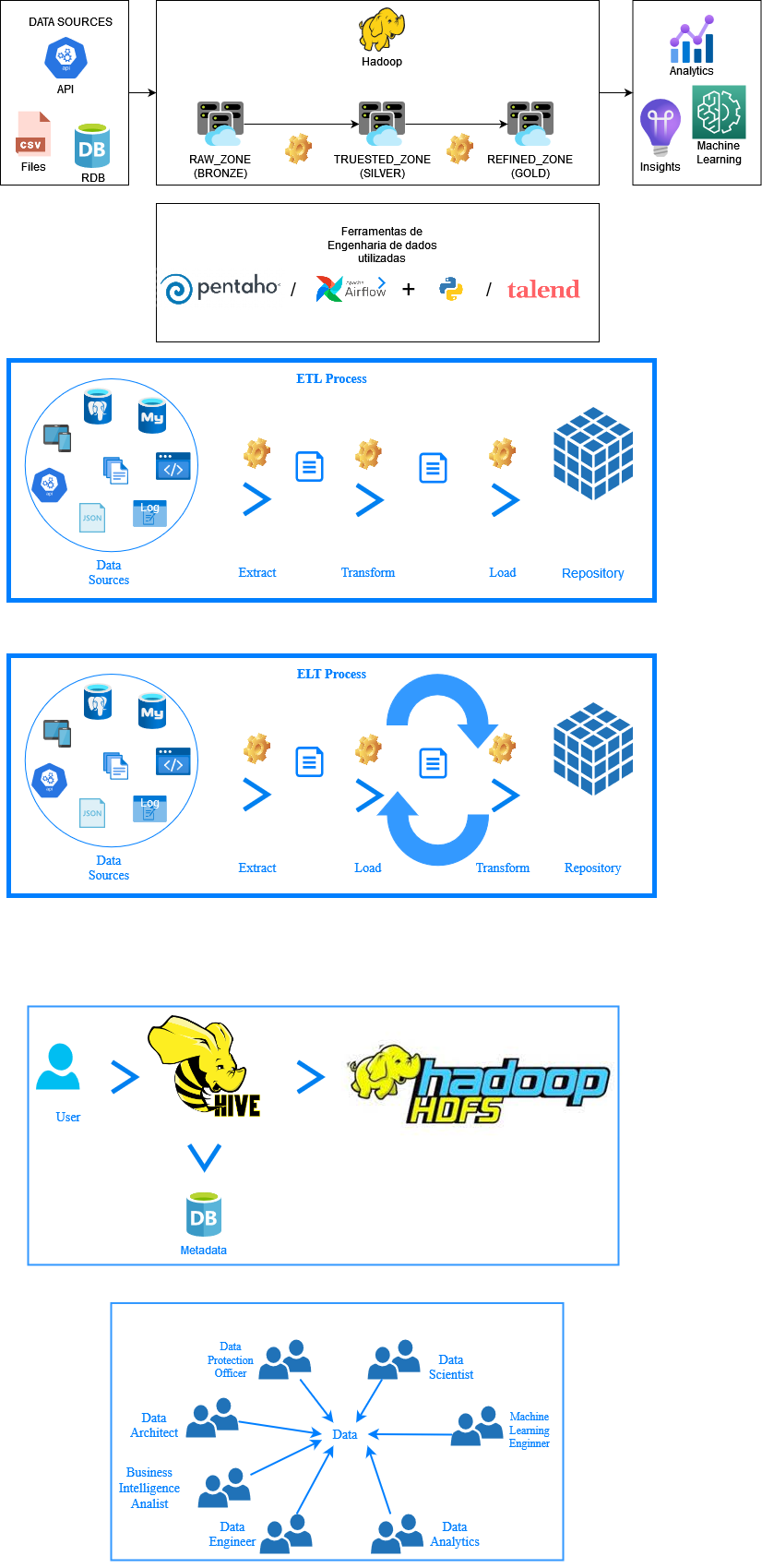
* Desenvolver um processo de ELT aplicável financeiramente e acessível para manutenção e reprodução
* Armazenar em ambiente de *big data*
* Utilizar unicamente ferramentas *open source*
* Integrar dados para que estejam apropriados para análise ou consumo final

# 3 FUNDAMENTAÇÃO TEÓRICA

### 3.1 Engenharia de Dados

Muitas são as possibilidades de quem deseja se aventurar com dados e desenvolver uma carreira na área, vide Figura 1.

Entre a gama de oportunidades advindas a engenharia de dados é um campo que se consolida como sendo essencial em ambientes direcionados a dados, e que o cenário exige tratamento de grandes volumes de dados brutos e disponibilidade.

Figura 1: Carreiras em dados

O engenheiro de dados trabalha na garantia de que os dados coletados possam ir da fonte ao consumidor através de fluxos, automáticos ou não, mas que democratizam o acesso aos dados pelas diferentes áreas do negócio tornando-as mais independentes, produtivas.

De um profissional dedicado à área é esperado que tenha conhecimento de banco de dados relacionais, não-relacionais, linguagem de programação, em especial SQL, e entendam como otimizar e recuperar dados. Seu trabalho é direcionado de acordo com as regras de negócio para o qual desenvolve o fluxo de dados.

WHITE (2018), observa que o engenheiro de dados pode estar alocado em ao menos três funções principais. A primeira, generalista, é responsável desde a aquisição de dados até a análise e visualização. A segunda é centralizada em pipeline, e trabalha em proximidade com os cientistas de dados na utilização dos dados coletados. Por fim, há o que é centralizado no banco de dados, sua dedicação é centrada no banco de dados analítico, trabalha com o desenvolvimento de esquemas, tabelas, data warehouses e armazenamento distribuído.

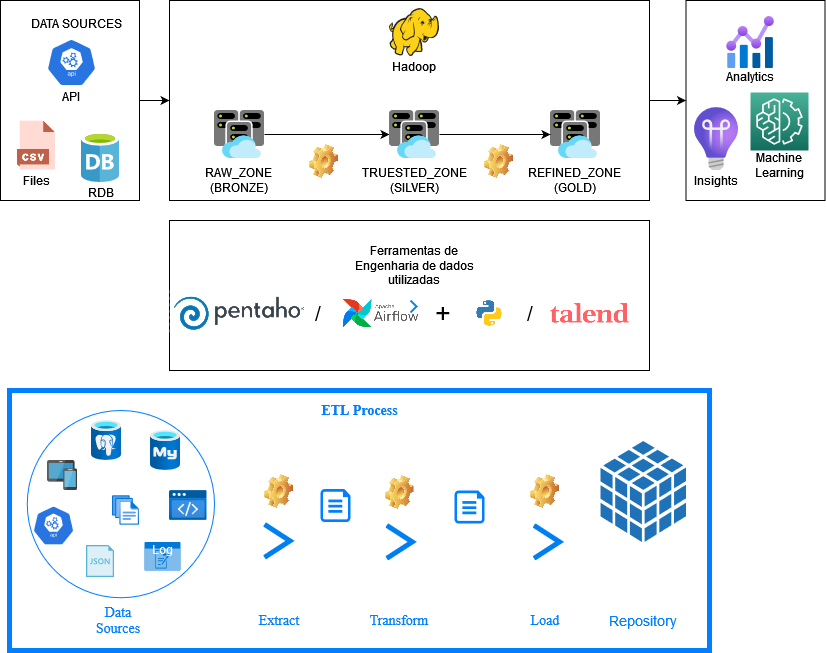
Tamir et all (2015) dizem que o engenheiro de dados trabalha na extração e construção de uma grande base de dados possibilitando o desenvolvimento de insights. Para eles, engenharia de dados, não se resume à simples manutenção de um repositório para grande volume de dados, mas sobre possibilitar com que todos, desenvolvedores ou não, possam consumi-los nas tarefas diárias, aumentando a gama de usuários e dialogando com a universalização e democratização dos dados.

Entre as atividades desempenhadas pelo engenheiro de dados, destaca-se a extração, transformação e carregamento de dados. Na bibliografia é comum encontrarmos a sigla em inglês ETL, – *Extract, Transform, Load* –, assim como sua variante ELT – *Extract, Load, Transform* –, assim sendo, também usaremos as siglas ao invés das variações completas com seus significados. Mas ressaltamos que antes daremos um entendimento geral sobre esse processo, que nos servirá de base para o fluxo de dados proposto na arquitetura deste trabalho.

### 3.2. ETL e ELT

### 3.2.1. ETL

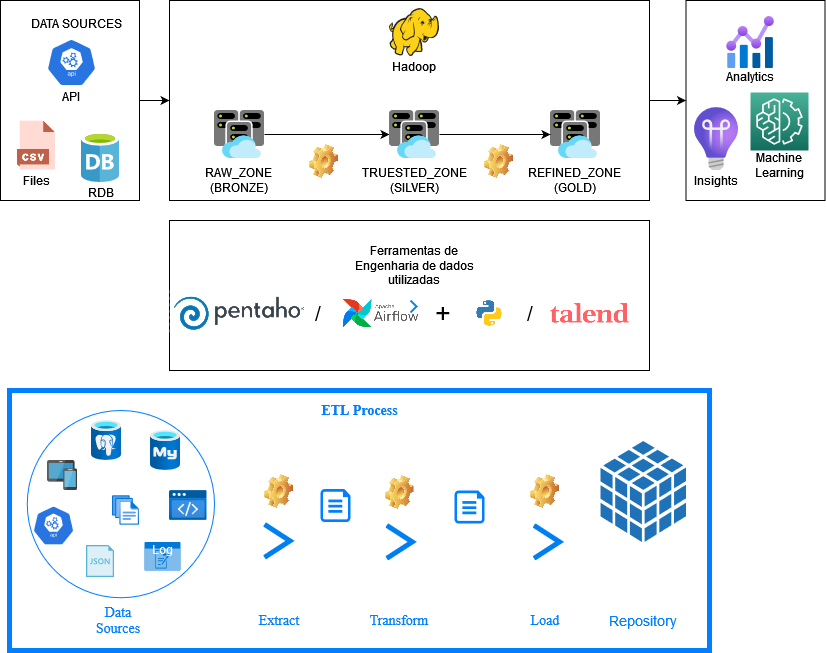
O processo de ETL é a etapa em que os dados são extraídos da fonte inicial, seja elas um banco ou diversos bancos, com alta ou baixa granularidade, recebem então tratamento, se necessário, e é realizado o carregamento no repositório. No formato ETL o dado chega pronto ao repositório, imobilizando qualquer intervenção posterior.

Figura 2: Processo de ETL

### 3.2.2. Extração

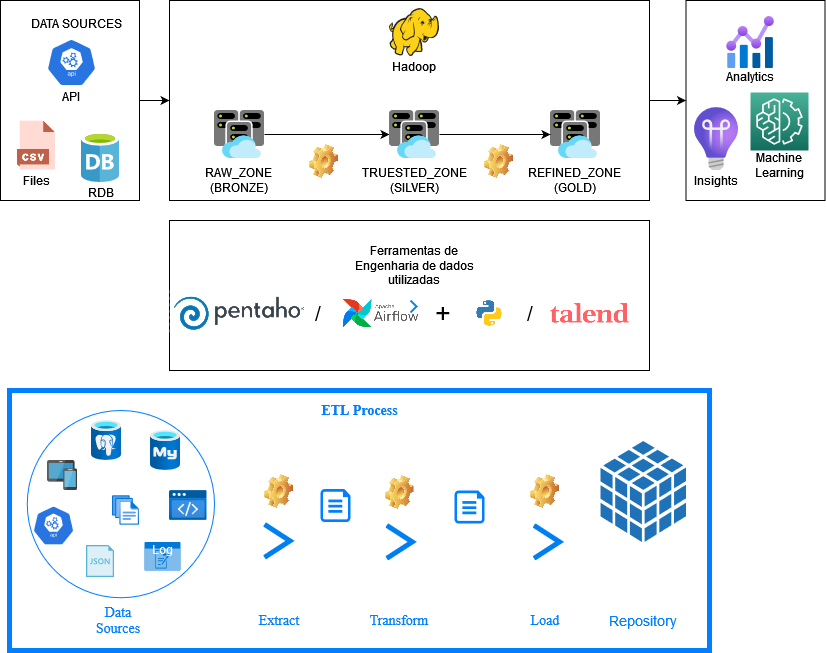
A extração, *Extract,* é a primeira fase do ETL. Nela é realizada a coleta de dados de diferentes fontes. As origens têm características próprias que são gerenciadas e só então realiza-se a extração. Deve-se observar a estrutura das fontes para que possam ser ajustadas ao repositório que as receberá.

Os dados são recuperados em sua forma crua, seja em bancos relacionais, arquivos estruturados como planilhas e csv, fontes on-line, como API’s, e assim por diante. É um processo que, por sua vez, pode ser separado em dois outros, a importação dos dados em si e a atualização dos dados periodicamente.

Figura 3: Processo de ETL - Destaque da extração

### 3.2.3. Transformação

Por sua vez, a transformação, *transform*, se ocupa em tornar os dados aptos para a utilização. Assim, são utilizadas técnicas de limpeza, classificação 一 categóricos e contínuos 一, e normalização sobre os dados. Assim sendo, tudo o que não condiz com o que é esperado passam por um procedimento de exclusão ou tratamento levando em conta as regras de negócio definidas.

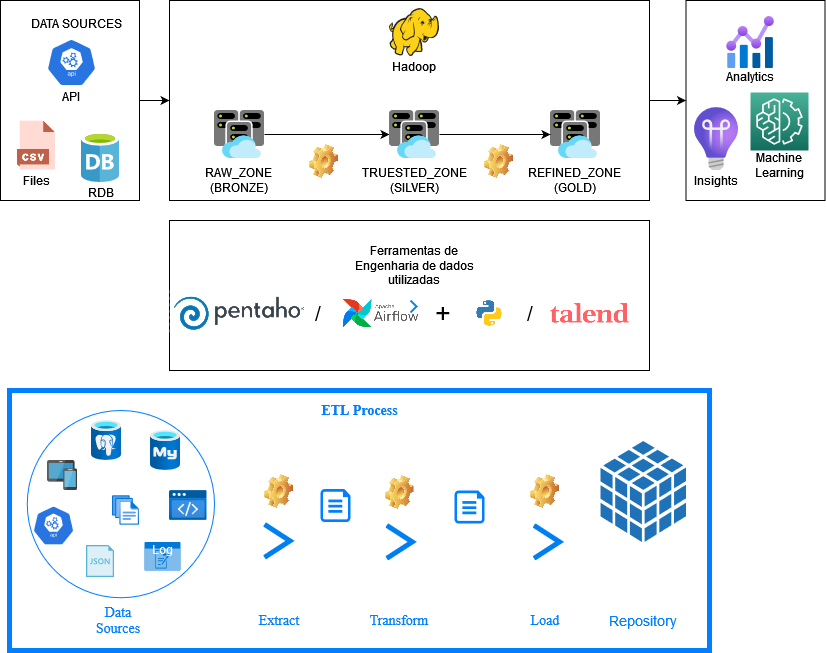
Figura 4: Processo de ETL - Destaque da transformação

Tais regras estão em conformidade com o destino final, garantindo a confiabilidade do processo. De maneira geral, os dados podem ser convertidos, padronizados e tipificados para se enquadrarem em seu uso. Assim, para análise de dados um formato pode ser especificado enquanto que para um projeto que envolva aprendizado de máquina, outra abordagem pode vir a ganhar destaque.

### 3.2.4. Carga

Por fim, a última fase do ETL, é a carga, *load*, dos dados. Nela os dados são gravados em um repositório de destino que permite o acesso ao consumidor. No caso do *data warehouse*, os dados podem ser carregados em tabelas dimensão e tabelas fato, que fazem parte do cubo de dados OLAP, *Online Analytical Processing*.

O cubo OLAP é um tipo de estrutura que expande os limites dos bancos de dados relacionais amplamente utilizados, melhorando a velocidade para a análise dos dados. Eles têm a capacidade de mostrar e executar cálculos de grandes quantidades de dados ao mesmo tempo que fornecem ao usuário meios de trabalhar tais dados. Assim sendo, as possibilidades de utilização analítica aumentam consideravelmente através da concentração, divisão e inclusão de dados de acordo com a demanda e os questionamentos que o uso dos dados podem vir a dar suporte para responder.

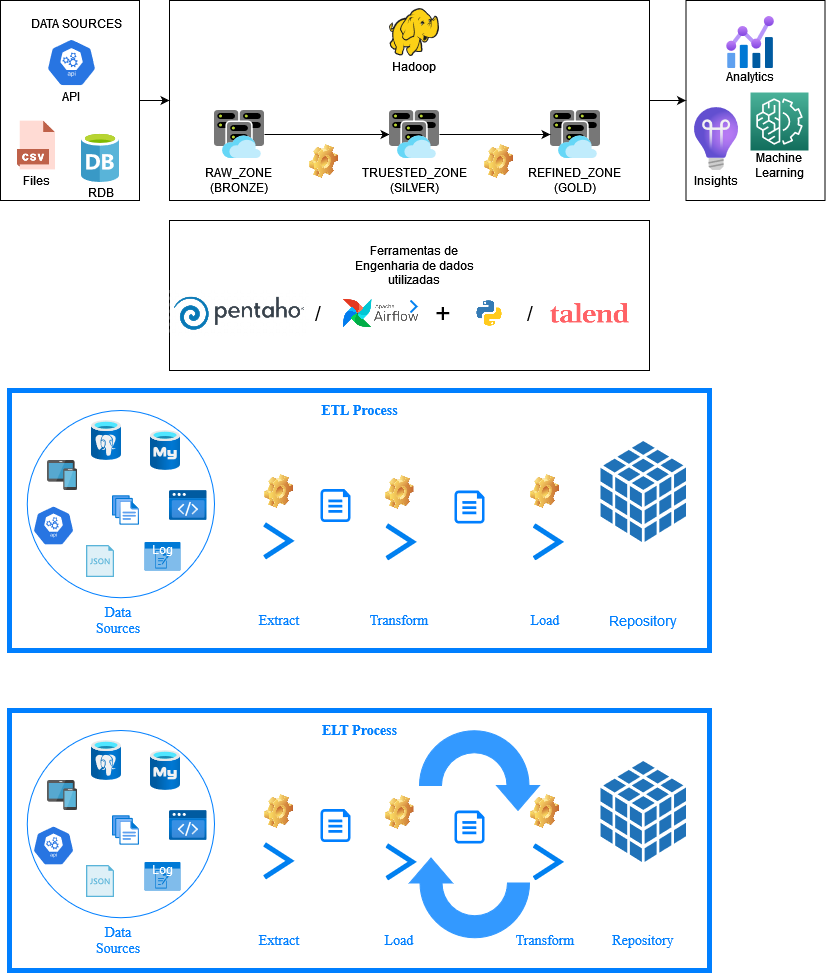
Figura 5: Processo de ETL - Destaque da carga

### 3.2.5. ELT

Com relação ao ETL há uma variação, o ELT, onde inverte-se a carga dos dados com a transformação. Nesse modelo, os dados são extraídos da fonte primária e carregados em uma base que aglutinará os diversos tipos de dados.

Com os dados já armazenados e acessíveis o sistema poderá então realizar as transformações e normalizações necessárias de modo mais otimizado, pois os dados estarão no mesmo ambiente.

Para o trabalho com Big Data convencionou-se a utilização do modelo ELT, pois os dados podem ser reestruturados para gerar novas tabelas que o processo no formato ETL não permite. Tendo em vista que gravaremos em um sistema de armazenamento voltado para Big Data, nossa escolha, então, recaiu sobre o ELT.

Figura 6: Processo de ELT

### 3.3. Data Warehouse

Surgindo de um conceito acadêmico da década de 1980, o *Data Warehouse*, DW, vem suprir a necessidade de organizar e obter dados de bases transacionais, que realizam apenas a ação de inserir, atualizar e excluir nos bancos de dados relacionais, como registros de vendas de mercadorias, e favorecer a análise de dados de negócio ou criação de relatórios de gestão por parte das equipes de *Business Intelligence*, BI, ou *Data Analytics*.

O DW é compreendido como parte de um planejamento para a organização eficiente dos dados e direcionado para dar suporte às estratégias empresariais. Os dados armazenados são mantidos por um longo período, permitindo que as análises desses dados, previamente consolidados, agrupados e indexados, possam ser estendidas na dimensão tempo. Proporciona o gerenciamento de um grande volume de dados que estariam, a princípio, dispersos em múltiplas fontes, sendo então concentrados e possibilitando a análise de dados coletados através de sistemas transacionais.

Por sua vez, complementando no que se relaciona a análise de dados, o OLAP, *Online Analytical Processing*, favorece a visualização e exploração dos dados. Tais tecnologias são voltadas para sustentar consultas sofisticadas e incomuns, assim como combinar informações por meio de análises comparativas e visualizações customizadas através do tempo. Tais tecnologias possibilitam observações claras de dados multidimensionais, através de compreensões múltiplas e variadas.

Os DW são modelados pensando que o sistema está em constante crescimento e que os dados devem estar integrados, são orientados por assunto e posições históricas. Os dados não são atualizados ou excluídos, mas apenas inseridos iterativamente ou substituídos por uma nova inserção. Os DW podem ser ainda divididos em data mart, que são subconjuntos voltados para um tema ou assunto específico

As abordagens normalizada e dimensional são as mais usadas para o armazenamento. Na perspectiva normalizada as tabelas são agrupadas por zonas temáticas que refletem as categorias dos dados em geral. Já na perspectiva dimensional os dados são divididos em fatos, os dados numéricos da transação, e em dimensões, informações de referência que contextualizam os fatos.

Porém, ao se falar em Big Data, o DW convencional não é capaz de armazenar ou processar essa massa de dados, somos então levados a utilizar sistemas distribuídos para melhorar os resultados nessa abordagem. Nesse sentido, o Apache Hadoop, é o *framework,* conjunto de ferramentas, bibliotecas e funcionalidades inseridas em uma base estrutural única*,* mais utilizado no momento com essa finalidade. Passamos assim a apresentá-lo.

### 3.4. Hadoop

O Apache Hadoop é um *framework* que surge da necessidade de utilização da computação distribuída para a análise de dados de grande volume e diversificado. Inicialmente criado dentro do Google para melhorar a performance das pesquisas, através da tecnologia MapReduce, foi concretizado e nomeado para Hadoop dentro do Yahoo, alguns anos depois, sendo também disponibilizado como software *open source* pela Apache Software Fundation desde então.

Entre as suas principais características, e que o tornaram amplamente utilizado, encontram-se:

* Código Aberto: como software livre, ele tem uma grande comunidade por trás, dando suporte ao desenvolvimento de suas ferramentas, encontrando problemas, melhorando a estabilidade e mantendo em desenvolvimento constante.
* Economia: Por ser um software de código aberto ele possibilita a análise e adaptação de suas funcionalidades a qualquer interessado. A análise e o processamento de dados podem ser feitos por máquinas e rede convencionais, diminuindo o custo com poderosos sistemas computacionais. Empresas – como Amazon, Google e Azure que são especializadas em computação em nuvem, implementaram versões próprias do Hadoop – podem “alugar” seu parque de computadores, diminuindo os custos com manutenção e compra por parte do cliente.
* Robustez: Possui mecanismos que garantem a integridade dos dados: replicando, armazenando metadados e informações de processamento.
* Escalabilidade: O Hadoop é configurável, permitindo que, independente da quantidade de máquinas que serão utilizadas, todas trabalhem em conjunto em um esforço menor de trabalho técnico e com efetividade.
* Simplicidade: Questões relativas à computação paralela, tais como tolerância a falhas, escalonamento e balanceamento de carga não ficam na responsabilidade do desenvolvedor, permitindo que ele direcione sua atenção ao problema que será processado no modelo *Map* e *Reduce*.

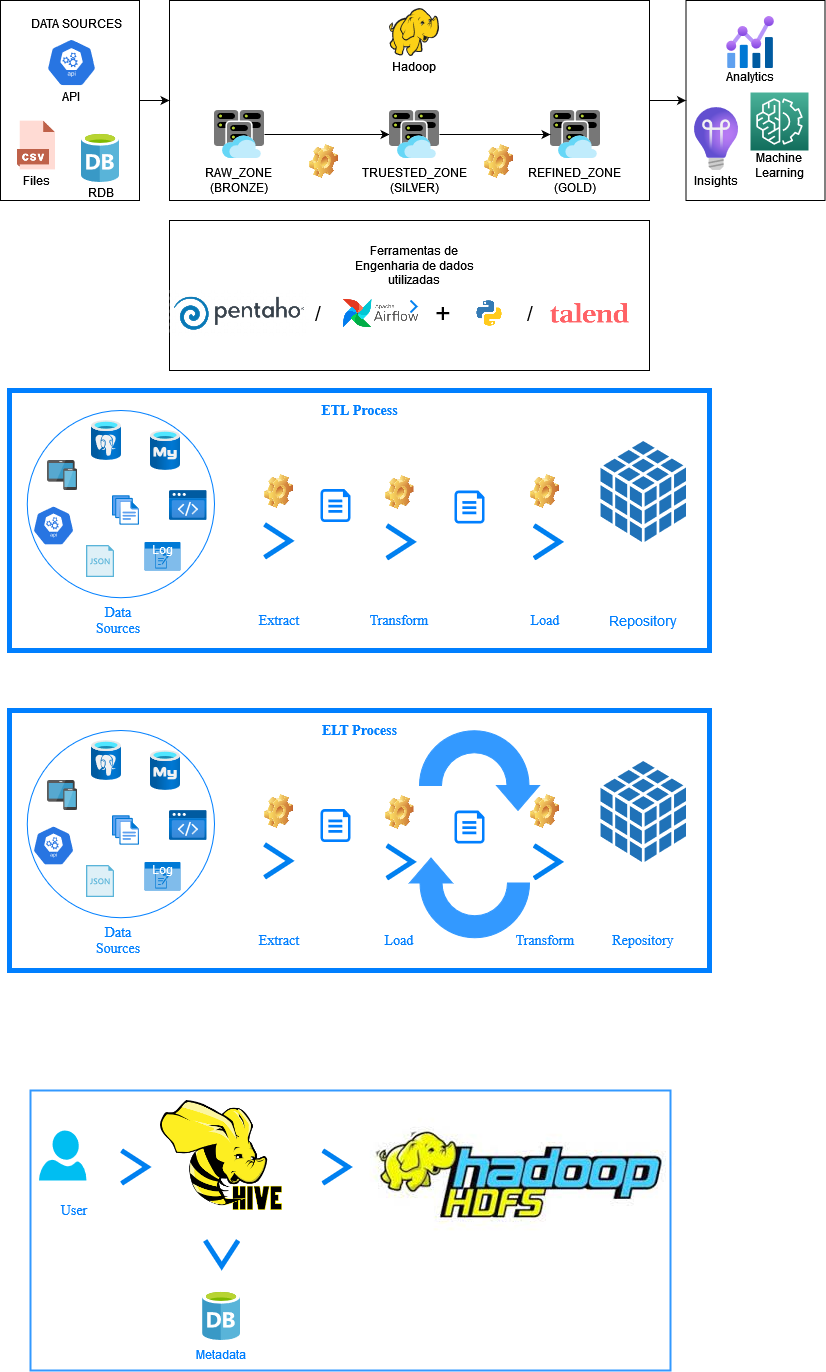
Por fim, o sistema de armazenamento de arquivos *Hadoop Distributed File System*, HDFS, integrado ao framework, é distribuído, altamente tolerante a falhas e aplicado sobre máquinas e rede de baixo custo geral. Um arquivo é distribuído em blocos de 64 MB, nos sistemas convencionais o bloco é de 512 bytes. Os blocos são replicados em diferentes *clusters*, assim, se um deles apresentar falhas, outro é utilizado, garantindo a integridade. Possui o mesmo tipo de transparência que um sistema convencional, o usuário não identifica diferenças na utilização. Possui algumas estruturas de controle exclusivas como: segurança no tráfego de informações, tolerância a falhas de nós, integridade no arquivo, consistência para os usuários acessarem o mesmo arquivo e o desempenho geral é similar aos convencionais.

O sistema HDFS em si não é relacional e portanto pode receber arquivos com qualquer tipo de extensão. Porém, como trabalharemos em um modelo de armazenamento relacional, faz se necessário a utilização de uma interface que compreenda o banco de dados convencional, assim como comandos em linguagem SQL, que é amplamente utilizada nesses tipos de bancos. Para isso, empregaremos o Apache Hive.

### 3.5. Hive

O Apache Hive é uma ferramenta open source presente dentro do ecossistema Hadoop e mantida pela Apache Fundation, que intenciona facilitar o trabalho com SQL e, portanto, com bancos de dados relacionais como o MySQL, MariaDB, Oracle entre tantos outros, mas contendo por “debaixo dos panos” o Hadoop e sua estrutura.. O SQL utilizado pelo Hive, assim como em outros bancos proprietário ou open source, é uma versão adaptada do original. No caso do Hive, o SQL utilizado foi batizado como HQL, Hive Query Language.

O Hive foi desenvolvido para atuar como um DW tendo no HDFS, do Hadoop, o sistema de armazenamento em execução. Com ele é possível criar grandes estruturas de dados que tem na clusterização, distribuição em grupos, sua base, mas que ainda assim é transparente para o desenvolvedor ou usuário.

Figura 7: Arquitetura Hive

O Hive realiza a interface entre o usuário, recebendo uma query em formato HQL por exemplo, e o Hadoop, traduz a query para um job map e reduce, figura 7. O Hive mantém as estruturas das tabelas e demais dependências em metadados dentro de um banco relacional, possibilitando preservar as construções como em outros DW. Por fim, ele também provê uma conexão jdbc, proporcionando que seja acessado e trabalhado como outros bancos existentes no mercado.

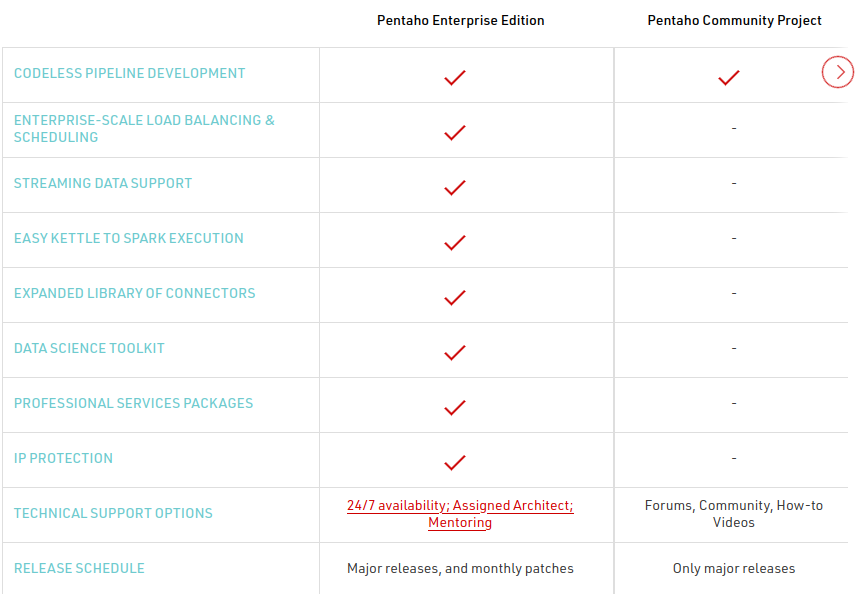
### 3.6. Pentaho Data Integration

Nascido de uma suíte para inteligência empresarial com ferramentas que propõe soluções desde a extração dos dados até a disponibilização, análise, e criação de *dashboards.* O Pentaho Data Integration, ou Kettle, é uma das ferramentas mais conhecidas para integração de dados. Contém versões proprietária e *Open Source* (figura 8).

Vem sendo desenvolvida desde 2004 pela própria Pentaho Corporation, sendo que em 2015 ela foi adquirida pela Hitachi Data Systems e em 2017 se tornou a Hitachi Vantara, enquanto marca, mas ainda é reconhecida pelo seu nome inicial.

É desenvolvido em linguagem java e apresenta quatro componentes principais: Spoon, Pan e Kitchen.

O Spoon é uma ferramenta de modelagem gráfica que permite ao usuário usar o *drag* e *drop*, arrastar e soltar, e é onde se define entradas, transformações e saída de dados. O Pan é uma aplicação por linha de comando que executa as transformações criadas na interface Spoon. O Kitchen, por sua vez, executa os jobs partindo de um sistema de arquivos ou banco de dados.

Figura 8: Versões do Pentaho. Fonte: Hitachi Vantara

Há de se pontuar que ao falarmos em jobs e transformações, no *Pentaho Data Integration,* estamos nos referindo a uma sequência de operações que são realizadas em sequência e completas e assim a automatização de uma dada tarefa, enquanto que a transformação é uma operação sobre os dados em si: leitura, seleção de campos, criação de tabela, agregação e assim por diante.

Pessotti da Rocha Magdaleno (2015) faz uma análise das características da ferramenta completa de BI em sua versão *community*, *open source*, concluindo que ela obteve resultados positivos, o que nos trás tranquilidade na sua utilização em nosso experimento de desenvolvimento de pipeline de dados.

Por sua vez, Lira Filho (2013), realiza uma comparação com outra ferramenta que também tem uma versão *open source*, o Talend Open Studio. Ele conclui que o Pentaho possui maior flexibilidade e velocidade em determinados cenários, mesmo deixando a desejar no quesito hardware.

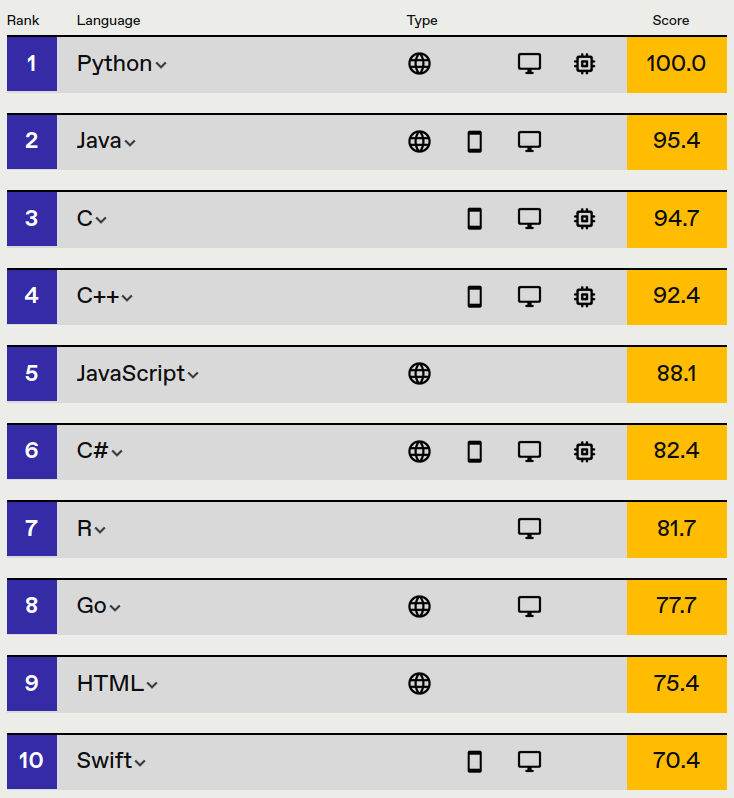
### 3.7. Python e Airflow

### 3.7.1. Python

Criada por Guido van Rossum em idos de 1991, a linguagem python hoje ocupa o primeiro lugar no ranking do Instituto de Engenheiros Eletricistas e Eletrônicos ou Instituto de Engenheiros Eletrotécnicos e Eletrônicos dos Estados Unidos, ou IEEE (figura 9). Apoiada sobre a premissa de ter uma curva de aprendizado rápida e uma sintaxe mais enxuta e simples, avançou em passos largos dentro do campo dos dados, sendo adotada por cientistas de dados, engenheiros e analistas.

Python é uma linguagem *open source* de propósito geral, ou seja, com ela é possível criar aplicações web, programas de computador e sistemas dos mais variados. Seus módulos, bibliotecas, estendem ainda mais seu poder de ação.

A comunidade é um ponto forte na sua adoção. Ela é responsável, não só pela divulgação, mas implementam e desenvolvem, não só, os módulos que a compõem e a própria linguagem em si.

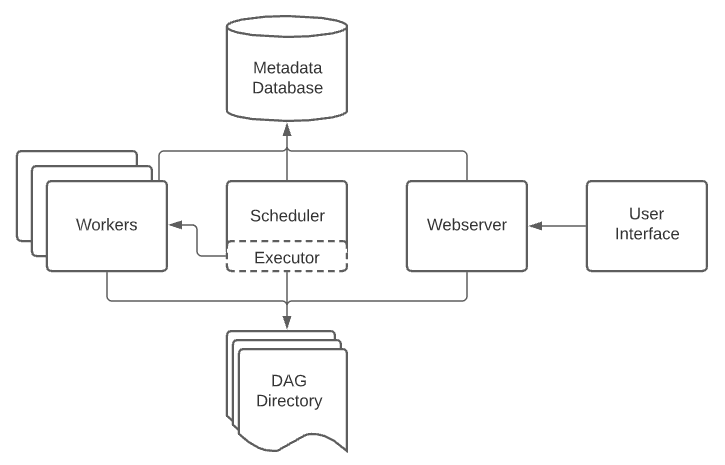
Figura 9: 10 Linguagens de programação mais utilizadas 2021. Fonte IEEE

Atualmente é facilmente observável que usuários não convencionais de linguagem de programação estão se dedicando a aprender e aplicá-la em suas atividades cotidianas, tais como: cientistas sociais, psicólogos, biólogos, geógrafos, jornalistas e muitos outros.

### 3.7.2. Airflow

Desenvolvido inicialmente dentro do Airbnb, 2014, para gerenciar o complexo fluxo de dados da empresa, foi posteriormente incorporada à Apache Fundation, 2016. Como software livre, se tornou um dos projetos mais utilizados da organização.

Atualmente, está em sua versão 2.0.2. Foi desenvolvido em python e utiliza scripts da linguagem na criação dos *workflows*, fluxo de trabalho. Tem uma interface web por onde o usuário pode gerenciar as atividades que estão sendo realizadas.

Figura 10: Arquitetura do Airflow. Fonte: Apache Airflow

Seguindo sua arquitetura, figura 10, apontamos os seguintes componentes essenciais:

DAG, *Directed Acyclic Graph*: coleta e organiza a série de tarefas. Suas dependências e relacionamentos são observados, gerenciando qual deverá executar.

*Webserver*: propicia uma interface visual ao usuário. Através dela, além de acompanhar o andamento das tarefas, entre outras coisas, podemos observar os logs com a execução do processo, facilitando a resolução de problemas, quando eles ocorrem.

*Scheduler*: monitora a ação das tarefas, DAGs e realiza a sincronização das mesmas. A cada minuto verifica no diretório de DAGs se há alteração no código de alguma e se precisa preparar para execução de alguma que esteja ativa.

*Metadata Database*: é onde ficam armazenados os dados relacionados a cada instância de tarefa de uma DAG. Hora da execução, situação de cada tarefa, tempo e agendamento são armazenados aqui.

*Executors*: são a forma pelo qual as instâncias das tarefas são executadas.

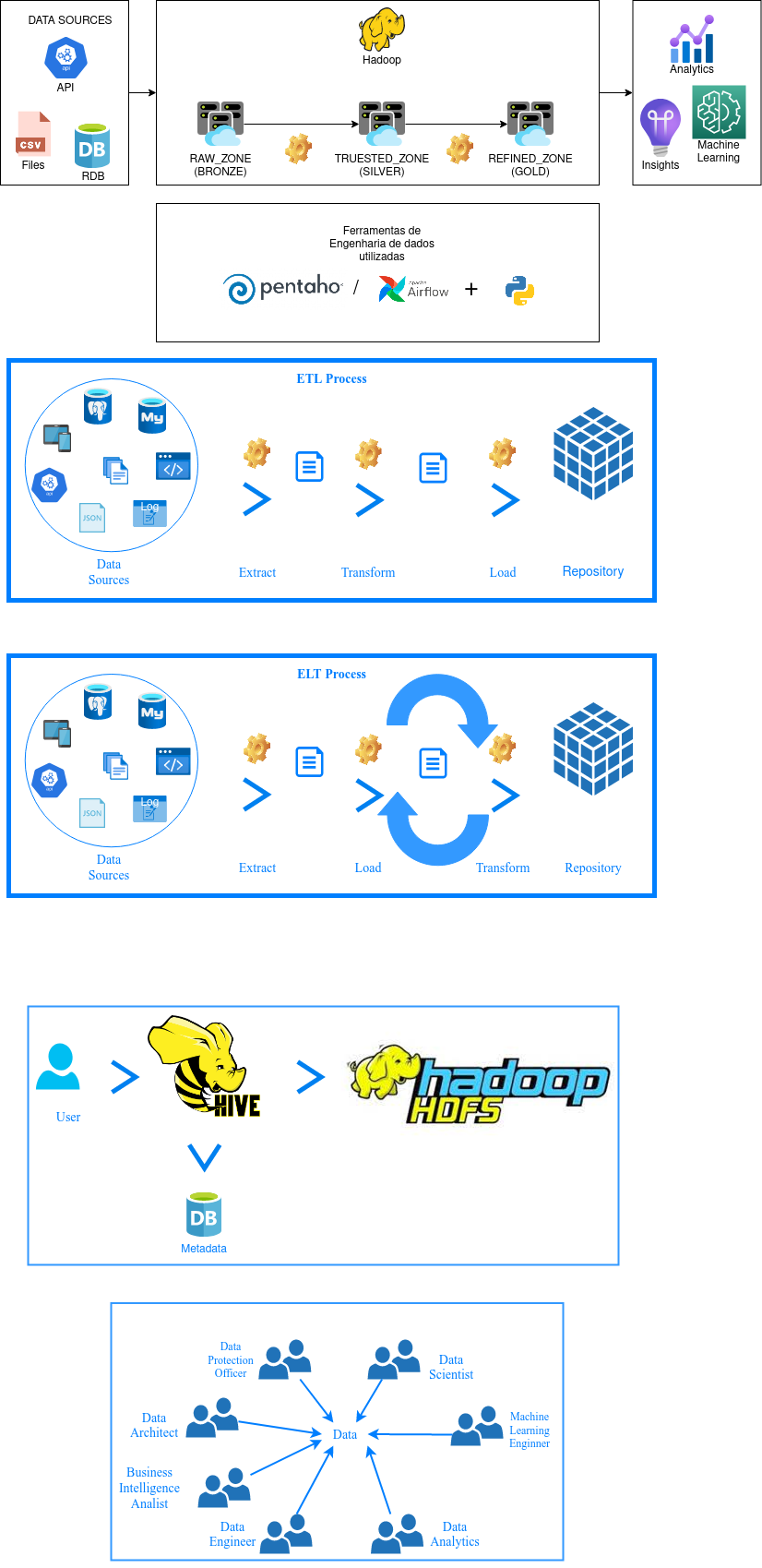
*Workers*: são responsáveis pela execução em si de uma tarefa.

# 4 Metodologia

Tendo como ponto de partida três fontes de dados distintas: um arquivo em formato csv, uma api com autenticação e uma tabela armazenada em banco de dados relacional, buscará criar um pipeline de dados de ELT.

O processo que irá ser realizado pode ser observado na arquitetura exemplificada na figura 11 abaixo.

O laboratório irá abarcar as seguintes etapas:

Figura 11: Arquitetura geral do pipeline

1. Importação e armazenamento dos dados em camada sem tratamento, bronze, dentro do Hadoop através de conexão com o Hive, no formato texto.
2. Tratamento básico de dados e arquivamento em camada pronta, silver, com tabelas particionadas e comprimidas em formato parquet.
3. Agrupamento e enriquecimento de dados com posterior armazenamento em camada disponibilizada para usuários, gold.

Serão observados os seguintes pontos relacionados as ferramentas para a criação do pipeline: instalação, documentação oficial, aprendizagem, execução do desenvolvimento.

Destaca-se que a arquitetura empregada se baseia no formato de camadas de data lake, porém a implementação será em ambiente de data warehouse com o Apache Hive, já que são dados oriundos e mantidos em tabelas do tipo relacionais.

# 5 DESENVOLVIMENTO

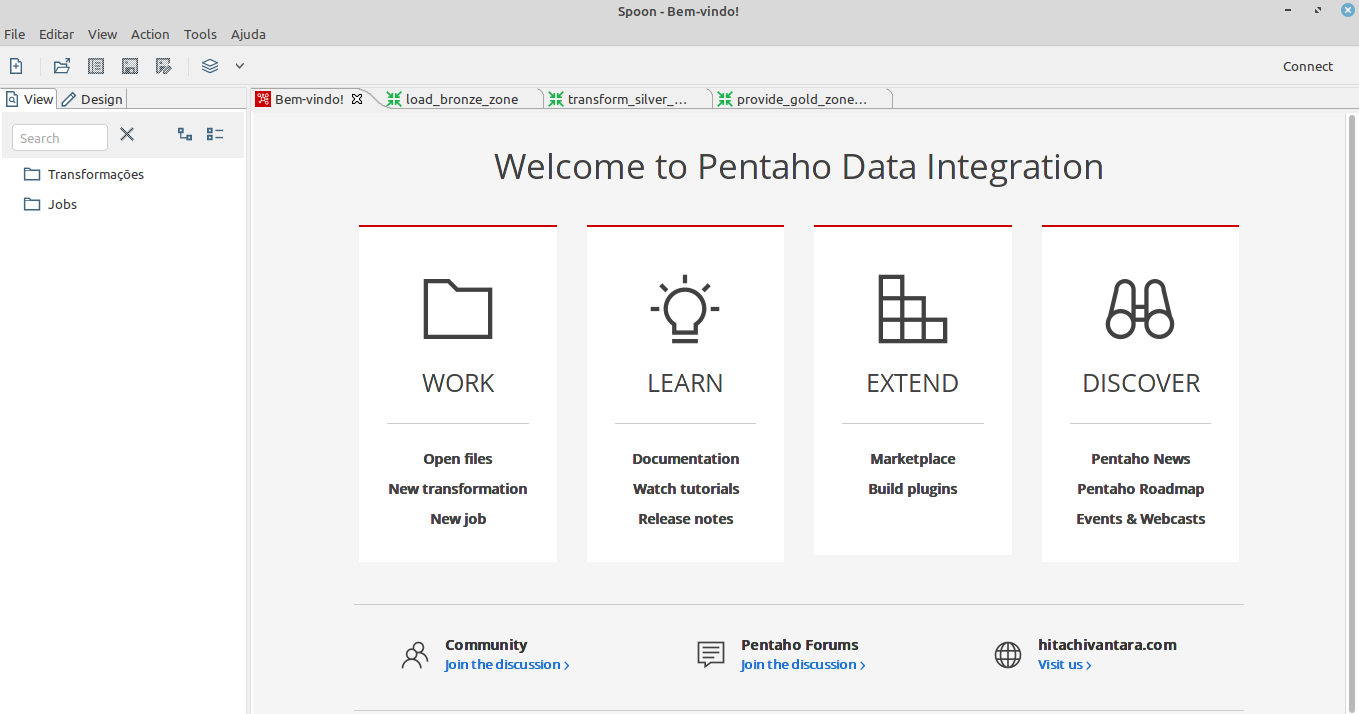
Algumas observações devem ser pontuadas com relação ao laboratório realizado. Os dados utilizados são retirados de portais abertos. Para o nosso experimento foram utilizados dados da campanha de vacinação contra a COVID 19, acessados por API. O arquivo em formato csv continha uma base com os dados de latitude e longitude de todas as cidades brasileiras. Por fim, foi utilizado um banco de dados relacional, MariaDB, onde foi previamente populado com informações de escolas e aulas.

Os softwares utilizados podem ser observados na tabela 1 enquanto um resumo do hardware pode ser visto na tabela 2.

Tabela 1: Softwares utilizados

Tabela 2: Hardware utilizado

### 5.1 Pentaho

Figura 12: Pentaho - tela inicial

### 5.1.1 Instalação

Com relação à instalação o Pentaho se mostra de fácil execução. Não contém um instalador específico para cada sistema operacional, mas executáveis em geral, e têm como dependência o java sdk 8. O arquivo de download é grande, na versão 9.2.0.0, lançada em 02 de agosto de 2021, contém 1,9 GB compactado.

Figura 13: Pentaho - Arquivos de download

Dentro dele contém os arquivos necessários para execução em ambiente Linux, Windows e Mac.

Para a utilização do programa é necessário apenas descompactar, ter certeza que o java está configurado nas variáveis de ambiente e executar o arquivo relacionado a cada sistema operacional.

### 5.1.2 Documentação Oficial

Em questão de documentação, o Pentaho mostra ter os principais pontos necessários em geral: instalação, primeiros passos e de consulta. Porém, o que ganha ênfase, foi o quanto há apoio por outros grupos. Blogs diretamente apontados no site oficial do serviço, vídeos, fóruns.

Destaque para os vídeos da comunidade hospedados no You Tube. Enquanto on-line encontramos a documentação apenas em Inglês, há uma farta gama de facilitadores que dispõe seus vídeos em português.

### 5.1.3 Aprendizagem

Nesse quesito ele se mostra bastante prático. Utilizando muito a ideia de arrastar, soltar e ligar. O engenheiro deve conhecer os dados e sua estrutura, mas para a utilização dos componentes não é necessário uma grande bagagem técnica. Em geral SQL pode ser bastante útil para algumas atividades com dados.

Se houver necessidade de utilizar outras linguagens, é possível realizar com instalando um plugin, sendo então disponível para Python, Php entre outras.

A comunidade é o grande diferencial. Contribuindo com inúmeras dicas, tutoriais e artigos que ajudam o usuário iniciante a conseguir montar seus primeiros jobs e cadeias de transformação.

### 5.1.4 Execução

A integração de dados do Pentaho permite que o usuário realize diversas atividades rotineiras de engenharia de dados. A ferramenta possui conexão com inúmeros bancos de dados. Embora ela se destaque com bancos relacionais, não relacionais também podem ser trabalhados com a ferramenta. Há integração com serviços distribuídos. Hadoop é suportado diretamente.

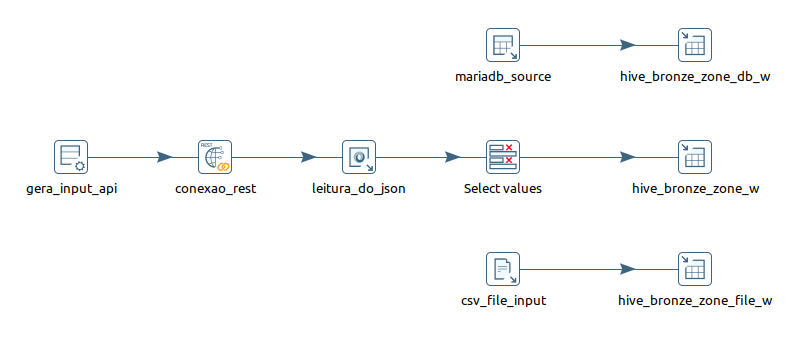
Em caso de falha em algum ponto do pipeline ou que fontes não estejam disponíveis, uma mensagem de e-mail pode ser configurada para gerar um alerta. Assim uma rápida reação ao problema ocorrido pode ser posta em prática. Por outro lado, possibilita também que informes de sucesso sejam enviados a um destinatário.

Outro ponto que vale pontuar, é a possibilidade de agendar uma atividade para ser realizada. Deixando a automação da execução aplicável para quando se desejar. Um arquivo que é disponibilizado mensalmente poderá então ser recuperado sem chamadas desnecessárias.

### 5.1.4.1 Camada bronze

As três fontes de dados foram bem atendidas com relação à primeira etapa de importação.

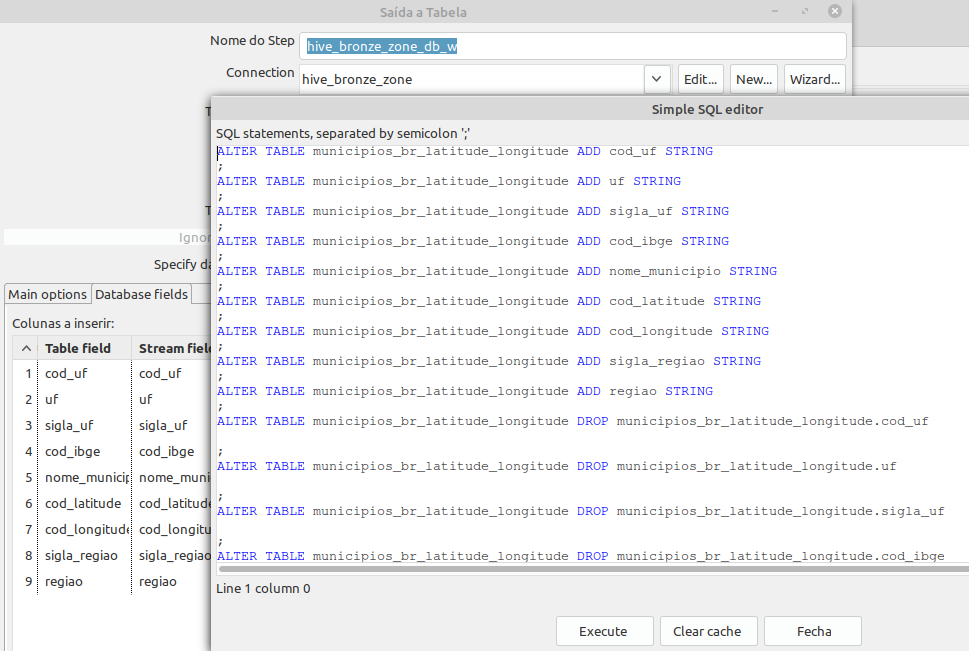
Na figura 14 temos na primeira linha a migração de dados de origem de banco de dados relacional, na segunda com dados de API e na terceira linha a origem dos dados é do arquivo em csv

Figura 14: Migração de dados na camada bronze

No primeiro cenário, com fonte advinda de arquivo, o Pentaho entendeu rapidamente o formato dos dados do arquivo csv, a edição dos formatos também foi facilitada, por fim disponibiliza uma sugestão em SQL para a criação da tabela. Assim, a população da tabela foi rápida e sem impedimentos técnicos.

Com relação ao cenário com dados oriundos de acesso via API, ele tem em seu arcabouço as ferramentas necessárias para acessar a endereço de acesso para conexão, realizar o login e acessar os dados em formato json. Porém, cabe observar, que a interpretação dos dados não eram feitas instantaneamente, mas foi necessário a utilização de um amostral do arquivo de dados fonte para que ele pudesse realizar a análise da estrutura e assim a interpretação da base acessada através da API.

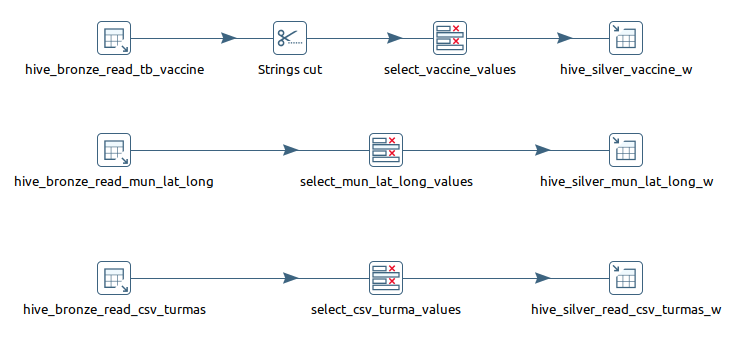
No cenário de importação com dados originados de banco de dados, a migração foi um pouco lenta, porém aceitável dentro da proposta. Há diversos conectores disponíveis, assim, não foi um problema realizar a conexão com o banco de dados utilizado para teste, MariaDB. E da mesma forma que em outros cenários, criar a tabela era facilitado com sugestão em sql, figura 15.

Figura 15: Sugestão do Pentaho para a criação de tabela

### 5.1.4.2. Camada silver

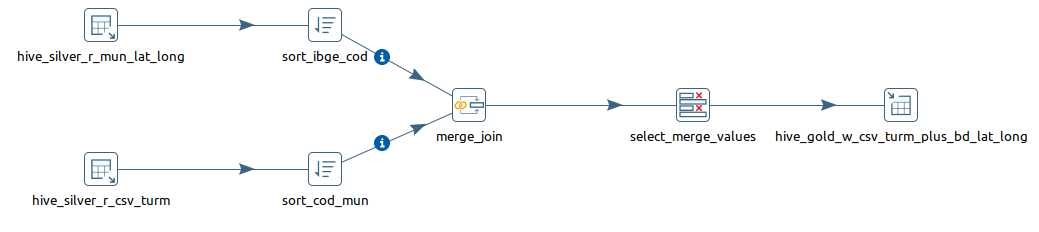
Adentrando a camada silver o laboratório passou a realizar conexão apenas com o Hadoop em seu modo pseudo distribuído, assim não houve um acréscimo no quesito tempo entre as diferentes tarefas, mas todas ocorriam em tempo de execução, figura 16.

O Pentaho Data Integration permite que os arquivos que forem popular as tabelas em formato parquet sejam mantidas, ainda viabilizam a inserção sobre escrita de modo direto, sem ações indiretas.

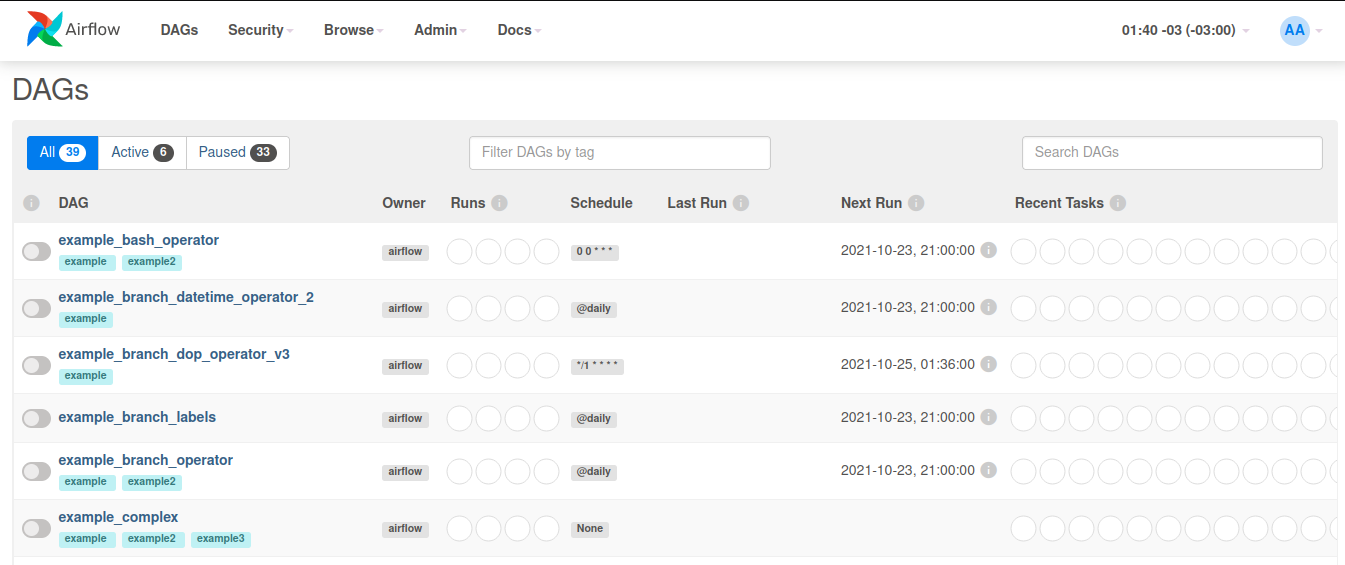
Figura 16: Camada silver

### 5.1.4.3. Camada gold

Nessa etapa foi realizada a ação de juntar, *join*, duas tabelas diferentes através de um identificador comum. Fizemos a junção de dados escolares com as localizações geográficas, figura 17. No Pentaho, após a junção dos dados, eles ficam disponíveis e novamente podemos escolher quais serão armazenados na tabela final. Assim como, com base neles, podemos criar a tabela que será utilizada para a população dos dados.

Figura 17: Camada gold

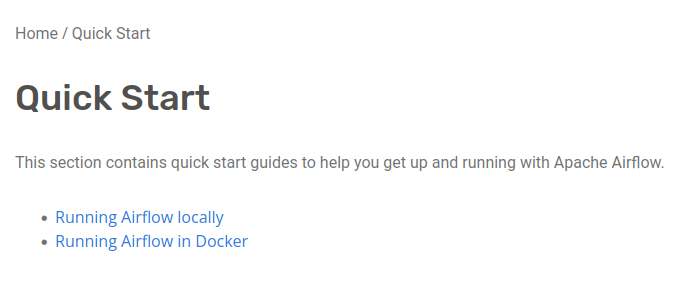
### 5.2 Airflow e Python

Figura 18: Airflow tela inicial

### 5.2.1 Instalação

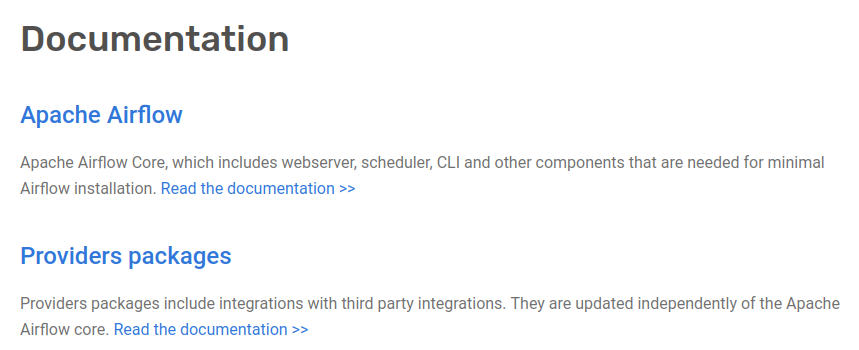
Em questão de instalação o Airflow oferece duas formas principais. Uma delas através de Docker, criando uma virtualização com a imagem, e outra localmente com o Python, pela instalação de módulo.

Ambas as opções são muito bem documentadas. Realizamos a instalação das duas formas disponíveis e seguindo o passo a passo do site o resultado foi positivo.

Figura 19: Tipos de instalação do Airflow

### 5.2.2 Documentação oficial

Nesse quesito o software é muito bem documentado. Tendo resposta ou apontando soluções para a maioria das questões. Seja de instalação, conceitos, componentes ou exemplos de como fazer ou utilizar as opções disponíveis.

Figura 20: Página de documentação - detalhe

### 5.2.3 Aprendizagem

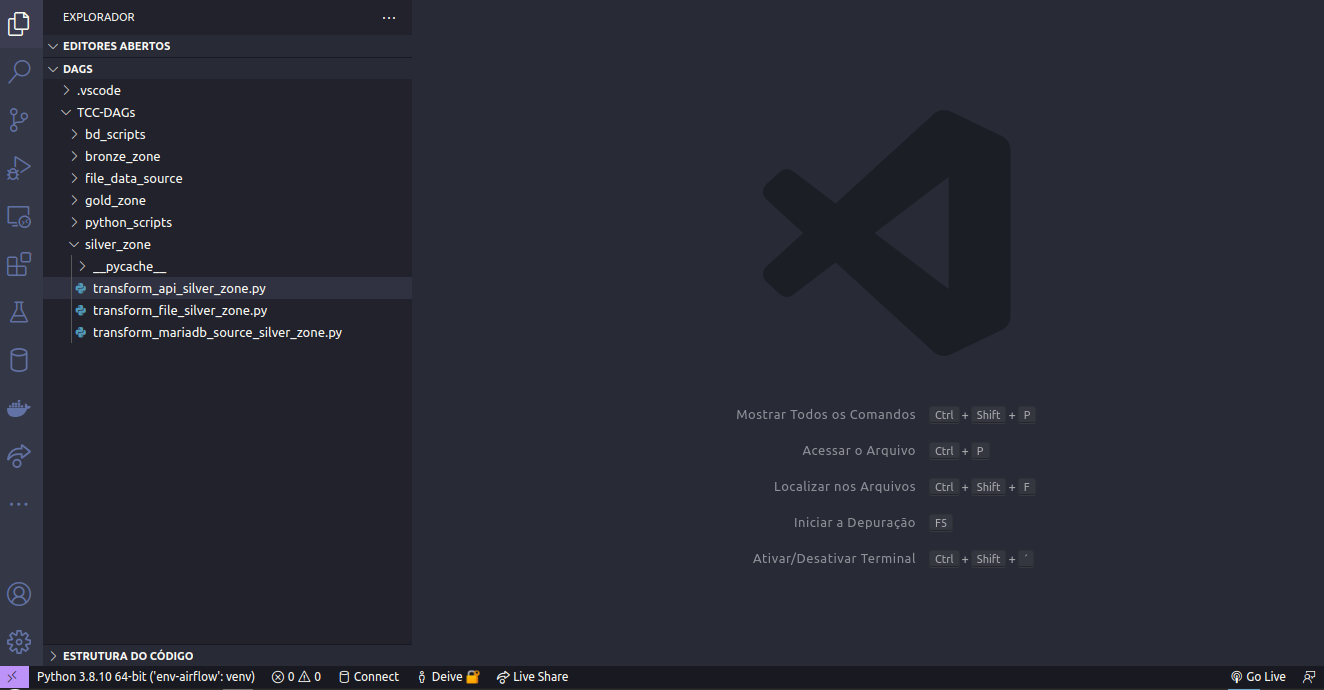
Como uma ferramenta voltada para gerenciamento de fluxo de de dados, a aprendizagem também é direcionada para esse público. Embora seja bem fácil encontrar tutoriais para os mais diferentes públicos, o entendimento é que o público principal é de desenvolvedores.

Observa-se que as DAGs são criadas por script Python, ou seja, o conhecimento prévio da linguagem é necessário para a execução apropriada das etapas. A interface web disponibilizada é para acompanhamento, gerenciamento e atividades relacionadas, não sendo para o desenvolvimento.

### 5.2.4 Execução

A integração de dados é feita pelo engenheiro de dados. Todas as etapas são pensadas e modeladas segundo as regras que chegam até o desenvolvedor. Na realização das tarefas foram utilizados o python puro e módulo como o request, para chamadas API, o pandas, para manipulação e criação de dados tabulares e tentou-se utilizar boas práticas de programação. Assim, os códigos foram estruturados em funções que executam apenas uma parte específica de cada etapa. Assim, há funções que direcionadas ao download de dados, outras que tratam e selecionam. Por seu turno, as tabelas foram desenvolvidas em SQL e HQL. O editor de código escolhido foi o Visual Studio Code da Microsoft, figura 21, que permite tanto o desenvolvimento em python puro quanto testes em jupyter notebook.

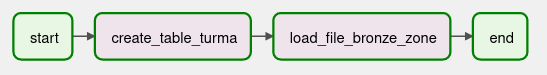
A relação entre as tarefas é dada pelo uso do sinal duplo de menor ou maior. Assim, a etapa 1 seguida pela etapa 2 é apontada da seguinte forma: etapa 1 >> etapa 2.

Figura 21: Visual Studio Code

### 5.2.4.1 Camada bronze

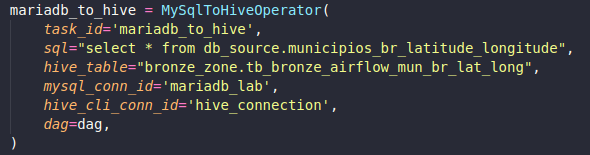
O arquivo csv foi previamente carregado no jupyter para verificação de integridade e desenvolvimento das etapas pelo qual deveria passar. O Hive permite que os dados de um arquivo sejam carregados diretamente em uma tabela. Essa foi a abordagem seguida. Primeiro criamos a tabela, depois realizamos inserção dos dados diretamente do arquivo.

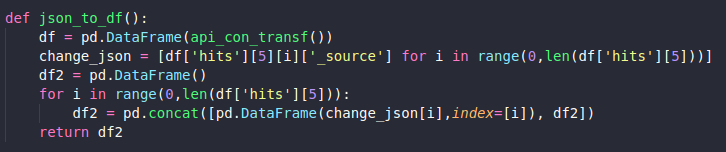
Sendo uma etapa temporária, já vinculamos a exclusão da tabela, na DAG da camada silver, e sua contínua criação, garantindo que os dados não serão duplicados e o sistema sobrecarregado com dados desnecessários.

Figura 22: Etapas em uma DAG de carregamento de arquivo csv

No caso da migração da base de dados armazenado no MariaDB, foi uma etapa facilitada pelo Ariflow, visto que após a instalação do pacote para trabalhar com bancos MySQL e consequentemente MariaDB, entre os componentes disponíveis há um especializado para esse tipo de processo. Bastando realizar o preenchimento com as informações de origem e destino, figura 23. Vale ressaltar que também já havia sido modelado a tabela que receberia esses dados.

Por último, o carregamento dos dados com origem em API foram feitas em quatro tarefas diferentes. Na primeira era realizado o download dos dados e a transformação dos mesmos em formato tabular, figura 24. Depois o mesmo arquivo era recarregado e então uma etapa de escolha de campos era realizada e um novo arquivo tabular gerado. Por fim, a tabela era criada no Hive e a inserção dos dados feira da mesma forma como no carregamento de arquivo csv já apresentado.

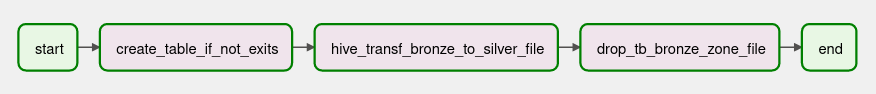
Figura 23: Detalhe da tarefa de migração

Figura 24: Detalhe da conversão de json para dataframe

### 5.2.4.2 Camada silver

Após a ingestão de dados da camada bronze, o trabalho com os dados passou a ser realizado apenas através de scripts HQL.

Em geral consistindo na criação da tabela que será populada no Hadoop pelo Hive, na transição de dados de uma camada para a outra e por fim na exclusão da tabela que ficava na camada bronze, figura 25.

Figura 25: Transição para silver - detalhe

### 5.2.4.3 Camada gold

Na camada gold a realização da junção das tabelas deu-se por um único script que já agregava o necessário. Não sendo necessário tratamentos pormenorizados dos dados nem um enriquecimento mais específico.

Figura 26: Junção das tabelas de educação e localização geográfica

# 6 CONSIDERAÇÕES FINAIS

Após a realização do desenvolvimento de pipeline com ambas as ferramentas alguns pontos ganham destaque em ambos os processos de criação.

Com relação ao Pentaho ela demonstra ser bem mais amigável para a execução das tarefas por quem não tem um relacionamento direto com a manipulação de dados. Sua interface drag e drop possibilita testes em tempo de desenvolvimento.

A criação de tabelas e manipulação de dados também é facilitado, tendo como premissa apenas o conhecimento dos tipos que irão ser usados e qual o resultado final pretendido. Encontrar tutoriais e modos de aprendizagem também não é um problema, sendo facilmente encontrados na internet. A instalação pode ser um pouco problemática se não houver conhecimento de variáveis de ambiente ou se o hardware for inadequado.

O Airflow, por sua vez, vem em ascensão, sendo cada vez mais adotado. Porém ele não é direcionado à iniciantes. O que não significa que em pouco tempo o usuário não possa estar produtivo, pelo contrário, sua base Python e interface intuitiva facilita muito a adoção. Outro ponto que merece destaque é a farta documentação disponível. Nela encontramos todo o necessário para iniciar o trabalho com a ferramenta e até como desenvolver componentes próprios.

Para alguém que já é desenvolvedor dá um grande domínio sobre os dados e sobre os diferentes aspectos que pode tomar. O log de erros fornece informações suficientes para a resolução de problemas e é muito importante no debug do código desenvolvido.

Por fim, observa-se que são ferramentas bem diferentes, mas que realizam bem as ações que se propõem a fazer e que podem até mesmo fazer etapas diferentes do desenvolvimento ganhando tempo acrescentando ao resultado final. Dessa forma, etapas que podem ser realizadas por um usuário comum pode ser realizado no Pentaho, enquanto manipulações mais específicas podem orquestradas com o Airflow.

# REFERÊNCIAS

APACHE AIRFLOW. Apache Airflow Documentation. Disponível em: https://airflow.apache.org/docs/apache-airflow/stable/index.html. Acesso em: 23 out. 2021.

COSTA, Marcelo . Pentaho Data Integration - ETL em Software Livre. 2017. Disponível em: https://www.infoq.com/br/articles/pentaho-pdi/. Acesso em: 01 set. 2021.

EL-SAPPAGH, Shaker H. Ali; HENDAWI, Abdeltawab M. Ahmed; BASTAWISSY, Ali Hamed El. **A proposed model for data warehouse ETL processes**. Journal Of King Saud University - Computer And Information Sciences, [S.L.], v. 23, n. 2, p. 91-104, jul. 2011. Elsevier BV. Disponível em: https://www.sciencedirect.com/science/article/pii/S131915781100019X. Acesso em: 27 de maio de 2021.

GARCIA, Marco. **Data Enginner ou Engenheiro de Dados - Conheça mais sobre**. Disponível em: https://www.cetax.com.br/blog/data-engineer-ou-engenheiro-de-dados/. Acesso em: 24 de maio de 2021.

GOUVEIA, Roberta Macêdo Marques; FREITAS, Charles Nicollas Cavalcante. **Implementação de um Data Warehouse para Análise de Dados Abertos Governamentais da Educação à Distância**. Revista de Educação Ciência e Tecnologia, Canoas, v. 7, n. 2, 2018. Disponível em: https://periodicos.ifrs.edu.br/index.php/tear/article/view/3037/2112. Acesso em: 25 maio 2021.

HARENSLAK, Bas; RUITER, Julian de. Data Pipelines with Apache Airflow. Shelter Island, Ny: Manning Publications Co., 2021.

HITACHI VANTARA. 2021. Disponível em: https://community.hitachivantara.com/s/. Acesso em: 01 set. 2021.

IEEE. Top Programming Languages 2021. 2021. Disponível em: https://spectrum.ieee.org/top-programming-languages/. Acesso em: 20 out. 2021.

LIRA FILHO, Hermanny Alexandre dos Santos. **Análise comparativa das Ferramentas de ETL – Kettle e Talend**. 2013. 79 f. TCC (Graduação) - Curso de Bacharelado em Sistemas de Informação, Universidade Federal da Paraíba, Rio Tinto, 2013. Disponível em: https://repositorio.ufpb.br/jspui/handle/123456789/17132?locale=pt\_BR. Acesso em: 29 maio de 2021.

MICROSOFT. **Visão geral dos cubos OLAP do Service Manager para análise avançada**. 2018. Disponível em: https://docs.microsoft.com/pt-br/system-center/scsm/olap-cubes-overview?view=sc-sm-2019. Acesso em: 30 maio de 2021.

PESSOTTI DA ROCHA MAGDALENO, Rafael. Analisando o Pentaho como Ferramenta de BI. 2015. 55 f. TCC (Graduação) - Curso de Bacharelado em Sistemas de Informação, Centro Universitário Eurípides de Marília, Marília, 2015. Disponível em: https://aberto.univem.edu.br/bitstream/handle/11077/1383/Analisando%20o%20Pentaho%20como%20ferramenta%20de%20BI.pdf?sequence=1&isAllowed=y. Acesso em: 15 set. 2021.

PYTHON.ORG. Site Oficial. Disponível em: https://python.org.br/. Acesso em: 12 out. 2021

RAMPAZO, Paolo. Apache Airflow: conceitos iniciais. 2021. Disponível em: https://medium.com/datarisk-io/apache-airflow-conceitos-iniciais-e09c0dd18141. Acesso em: 23 out. 2021.

RASLAN, Daniela Andrade; CALAZANS, Angélica Toffano Seidel. **Data Warehouse: conceitos e aplicações**. Universitas: Gestão e TI, Brasília, ago. 2014. Centro de Ensino Unificado de Brasília. Disponível em: https://www.researchgate.net/publication/287517542. Acesso em: 25 de maio 2021.

SAS. **ETL: o que é e qual sua importância?**. [20--]. Disponível em: https://www.sas.com/pt\_br/insights/data-management/o-que-e-etl.html. Acesso em: 30 maio 2021.

SENE, Allan. **O que faz um Engenheiro de Dados?**. 2018. Disponível em: https://medium.com/data-hackers/o-que-faz-um-engenheiro-de-dados-fdcb0bca966b. Acesso em: 25 de maio 2021.

TAMIR, Mike; MILLER, Steven; GAGLIARDI, Alessandro. **The Data Engineer**. Ssrn Electronic Journal, [S.L.], 2015. Elsevier BV. Disponível em: https://papers.ssrn.com/sol3/papers.cfm?abstract\_id=2762013. Acesso em: 29 maio 2021.

WHITE, Sarah K.. **O que é e o que faz um engenheiro de dados?**.2018. Disponível em: https://cio.com.br/tendencias/o-que-e-e-o-que-faz-um-engenheiro-de-dados/. Acesso em: 24 de maio 2021.

