NOVA IMS Information Management School

MGI

Mestrado em

Gestão de Informação

Implementação de uma solução de Business Intelligence em Cloud

Processos de Extração e Transformação de dados

Vasco Martins Fogaça Infante Bento

Relatório de Estágio

apresentada(o) como requisito parcial para obtenção do grau de Mestre em Gestão de Informação

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

IMPLEMENTAÇÃO DE UMA SOLUÇÃO DE BUSINESS INTELLIGENCE EM CLOUD - PROCESSOS DE EXTRAÇÃO E TRANSFORMAÇÃO DE DADOS

por

Vasco Bento

Dissertação / Projeto apresentada(o) como requisito parcial para obtenção do grau de Mestre em Gestão de Informação, com especialização em Gestão do Conhecimento e Business Intelligence

Orientador: Prof. João Bruno Morais de Sousa Jardim

Fevereiro 2023

DECLARAÇÃO DE INTEGRIDADE

Declaro ter realizado o presente trabalho académico com integridade. Confirmo que não recorri à prática de plágio ou de qualquer outra forma de utilização indevida de informação ou de falsificação de resultados durante o processo de elaboração deste trabalho. Declaro ainda que tenho conhecimento das Regras de Conduta e do Código de Honra da NOVA Information Management School.

Vasco Bento

Lisboa, 28 fevereiro de 2023

RESUMO

Num mundo em que as tecnologias estão em constante evolução e os mercados cada vez mais competitivos, as organizações necessitam de estar em constante alerta para compreender as tecnologias de ponta e as ferramentas que ajudarão a superar qualquer competição que surja. Plataformas de dados modernas que incorporam tecnologias de *cloud* apoiam as organizações a ficar à frente da concorrência, fornecendo soluções que as ajudam a capturar e utilizar dados inexplorados de forma otimizada e armazenamentos escaláveis, para se adaptem a quantidades de dados cada vez maiores. Além disso, adotar ferramentas de processamento e visualização de dados que ajudem a melhorar o processo de tomada de decisão. Com a quantidade de plataformas de *cloud* disponíveis no mercado, das mais pequenas até às grandes corporações, existe uma vasta oferta com muita flexibilidade para as organizações escolherem qual a melhor tecnologia de *cloud* que se alinha com os seus casos de uso e estratégia geral de produtos e serviços. Este relatório de estágio está focado no desenvolvimento de uma solução de Business Intelligence em *cloud* para um cliente que opera no setor da saúde. Para tal, será utilizada a plataforma Microsoft Azure e o Databricks para o desenvolvimento dos processos de extração e transformação de dados.

PALAVRAS-CHAVE

Big Data; Business Intelligence; Computação em Nuvem; ETL

ABSTRACT

In a world where technologies are constantly evolving and markets are increasingly competitive, organizations need to be constantly alert to understand cutting-edge technologies and tools that will help them overcome any competition that arises. Modern data platforms that incorporate cloud technologies support organizations in staying ahead of the competition by providing solutions that help them capture and utilize untapped data in an optimized way, and scalable storage to adapt to increasing amounts of data. Additionally, adopting data processing and visualization tools helps improve the decision-making process. With the number of cloud platforms available in the market, from small to large corporations, there is a vast range of flexibility for organizations to choose the best cloud technology that aligns with their use cases and overall product and service strategy. This internship report focuses on the development of a cloud-based Business Intelligence solution for a client operating in the healthcare sector. To this end, the Microsoft Azure platform and Databricks will be used for the development of data extraction and transformation processes.

KEYWORDS

Big Data; Business Intelligence; Cloud Computing; ETL

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

EDL	Enterprise Data Lake
HDFS	Hadoop Distributed File System
IoT	Internet of Things
BI	Business Intelligence
DSS	Decision Support System
OLAP	Online analytical processing
EDL	Extract, Transform and Load
PaaS	Platform as a service
IaaS	Infrastructure as a Service
SaaS	Software as a service
IT	Information Technology
ETL	Extract, Transform, Load
IBM	International Business Machines
CPM	Corporate Performance Management

1. INTRODUÇÃO

1.1. CONTEXTO GERAL

Os dados sempre desempenharam um papel crucial no impulsionamento dos negócios. No entanto, nas últimas décadas, testemunhamos uma mudança dramática na forma como os dados são gerados, armazenados, processados e analisados. A emergência de novas tecnologias e a crescente dependência de dados transformaram a maneira como os negócios são feitos, e essa evolução foi impulsionada pelo surgimento da computação em *cloud*, *big data e business intelligence*.

A ascensão da computação em nuvem revolucionou a forma como armazenamos e processamos dados. Com soluções baseadas na nuvem, as empresas podem facilmente armazenar e aceder aos seus dados de qualquer lugar do mundo, sem a necessidade de *hardware* local. Isso levou a um aumento da eficiência e escalabilidade, permitindo que as organizações se adaptem rapidamente às mudanças das condições de mercado e às necessidades dos clientes.

Este relatório de estágio irá focar-se no desenvolvimento de uma solução de business intelligence em *cloud*, mais especificamente nos processos de extração e transformação de dados. Esta solução foi desenvolvida utilizando as tecnologias de *cloud* da Microsoft assim como *notebooks* de Databricks. A solução final irá ser implementada em diversos países numa empresa multinacional que opera na àrea da saúde, de forma a melhorar os seus processos de dados assim como aumentar a capacidade de visualização e tomada de decisão.

Este projeto foi desenvolvido por uma equipa da BI4ALL, empresa onde atualmente trabalho, composta por cinco pessoas, um gestor de projeto, dois *business analysts* e dois desenvolvedores. O trabalho que desenvolvi baseia-se na construção de pipelines dinâmicos em *cloud* para extração de dados e construção de *notebooks* Databricks em Python responsáveis pela transformação dos dados.

1.2. CONTEXTO ORGANIZACIONAL

Este estágio irá ser conduzido pela empresa BI4ALL. A BI4ALL é uma empresa portuguesa de consultoria que opera nas áreas de *business intelligence* e *data analytics*. A empresa foi fundada em 2004 e está sediada em Lisboa; tem como propósito transformar dados em insights e a sua missão é ajudar organizações a serem mais ágeis, flexíveis e vigorosas; antecipar o que não é previsível e ajudar na rápida adaptação às mudanças de mercado, pois, as organizações ficam melhor preparadas para o futuro.

Considerada uma das *Top 10 Big Data Analytics Consulting/Service Companies in Europe 2020*, e uma das *10 Most Successful Consultant Companies to Watch in 2022*, a BI4ALL ajuda as organizações no processo de Transformação Digital e *Data Strategy* e conta com competências de excelência nas áreas:

- *Data Analytics*
- *Big Data*
- *Inteligência Artificial*
- *Data Science*
- CPM

- *Data Visualization*
- *Software Engineering*

A empresa possui uma vasta experiência respondendo a uma ampla gama de questões e requisitos de negócios de muitas indústrias. Auxilia as empresas a prosperar e a prepararem-se para os desafios do presente e do futuro. Alguns setores onde a BI4ALL atua como: Governo e Serviços Públicos, Alimentar, Retalho, Automóvel, Saúde, Bancários e Financeiros, Jurídico, entre outos.

2. REVISÃO DE LITERATURA

2.1. BUSINESS INTELLIGENCE

2.1.1. História

Business Inteligence (BI) tem uma grande história que se estende por várias décadas e sofreu uma significativa evolução ao longo do tempo. Nesta revisão de literatura, iremos explorar a história da inteligência empresarial, desde as suas primeiras origens até às suas aplicações atuais.

A primeira referência à inteligência empresarial pode ser rastreada até ao final da década de 1950 e início da década de 1960. Nesse momento, os Sistemas de Suporte à Decisão (DSS) começaram a surgir como meio para as organizações analisarem dados e tomarem decisões mais informadas. Hans Peter Luhn é creditado por ter cunhado o termo "inteligência empresarial" no seu artigo de 1958 "A Business Intelligence System", publicado no IBM's Systems Journal (Luhn, 1958). O trabalho de Luhn descreveu o conceito de usar computadores para analisar dados empresariais, preparando as bases para o desenvolvimento da inteligência empresarial moderna.

Nas décadas seguintes, a área da inteligência empresarial continuou a evoluir, com o desenvolvimento de novas ferramentas e tecnologias, tornou-se cada vez mais fácil para as organizações analisarem e compreenderem grandes quantidades de dados. Durante este período, a BI era usada principalmente em grandes organizações e concentrava-se principalmente em fornecer, a quem tomava as decisões, relatórios e *dashboards* que resumiam o desempenho empresarial (Inmon, 2005).

Na década de 1990, a ampla adoção do armazenamento de dados e a emergência de novas tecnologias, como OLAP (Processamento Analítico Online) e mineração de dados, proporcionaram às organizações novas capacidades de gestão e análise de grandes quantidades de dados (Han, 2011). Esses desenvolvimentos abriram caminho para o surgimento da inteligência empresarial moderna e sua ampla utilização em organizações, independentemente da sua dimensão e do setor onde operam.

Nos últimos anos, com a produção de grandes quantidades de dados e análises avançadas o cenário da inteligência empresarial transformou-se cada vez mais(Chandarana, 2017). Hoje, as organizações conseguem analisar grandes quantidades de dados em tempo real, fornecendo insights valiosos sobre o desempenho da empresa, de forma a ajudar a tomar decisões melhores (Kimball, 2013).

Em conclusão, a história da inteligência empresarial é rica e complexa e abrange várias décadas. Nos dias de hoje, a BI é uma parte essencial das organizações modernas e continua a evoluir de forma a atender às mudanças nas necessidades das empresas e de seus decisores (Berkeley, 1964).

2.1.2. Conceito

Business Intelligence (BI) é um termo que se refere às tecnologias, ferramentas e práticas usadas para recolher, armazenar, processar, analisar e visualizar de forma a apresentar os dados com o objetivo de melhorar a tomada de decisões e melhorar o desempenho organizacional (Almeida, 2013). Os sistemas de BI são projetados para responder às necessidades e interesses de diferentes *stakeholders*, tais como, membros executivos, gerentes e analistas, que usam as informações obtidas para tomar

decisões mais acertadas, alocar recursos com maior eficiência e otimizar processos de negócios (Smith, 2019).

O processo de BI começa com a recolha e integração de dados de diversas fontes. Essas fontes podem incluir bases de dados transacionais, redes sociais, sensores e muitas outras fontes de dados estruturados e não estruturados (Sharda, 2018). Os dados geralmente são recolhidos e armazenados de forma “bruta” ou "impura", aos quais devem ser limpos e transformados antes de serem analisados (Kumar, 2017). Esse processo de limpeza e transformação de dados é conhecido como preparação de dados e é uma etapa crítica no processo de BI (Meyer, 2018).

Depois de os dados terem sido recolhidos e preparados, são integrados num data warehouse ou num *data lake* (Inmon, 2005). Um *data warehouse* é um repositório de dados estruturados que é projetado para suportar as necessidades de BI e análise de dados (Kimball, 2013). Um *data lake*, por outro lado, é um repositório de dados brutos e não estruturados projetados para atender às necessidades de análise de *Big Data* (Zikopoulos, 2011). Posteriormente, irei explicar mais aprofundadamente estes dois conceitos assim como as suas diferenças.

As ferramentas e técnicas de BI são usadas para analisar os dados já alocados no *data warehouse* ou no *data lake*. As ferramentas de BI são projetadas para suportar uma ampla gama de tarefas analíticas, incluindo relatórios, *dashboards* , visualização e mineração de dados (Brychcy, 2016). Técnicas de BI, como processamento analítico online (OLAP) e mineração de dados, permitem que os analistas explorem e analisem os dados com mais profundidade e identifiquem padrões, tendências e insights que podem ser usados para informar a tomada de decisões (Zhang, 2017).

O objetivo final do BI é transformar dados em insights acionáveis que podem ser usados para melhorar a tomada de decisões e o desempenho organizacional. Os sistemas de BI são frequentemente usados para dar suporte a uma variedade de funções de negócios, incluindo vendas e marketing, análise financeira e gestão de operações (Lee, 2018).

Apesar dos benefícios potenciais do BI, implementar e usar sistemas de BI pode ser complexo e caro, e requer planeamento e coordenação cuidadosos (Brychcy, 2016). Além disso, os sistemas de BI podem ser difíceis de usar e podem exigir algum treino e suporte significativo para que os utilizadores usufruam plenamente do seu potencial (Lee, 2018).

Apesar desses desafios, a literatura sugere que o BI pode ser um recurso valioso para organizações capazes de implantá-lo e usá-lo com eficiência, conseguindo tomar decisões mais acertadas e, consecutivamente, obtêm vantagem no mercado em que se inserem.

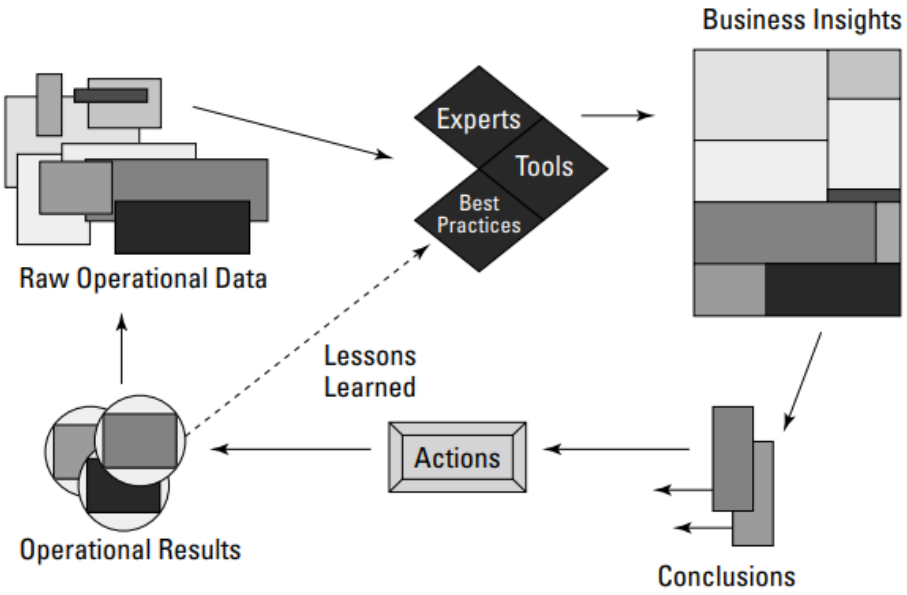

Figura 1 - Novos dados e resultados alimentam constantemente o ciclo da decisão de resultados. Fonte:(Scheps, 2014)

2.1.3. Data Warehouse

O *data warehouse* é um repositório centralizado utilizado para armazenar e gerir grandes quantidades de dados estruturados. É um componente crítico nas estratégias de inteligência empresarial e gestão de dados de muitas organizações, pois permite às organizações realizar análises complexas e tomar decisões informadas. Nesta revisão de literatura, irei apresentar os conceitos e benefícios fundamentais do *data warehouse* e vou explorar alguns dos desafios que as organizações enfrentam ao implementar e manter um *data warehouse*.

Um *data warehouse* é projetado para apoiar a tomada de decisões e é otimizado para consultas e relatórios. Está responsável por armazenar dados de diversas fontes de uma maneira consistente e organizada, normalmente é construído utilizando um modelo de dados estruturado e está baseado num modelo *star schema* ou *snowflake*, que separa os dados de facto dos dados de dimensão (Kimball, 2013). Os dados armazenados num *data warehouse* são geralmente históricos, o que significa que cobrem um período de tempo específico e são integrados a partir de múltiplas fontes, como sistemas transacionais, bases de dados operacionais e fontes externas (Inmon, 2005).

Um dos principais benefícios de um *data warehouse* é fornecer às organizações uma única fonte de verdade dos seus dados. Isso significa que dados de várias fontes são consolidados e integrados num um único local, tornando mais fácil para as organizações aceder e analisar dados, reduzindo o risco de duplicação de dados e erros (R. Kimball, 2002). Além disso, os *data warehouses* são projetados para suportar análises complexas, como análise de tendências e modelagem preditiva, permitindo que as organizações tomem decisões informadas e melhorem a sua performance geral (Chaudhuri, 1997)

Outro benefício de utilizar um data warehouse é que permite às organizações realizem mineração e descoberta de dados. A mineração de dados é o processo de descobrir padrões e relações em grandes conjuntos de dados, que podem ser usados para obter *insights* e fazer previsões sobre tendências

futuras (Fayyad, 1996). A descoberta de dados envolve a exploração de forma *ad hoc*, sem hipóteses predefinidas, para encontrar *insights* e tendências que podem não ter sido inicialmente esperadas (Shneiderman, 1996).

No entanto, implementar e manter um *data warehouse* pode ser um desafio para as organizações. Um dos principais desafios é garantir que os dados no *data warehouse* sejam de alta qualidade e livres de erros. Isso requer processos extensos de limpeza, validação e integração de dados, que podem consumir muito tempo e muitos recursos (Wang, 1996). Além disso, o armazenamento de dados também pode ser caro, pois requer competências e tecnologias especializadas, como modelagem de dados e ferramentas ETL (extrair, transformar, carregar) (Han, 2011).

Concluindo, o *data warehouse* é um componente crítico das estratégias de *business intelligence* e gestão de dados de muitas organizações. Fornece às organizações uma fonte única de confiança para os seus dados, permite que se executem análises complexas e oferece suporte à mineração e descoberta de dados. No entanto, implementar e manter um *data warehouse* pode ser desafiador, pois requer dados de alta qualidade, competências especializadas e recursos significativos.

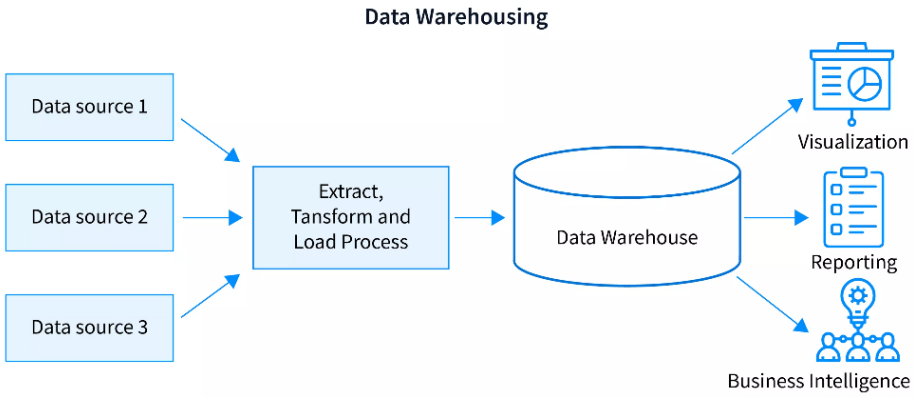

Figura 2 - Architecture of Data Warehouse. Fonte: (Srikanth Varma, 2023)

2.2. Cloud Computing

2.2.1. Conceito

A computação em nuvem é uma tecnologia em rápido crescimento que permite aos utilizadores armazenar, aceder e gerir dados e aplicações através da internet, em vez de nos servidores locais ou dispositivos pessoais (Armbrust, 2010). O conceito de computação em nuvem existe há várias décadas, mas só recentemente se tornou uma tecnologia de massa devido aos avanços na velocidade e credibilidade da internet (O'Reilly, 2009).

Existem três tipos principais de serviços de computação em nuvem: infraestrutura como serviço (IaaS), plataforma como serviço (PaaS) e software como serviço (SaaS) (Mell e Grance, 2011). O IaaS fornece aos utilizadores recursos de computação virtualizados, como servidores, armazenamento e rede, que

podem ser acedidos e geridos pela internet. O PaaS fornece uma plataforma para desenvolver, testar e implantar aplicações de software, enquanto o SaaS entrega aplicativos de software pela internet, eliminando a necessidade de os utilizadores instalarem e executarem as aplicações nos seus próprios computadores (Salesforce, 2023).

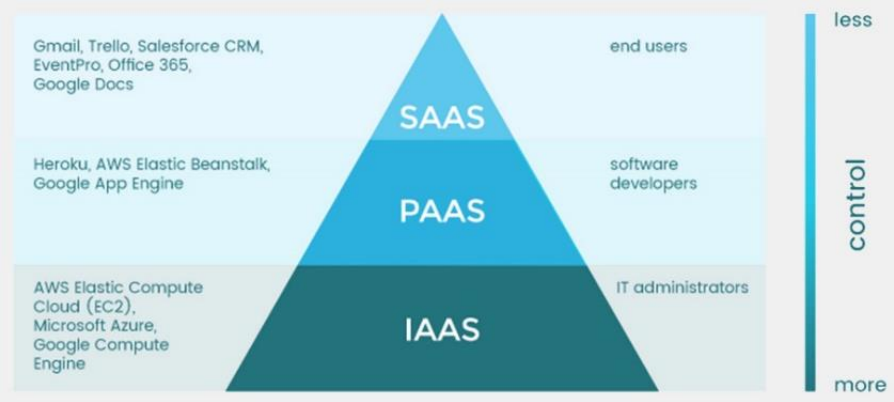

Figura 3 - IaaS, PaaS, and SaaS – how do they differ? Fonte: (Elvis Plesky, 2019)

Um dos principais benefícios da computação em nuvem é sua escalabilidade (Amazon Web Services, 2023). Com a computação em nuvem, os utilizadores podem aumentar ou diminuir rapidamente a quantidade de recursos de computação que estão a utilizar, dependendo de suas necessidades. Isso permite que as organizações respondam a mudanças de demandas, como picos de tráfego, sem ter que fazer grandes investimentos em novo hardware ou software (Microsoft Azure, 2023).

Outro benefício da computação em nuvem é sua acessibilidade (IBM, 2023). Como os recursos de computação em nuvem são acessíveis pela internet, os utilizadores podem aceder aos dados e aplicações em qualquer lugar, e a qualquer momento, necessitando apenas de um dispositivo com uma conexão estável à internet. Esta tecnologia facilita a colaboração e partilha de informações das organizações, mesmo quando os seus funcionários estão a trabalhar de forma remota (Gartner, 2019).

A computação em nuvem também proporciona economias de custo em comparação aos modelos de computação tradicionais. As plataformas de computação em nuvem, como a Amazon Web Services (AWS), Google Cloud Platform (GCP) e Microsoft Azure operam em grande escala, podendo aproveitar as economias de escala como uma vantagem, fornecendo aos seus clientes serviços com custos reduzidos. Além disso, as organizações podem reduzir os custos de gestão da sua própria infraestrutura de IT e podem ainda evitar os custos iniciais associados à compra e manutenção de hardware e software (de Haan, 2017).

No entanto, também há preocupações de segurança e privacidade associadas à computação em nuvem. Armazenar dados e aplicações na nuvem pode aumentar o risco de violações de dados, já que os dados não estão armazenados em servidores locais protegidos por *firewalls* e outras medidas de segurança (Buyya, 2008). Adicionalmente, as organizações devem confiar nas medidas de segurança

implementadas pela sua plataforma de computação em nuvem, visto que, não têm controlo total sobre a segurança dos seus dados pois estão armazenados na nuvem (Khan, 2015).

Em conclusão, a computação em nuvem é uma tecnologia em rápido crescimento que oferece vários benefícios, incluindo escalabilidade, acessibilidade e economias de custo. No entanto, as organizações também devem considerar as implicações de segurança e privacidade de armazenar os seus dados e aplicativos na nuvem (Zhang, 2014).

2.2.2. Big Data

Big data refere-se a conjuntos de dados extremamente grandes e complexos que estão além da capacidade dos métodos tradicionais de processamento de dados. É caracterizado pelos seus 5V's - volume, velocidade, variedade, veracidade e valor. Esses 5V's representam os principais desafios e oportunidades associados ao *big data* (Kwon, 2017).

O primeiro V é volume, que se refere à quantidade massiva de dados gerados todos os dias. De acordo com a IBM, mais de 2,5 triliões de bytes de dados são criados todos os dias, e esse número suscetível a um aumento exponencial nos próximos anos (IBM, 2021). O segundo V é velocidade, que se refere à velocidade com que os dados são gerados e processados. Com o aumento dos fluxos de dados em tempo real, os sistemas de processamento de *big data* devem ser capazes de lidar com dados em alta velocidade para acompanhar o ritmo da criação de dados (Kwon, 2017).

O terceiro V é variedade, que se refere à diversidade de tipos e fontes de dados. O *big data* não se limita a dados estruturados, também inclui dados não estruturados, como texto, imagens e vídeos, que requerem métodos de processamento mais sofisticados (Kwon, 2017). O quarto V é veracidade, que se refere à precisão e confiabilidade dos dados. À medida que os dados se tornam mais complexos e diversos, garantir a sua veracidade torna-se cada vez mais desafiador (Manyika, 2011).

Por fim, o quinto V é valor, que se refere aos *insights* e potenciais benefícios que podem ser derivados do *big data*. A análise de *big data* permite que as organizações extraiam *insights* valiosos e tomem decisões orientadas por dados que podem melhorar seus resultados de negócios (Manyika, 2011). Isso é especialmente importante para as organizações que geram grandes quantidades de dados, como dados de redes sociais, financeiros e de saúde, pois esses *insights* podem informar a tomada de decisões críticas e impulsionar o crescimento dos negócios.

Os benefícios do *big data* são numerosos, mas também existem desafios que devem ser abordados. Um dos principais desafios é a necessidade de ferramentas avançadas de processamento e análise de dados que possam lidar com o volume, velocidade e variedade do *big data*. Além disso, garantir a precisão e confiabilidade dos dados é fundamental para tomar decisões informadas (Kwon, 2017).

Outro desafio é a necessidade de políticas sólidas de governança e privacidade de dados. À medida que o *big data* se torna mais difundido, garantir o uso ético e responsável dos dados é primordial. As organizações devem tomar medidas para proteger as informações pessoais e cumprir as regulamentações relevantes de proteção de dados (Manyika, 2011).

Em conclusão, o *big data* representa uma grande oportunidade para as organizações obterem *insights* valiosos e melhorarem seus resultados de negócios. No entanto, também apresenta desafios

significativos que devem ser cuidadosamente geridos. Ao abordar os 5V's do *big data* e implementar políticas sólidas de governança e privacidade de dados, as organizações podem desbloquear todo o potencial do *big data* enquanto minimizam os seus riscos.

2.2.3. Data Lake

Os *data lakes* podem ser descritos como repositórios centralizados que armazenam grandes volumes de dados brutos, não estruturados e semiestruturados em qualquer escala. O conceito de *data lakes* foi introduzido pela primeira vez no final dos anos 2000 e desde então que se tornou uma solução popular para as organizações que procuram armazenar e analisar grandes quantidades de dados (Dutta, 2018). Ao contrário dos *data warehouses* tradicionais, que são projetados para armazenar dados estruturados e otimizados para consultas e análises rápidas, os data lakes são projetados para armazenar dados no seu formato bruto, permitindo maior flexibilidade e escalabilidade.

Os *data lakes* ganharam popularidade nos últimos anos devido ao aumento de grandes quantidades de dados e à necessidade crescente das organizações de armazenar, gerir e analisar grandes quantidades de dados. Estando os *data lakes* capacitados para armezenar os dados de forma bruta, permite que as organizações mantenham todos os seus dados num só repositório, tornando mais fácil o acesso e a análise. Torna-se uma grande vantagem para organizações que armazenam grandes quantidades de dados não estruturados, como redes sociais, arquivos de log e dados de IoT, uma vez que, esses tipos de dados são difíceis de armazenar e analisar usando métodos tradicionais de *data warehouse* (Dutta, 2018).

Um dos principais benefícios dos *data lakes* é a sua escalabilidade. Os *data lakes* podem armazenar e processar grandes quantidades de dados, permitindo que as organizações ampliem a sua capacidade de armazenamento de dados conforme necessário, ajustando-se às suas necessidades (Zikopoulos, 2011).

Outro benefício dos *data lakes* é a sua eficiência de custos. Ao contrário dos *data warehouses* tradicionais, que exigem *hardware* e *software* caros, os *data lakes* podem ser implementados usando *hardware* de mercado e *software* de código aberto, tornando-se uma solução eficaz de custos para organizações de todas as dimensões. Isso é particularmente importante para organizações de menores dimensões que podem não ter o orçamento para investir em soluções caras de armazenamento de dados (Gandomi & Haider, 2015).

Os *data lakes* também permitem que as organizações usufruam de técnicas analíticas avançadas, como aprendizagem automática e mineração de dados, para extrair *insights* valiosos de seus dados. Isto é particularmente importante para organizações que desejam usar análises de grandes dados para melhorar seus resultados empresariais. Ao armazenar dados num *data lake*, as organizações podem aceder e analisar os seus dados usando uma variedade de ferramentas e técnicas analíticas, permitindo-lhes extrair *insights* significativos que podem informar a tomada de decisão e impulsionar o crescimento empresarial (Gandomi & Haider, 2015).

3. METODOLOGIA

De forma a garantir que os objetivos do projeto de estágio são alcançados com sucesso, terei a oportunidade de pôr em prática os conhecimentos e técnicas que adquiri no mestrado. Com os meus conhecimentos serei capaz de extrair dados de uma fonte de dados externa, realizar todo o fluxo de dados através de diferentes camadas (Raw, Standardized, Stage e Persistence) e, por fim, apresentar os dados de forma a estarem aptos a ser consumidos por ferramentas de análise e visualização.

O foco deste relatório de estágio irá recair de uma forma generalizada sobre a extração dos dados da fonte, utilizando pipelines no Azure Data Factory, assim como, os processos existentes nas diferentes camadas de transformação de dados, processos estes desenvolvidos em *notebooks* de Databricks. De forma a auxiliar este fluxo utilizamos também uma Azure SQL database para alocar tabelas de configuração necessárias durante todo o processo e a ferramenta AzureKeyVault para armazenar e encriptar todas as chaves e dados sensíveis. Todos os processos mencionados acima irão ser detalhados nos próximos passos.

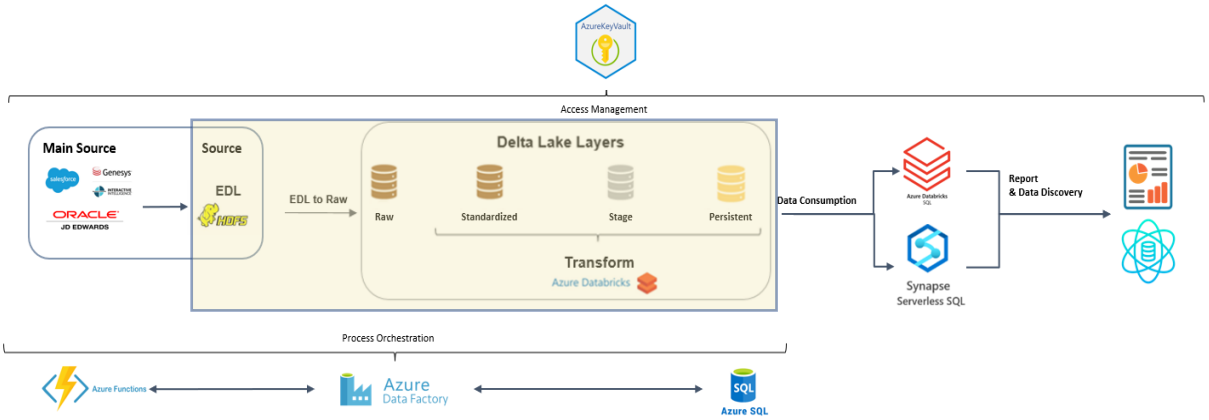

Figura 4 - Arquitetura e scope do projeto

3.1. PROCESSOS

3.1.1. Extração

De uma forma geral, o primeiro passo da *framework* baseia-se em extrair os dados existentes das variadas fontes e armazená-los em *cloud* num Azure Storage Account (Delta Lake Gen 2). Para tal, para tal foram utilizados serviços de *cloud* como Azure KeyVault, Azure Storage Account, Azure SQL Database e Azure Data Factory para a execução destes procedimentos.

Por norma, quando se quer aplicar uma solução de *analytics* é necessário conectar diferentes sistemas, trabalhar os dados e integrá-los de forma a que seja possível consultar os dados de forma única e estandardizada, para tal, é necessário extrair a informação desses sistemas e construir uma solução numa plataforma única. Neste projeto, os dados inicialmente são provenientes de fontes como Oracle, Salesforce e Genesys, no entanto, foi utilizada a tecnologia a HDFS (Hadoop Distributed File System)

que irá armazenar os dados de todas as diferentes fontes num repositório único, a partir daí é possível aplicar os processos ETL. Enterprise Data Loader (EDL) é o nome dado ao HDFS em que armazenamos os dados dos diversos sistemas.

Após os dados estarem compilados e organizados no sistema de HDFS, foram desenvolvidos pipelines dinâmicos no Azure Data Factory com o objetivo de extrair dados do sistema fonte EDL para a pasta Raw que se encontra na Azure Storage Account (Delta Lake Gen 2).

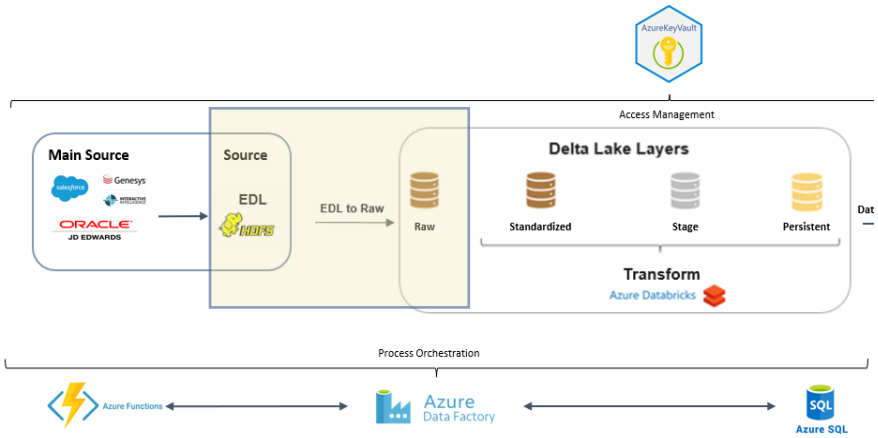

Figura 5 - Arquitetura da extração

3.1.1.1. Pipelines de Extração para a camada Raw

Os pipelines desenvolvidos têm como objetivo realizar um processo de *Lift and Shif*, isto é, o processo de mover uma aplicação ou fluxo de dados de uma infraestrutura IT tradicional (*on-premesis*) para uma infraestrutura em *cloud*, sem que se faça nenhuma alteração na arquitetura ou no design do mesmo. Desta forma, o processo de migração e carregamento torna-se bastante mais rápido comparando com outros modelos de migração. Esta foi a solução encontrada pela equipa de trabalho tendo em conta as exigências do cliente que, neste caso, era reduzir ao máximo os tempos de extração e carregamento dos dados na *cloud*.

Foram desenvolvidos dois pipelines no Azure Data Factory para a realização deste processo, o pipeline PL_E_MASTER que orquestra todo o processo com o auxílio de tabelas de configuração alocadas na Azure SQL database e o pipeline PL_E_EDL_PARQUET que realiza o processo de migração em si. Os pipelines desenvolvidos ligam-se aos diferentes serviços através de *linked services* como podemos verificar na tabela abaixo apresentada.

Connection Type	Linked Service Name	Connects to
Azure Data Lake Storage Gen2	ls_asa_eus_delanalytics_01	asa-eus-delanalytics-01
Azure Key Vault	ls_akv_eus_delanalytics_01	akveusdelanalyticsdev01
Azure SQL Database	ls_adb_eus_delanalytics_01	adb-eus-delanalytics-dev-01
HDFS	ls_hdfs_edl	EXTERNAL SOURCE: EDL

Tabela 1 - Serviços utilizados e respetivos linked services

Pipeline PL_E_MASTER

O pipeline PL_E_MASTER é responsável por orquestrar o processo de migração e carregamentos de dados e, para tal, utiliza uma *view* da tabela de configuração que se encontra alocada na Azure SQL Database; a ligação a esta base de dados é feita através do *linked service* anteriormente mencionado.

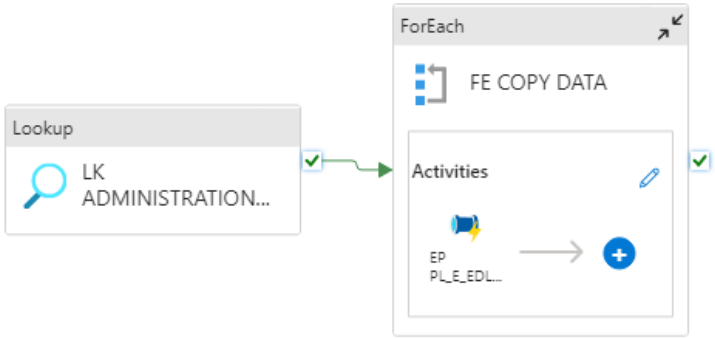

Figura 6 - Pipeline PL_E_MASTER

O pilpeline PL_E_MASTER apresenta como parâmetro de *input* o Model do tipo *string*, é através deste parâmetro que é realizada a leitura à *view* da tabela de configuração.

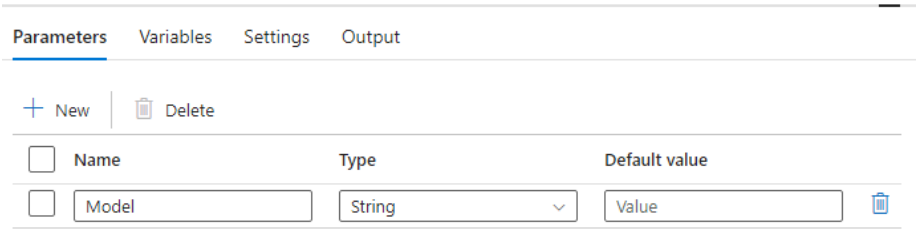

Figura 7 - Pipeline PL_E_MASTER – Parâmetros de *input*

A *view* contém informação relativa à fonte e ao destino dos dados. Alguma desta informação é meramente informativa, no entanto, grande parte é utilizada no processo, como por podemos observar:

- Model – O Nome do modelo a processar.
- SourceSystemName -O Nome da fonte de dados.
- SourceSystemType - O tipo de fonte de dados.
- SourceLocationName – A localização (*path*) da fonte de dados.
- SourceObjectName - O *schema* e a tabela da fonte de dado.
- SourceFilter – O Filtro opcional a aplicar à fonte de dados (NULL caso nao seja aplicado).
- DestinationSystemName – O nome do destino.
- DestinationSystemType – O tipo de destino.
- DestinationContainerName – O nome do container de destino.
- DestinationLocationName – A localização (path) do destino.
- DestinationObjectName – O schema e tabela de destino.
- DestinationObjectType – O tipo de objeto no destino.
- FlagActive -Pode ter dois valores 0 e 1. Caso tenha o valor 1 será processado.
- SourceReadCommand – Coluna gerada na view, sendo que apresenta como valor uma *query* construida com bases nas colunas SourceObjectName e SourceFilter. Esta query é utilizada para selecionar qual a tabela que irá ser extraída assim como quais os filtros a aplicar em cada extração. (exemplo: 'SELECT * FROM' + SourceObjectName + ' WHERE ' + SourceFilter)

O pipeline é constituído por duas atividades, a primeira atividade é um *Lookup* e tem como função ler os dados provenientes da *view* da tabela de configuração ([bi_admin].[v_CopyDataConfig]), para tal, utilizando a seguinte expressão:

```
@concat ('SELECT * FROM bi_admin].[v_CopyDataConfig] WHERE FlagActive = 1 AND Model = ''',pipeline().parameters.Model,'''')
```

Com esta expressão vamos selecionar todas os registos da *view* ([bi_admin].[v_CopyDataConfig]) onde a FlagActive tem o valor de 1 e o Model corresponde com o parâmetro de *input* do pipeline, por outras palavras, iremos selecionar todas as tabelas com as suas respetivas informações que queremos extrair cujo tendo em conta o modelo de *input*.

A segunda atividade deste pipeline é um *ForEach* que tem como função executar as atividades que estiverem no seu interior a cada iteração, neste caso é um pipeline. Desta forma, a cada iteração os parâmetros de *input* do pipeline PL_E_EDL_PARQUET são preenchidos com os dados provenientes da expressão realizada à *view* e o pipeline PL_E_EDL_PARQUET é executado. Uma vez que, cada registo da *view* representa uma tabela a ser extraída, cada iteração com a respetiva execução do pipeline PL_E_EDL_PARQUET representa a extração de uma tabela.

Pipeline PL_E_EDL_PARQUET

O pipeline PL_E_EDL_PARQUET é responsável pela migração das tabelas fonte, no entanto, está contruído para migrar tabela a tabela. Para além de ser responsável pela migração o pipeline regista o estado da execução numa tabela de Log que está alocada no Azure SQL database.

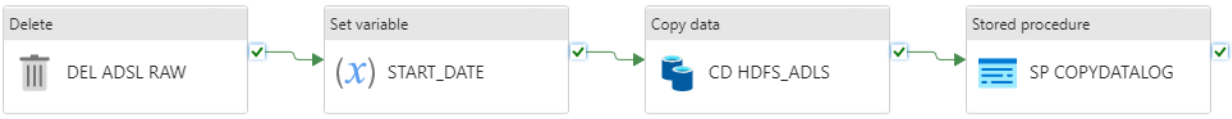

Figura 8 - Pipeline PL_E_EDL_PARQUET

O pipeline PL_E_EDL_PARQUET apresenta diversos parâmetros de input como podemos verificar na Figura 9. Estes parâmetros são preenchidos com valores provenientes do pipeline *master*, mais especificamente, os valores de cada iteração da atividade *ForEach*.

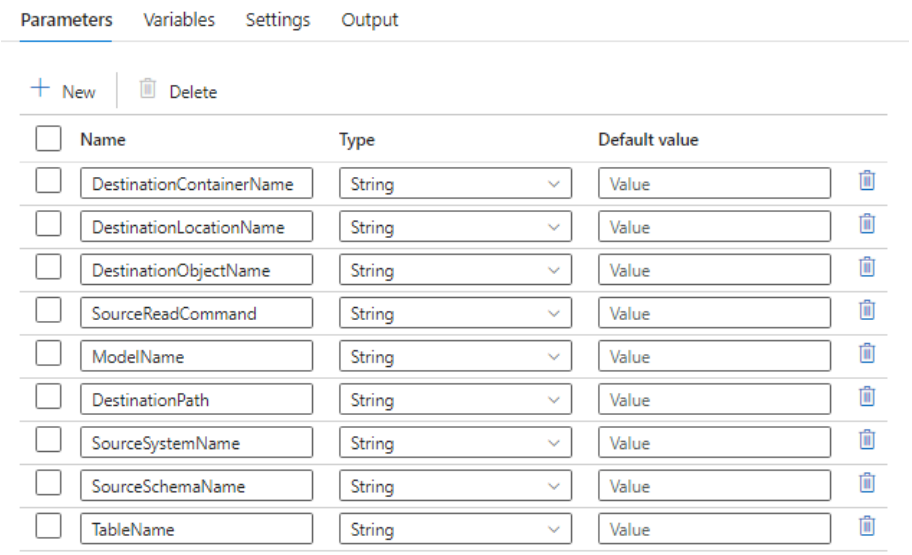

Figura 9 - Pipeline PL_E_EDL_PARQUET – Parâmetros de *input*

O pipeline é constituído por quatro atividades diferentes, sendo a primeira um *Delete*. Uma vez que estamos a fazer uma migração do tipo *Lift and Shift,* sempre que os pipelines são executados a extração das tabelas é realizada como um todo (*full*) daí existir esta atividade de *Delete* para eliminar todos os registos que possam existir no Azure Storage Account (Delta Lake Gen 2) de destino, para que possam ser ingeridos novamente na íntegra e tal como se encontram na fonte.

A segunda atividade é uma *Set variable*, que é utilizada para registar a data de início da extração. Esta variável é utilizada mais tarde para registar a data e tempo de começo da execução e a duração do pipeline na tabela de Log alocada na Azure SQL database.

A terceira atividade é um *CopyData*, esta atividade está responsável por se conectar à fonte (HDFS) e ao destino (Azure Delta Lake) através dos linked services e realizar a cópia dos ficheiros parquet. Após a cópia dos ficheiros estes ficaram alocados no Azure Storage Account no caminho indicado pelos parâmetros de input que o pipeline recebe, por exemplo, neste caso o caminho de destino no Azure Storage Account seria a concatenação dos parâmetros *string* de *input* DestinationContainerName, DestinationLocationName e DestinationObjectName. As restantes partições da tabela são realizadas de forma automática.

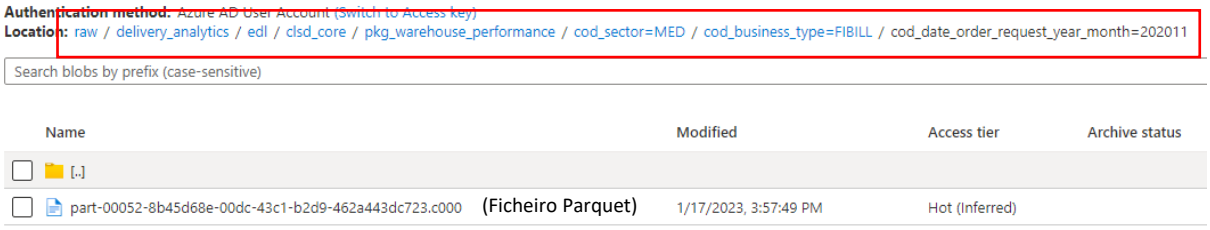

Figura 10 - Azure Storage Account – Exemplo de path de destino do ficheiro parquet

Por último, a quarta atividade é uma *Store Procedure* responsável por preencher a tabela de Log que se encontra alocada na Azure SQL database. Esta *Store Procedure* regista para além dos parâmetros informativos como o nome do modelo, nome da tabela, *schema* utilizado e o caminho de destino regista também o *status* da execução (Succeeded /Failed), a data e tempo em que começou e acabou a execução, a duração da extração (GETDATE() – (Set variable da Start Date) em minutos) e o número de tabelas que extraiu. Assim, é possível agregar todos os registos das extrações numa só tabela, ficando com um histórico das migrações.

Desta forma, termina a extração dos dados da fonte de dados EDL (*on-premesis*) para o Azure Storage Account (Delta Lake Gen 2), assim sendo, os dados encontram-se prontos a iniciar a próxima fase do projeto, a transformação dos dados.

3.1.2. Tranformação

Ultrapassada a etapa de extração dos dados inicia-se a próxima fase do projeto, a transformação dos dados, a qual é composta por 3 camadas (Standardized, Stage e Persistence). Os processos de transformação foram desenvolvidos em *notebooks* Databricks e têm como principais objetivos converter, limpar, agregar e estruturar os dados que se encontram de forma "bruta" na camada Raw de forma a torná-los mais adequados e úteis para análise. O processo de transformação também é a etapa onde são aplicadas todas as regras de negócio adjacentes aos dados, sendo que a aplicação das regras de negócio apenas se realiza na camada Stage, isto é, os dados já sofreram algumas transformações na camada Standardized não se encontrando no seu estado primário.

Nos seguintes capítulos deste relatório de estágio irei explicar, camada a camada, todos os processos e metodologias utilizadas durante a transformação dos dados.

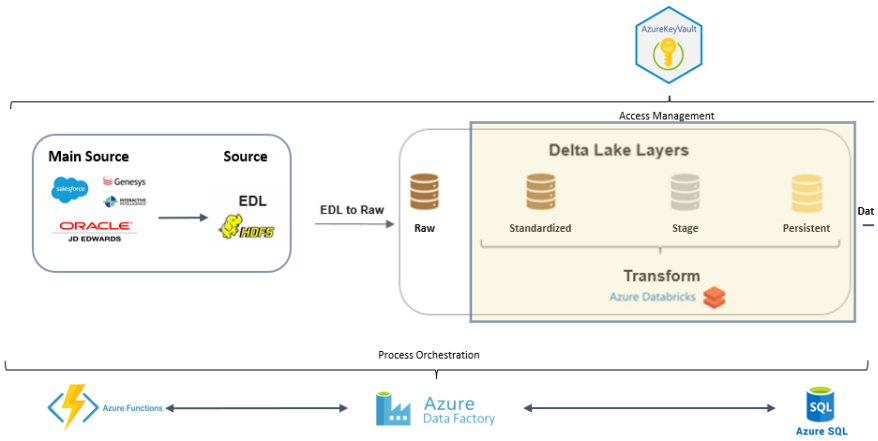

Figura 11 - Arquitetura – Fase de Transformação

Durante a fase de transformação existem alguns *notebooks* que são transversais às diferentes camadas, isto é, de forma a não repetir código em cada camada foram criados *notebooks* que são acedidos pelas diferentes camadas e que contêm funções comuns, desta forma, em cada camada basta importar o *notebook* e chamar a função que se pretende utilizar.

Notebooks	Funcionalidade
Commun Functions	Notebook responsável por centralizar funções que são utilizadas pelas diversas camadas de transformação.
AzureKeyVault	Notebook responsável por aceder aos segredos do Azure KeyVault, como passwords, conectionStrings, entre outros.
Connection SQl Admin	Notebook responsável por se conectar à Azure SQL Database e materializar as tabelas de configuração necessárias em Databricks delta tables.

Tabela 2 - Notebooks transversais às camadas de Transformação

3.1.2.1. Camada Standardized

A camada Standardized é responsável por ler as tabelas (ficheiros parquet) que se encontram armazenados na pasta Raw da Azure Storage Account, por sua vez, a camada Standardized filtra os campos para cada tabela, faz as conversões de tipo de dados tendo em conta as necessidades e escreve como tabelas delta na pasta Standardized na Storage Account. Para uma melhor interpretação da camada podemos visualizar no gráfico abaixo.

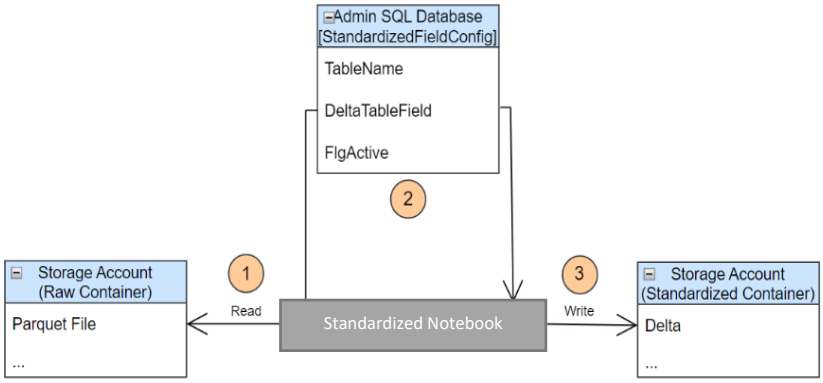

Figura 12 - Fluxo de trabalho da camada Standardized

Para a realização do processo de filtragem dos campos de cada tabela é utiliza uma tabela de configuração ([admin].[bi_admin_v_edl_fieldconfig]) materializada no Databricks, esta tabela é proveniente da *view* [bi_admin].[v_EDLFieldConfig] que está alocada na Azure SQL Database (esta materialização foi realizada pelo *notebook* Connection SQl Admin). Para obter os campos de filtragem (deltaTableField), primeiramente, é necessário iterar a tabela de configuração e para cada iteração filtrar pelo nome da tabela (tableName) e pela flgActive com valor "*True*".

tableName	deltaTableField	flgActive
pkg_warehouse_performance	cod_sector	True
pkg_warehouse_performance	load_dttm	True
pkg_warehouse_performance	cod_date_order_request_year_month	True

Tabela 3 - Tabela de Configuração [admin].[bi_admin_v_edl_fieldconfig]

Depois de iterar a tabela de configuração com as respetivas filtragens, obtemos uma lista com o nome das colunas (deltaTableField) da tabela em questão. Com esta lista é agora possível filtrar o *raw dataframe* que contém várias colunas que não precisamos.

Exemplo: columns = ['cod_sector', 'load_dttm','cod_date_order_request_year_month']

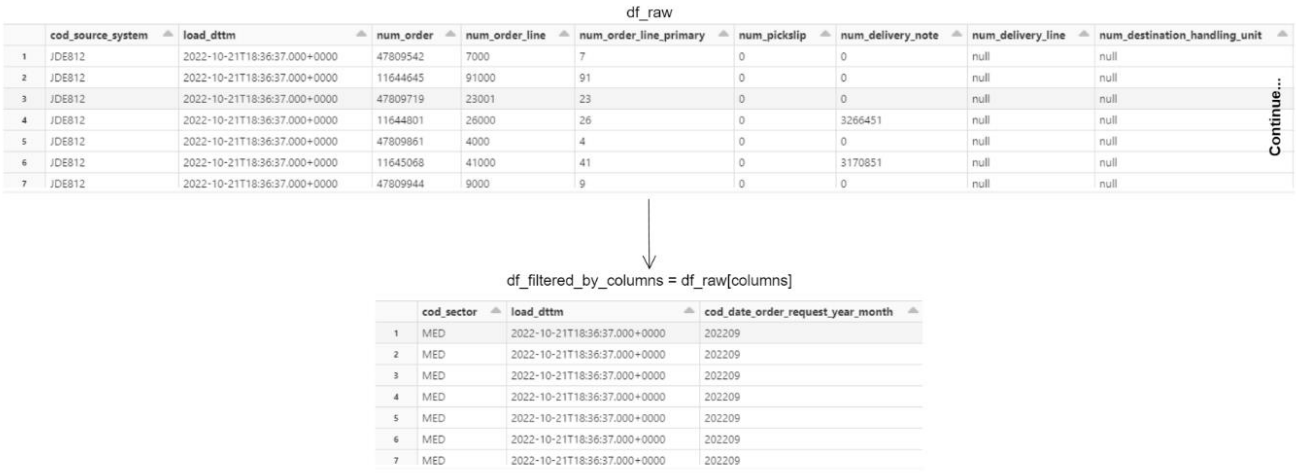

Figura 13 - Exemplo de filtragem de colunas de um *dataframe raw*

Este *notebook* está concebido para carregar na pasta Standardized da Storage Account apenas os novos registos ou registos que sofreram alterações. Para detetar estes casos, o *notebook* tem em consideração duas colunas, a batchDate e a incrementalDate.

Para identificar quais os registos que devem ser carregados, o *notebook* filtra os registos da pasta Raw na Storage Account onde os valores da batchDate são superiores ao máximo valor da batchDate na pasta Standardized. Desta forma, é possível detetar os novos registos e os que sofreram alterações que ainda não se encontram carregados na pasta Standardized na Azure Storage Account.

Para identificar qual o range de incrementalDate de registos que devem ser carregados, é necessário identificar o valor mínimo e máximo da incrementalDate que existe na pasta Raw, tendo em conta que já foi aplicado o filtro acima mencionado.

Caso ainda não existam registos na pasta Standardized, isto é, o *notebook* está a correr pela primeira vez, o processo está idealizado para carregar os últimos dois anos de registos, iniciando-se sempre no dia 1 de janeiro.

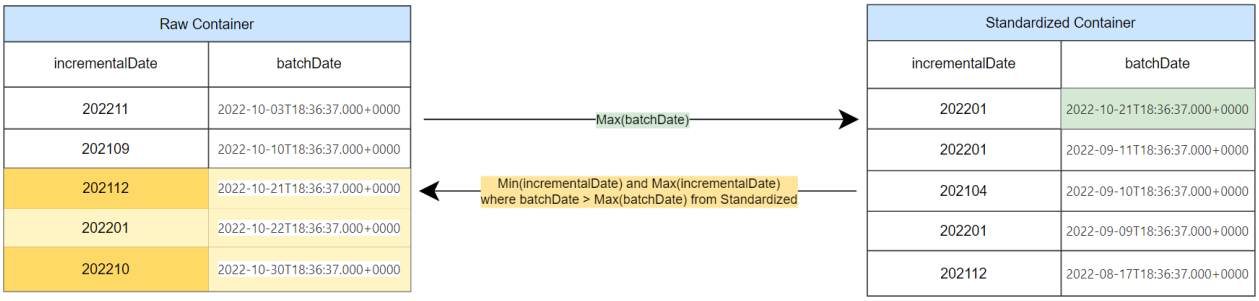

Figura 14 - Exemplo de *range* de registos que devem ser carregados

Esta camada quando está a escrever os dados na Azures Storage Account utiliza o método de *Overwrite* onde é aplicada uma clausula de *ReplaceWhere* com os filtros e regras acima mencionados: este método é utilizado apenas com o objetivo de atualizar ou inserir registos.

Após carregar os registos identificados anteriormente na pasta Standardized da Azure Storarge Account, o notebook materializa a tabela como uma Databricks delta table. As tabelas são materializadas como Databricks delta tables de forma a agilizar a conexão e execução de queries à tabela na camada seguinte.

3.1.2.2. Camada Stage

Esta camada é responsável por ler a Standardized delta table que foi materializada no Databricks pela camada anterior, aplicar as regras de negócio adjacentes e gerir o período de tempo que o processo irá ser carregado e escrito no Azure Storage Account na pasta Stage como delta, para tal, utiliza uma tabela de configuração materializada no Databricks.

O gráfico abaixo ajuda-nos a perceber o fluxo do *notebook* Stage acima descrito. A aplicação das regras de negócio é realizada depois da leitura, isto é, depois do passo 1.

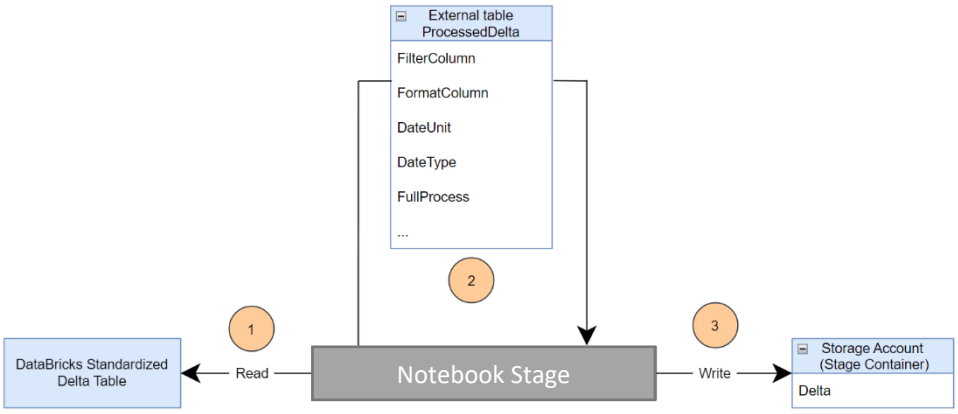

Figura 15 - Fluxo de trabalho da camada Stage

As regras de negócio foram recolhidas e construídas pelos *business analists* da equipa, em concordância com o cliente e as suas necessidades. Esta camada, ao contrário das restantes, é composta por vários *notebooks*, um por tabela, uma vez que, as regras de negócio aplicam-se individualmente a cada tabela.

Este *notebook* utiliza a tabela de configuração [bi_admin].[ProcessDelta], materializada no Databricks, de forma a controlar o período de tempo que o processo irá carregar. Esta tabela irá ser filtrada tendo em conta os parâmetros tableName e modelName que o *notebook* recebeu como *input*.

modelName	tableName	filterColumn	formatColumn	dateUnit	dateType	fullProcess
WEEKLY-01	pkg_warehouse_performance	cod_date_order_request_year_month	yyyyMM	NULL	NULL	1
DAILY-01	pkg_warehouse_performance	cod_date_order_request_year_month	yyyyMM	3	Month	0

Tabela 4 - Tabela de Configuração [bi_admin].[ProcessDelta]

Se o valor da coluna fullProcess for 1, significa que é para carregar os últimos dois anos de registos, iniciando-se sempre no início do ano, isto é, 1 de Janeiro. As colunas filterColumn e formatColumn são utilizadas para filtrar os dados com as respetivas datas.

Se o valor da coluna fullProcess for 0, é necessário ter em consideração as colunas dateUnit e dateType, à data de hoje iremos então subtrair os valores que se encontram nessas colunas, iniciando-se sempre no início do mês ou do ano. Por exemplo, se o valor da dateUnit for 3 e o valor do dateType for *Month* queremos carregar dados desde a data de hoje -3 meses começando no dia 1.

Depois de executados todos os processos descritos anteriormente os registos são escritos no Azure Storage Account na pasta da Stage em formato delta. Para a escrita é utilizado o método de *overwrite* onde é aplicada uma cláusula de *replaceWhere* com os filtros e regras acima mencionados, este método é utilizado com o objetivo de apenas atualizar ou inserir registos.

Após carregar os registos identificados anteriormente na pasta Standardized da Azure Storarge Account, o notebook materializa a tabela como uma Databricks delta table. As tabelas são materializadas como Databricks delta tables de forma a agilizar a conexão e execução de queries à tabela na camada seguinte.

3.1.2.3. Camada Persistent

Esta camada é responsável por ler as várias Stage delta tables que foram materializadas em Databricks pela camada anterior e trazer todos os registos das tabelas tendo em conta que são aplicados filtros (por tabela).

Figura 16 - Fluxo de trabalho da camada Persistent

Esta camada está construída de modo a carregar tabela a tabela todos os registos que se encontrem nas Stage delta tables. De forma a identificar quais os registos que devem ser carregados na camada Persistent temos de identificar qual o mínimo e máximo valor da incrementalDate que existe em cada tabela da Stage.

Tomando a incrementalDate como sendo a coluna cod_date_order_request_year_month podemos identificar que o valor máximo na Stage é 202211 e o mínimo 202209, para os filtros (parâmetro de *input*) aplicados cod_sector='MED' e cod_source_system='EDL'.

Stage Table

Data....	cod_sector	cod_source_system	cod_date_order_request_year_month
xy4	MED	EDL	202209
xy5	MED	EDL	202210
xy6	MED	EDL	202211
xy7	MED	CDL	202209
xy8	MED	CDL	202210
xy9	MED	CDL	202211
xy10	PHARMA	EDL	202209
xy11	PHARMA	EDL	202210
xy12	PHARMA	EDL	202211
xy13	PHARMA	CDL	202209
xy14	PHARMA	CDL	202210
xy15	PHARMA	CDL	202211

Persistent Table

Data....	cod_sector	cod_source_system	cod_date_order_request_year_month	audit_load_date_persistent
xy1	MED	EDL	202206	2022-11-21T14:39:54.778+0000
xy2	MED	EDL	202207	2022-11-21T14:39:54.778+0000
xy3	MED	EDL	202208	2022-11-21T14:39:54.778+0000
xy4	MED	EDL	202209	2022-11-21T14:39:54.778+0000
xy5	MED	EDL	202210	2022-11-21T14:39:54.778+0000
xy7	MED	CDL	202209	2022-11-21T13:19:15.778+0000
xy8	MED	CDL	202210	2022-11-21T13:19:15.778+0000
xy10	PHARMA	EDL	202209	2022-11-21T12:34:52.778+0000
xy11	PHARMA	EDL	202210	2022-11-21T12:34:52.778+0000
xy13	PHARMA	CDL	202209	2022-11-21T11:22:12.778+0000
xy14	PHARMA	CDL	202210	2022-11-21T11:22:12.778+0000

Figura 17 - Exemplo de escrita na Persistent – Antes de correr o *notebook*

Stage Table

Data....	cod_sector	cod_source_system	cod_date_order_request_year_month
xy4	MED	EDL	202209
xy5	MED	EDL	202210
xy6	MED	EDL	202211
xy7	MED	CDL	202209
xy8	MED	CDL	202210
xy9	MED	CDL	202211
xy10	PHARMA	EDL	202209
xy11	PHARMA	EDL	202210
xy12	PHARMA	EDL	202211
xy13	PHARMA	CDL	202209
xy14	PHARMA	CDL	202210
xy15	PHARMA	CDL	202211

Persistent Table

Data....	cod_sector	cod_source_system	cod_date_order_request_year_month	audit_load_date_persistent
xy1	MED	EDL	202206	2022-11-21T14:39:54.778+0000
xy2	MED	EDL	202207	2022-11-21T14:39:54.778+0000
xy3	MED	EDL	202208	2022-11-21T14:39:54.778+0000
xy4	MED	EDL	202209	2022-11-22T14:39:54.778+0000
xy5	MED	EDL	202210	2022-11-22T14:39:54.778+0000
xy6	MED	EDL	202211	2022-11-22T14:39:54.778+0000
xy7	MED	CDL	202209	2022-11-21T13:19:15.778+0000
xy8	MED	CDL	202210	2022-11-21T13:19:15.778+0000
xy10	PHARMA	EDL	202209	2022-11-21T12:34:52.778+0000
xy11	PHARMA	EDL	202210	2022-11-21T12:34:52.778+0000
xy13	PHARMA	CDL	202209	2022-11-21T11:22:12.778+0000
xy14	PHARMA	CDL	202210	2022-11-21T11:22:12.778+0000

Figura 18 - Exemplo de escrita na Persistent – Depois de correr o *notebook*

Desta forma conseguimos identificar que para a escrita na Persistent do Azure Storage Account as cláusulas devem ser as seguintes:

- incrementalDate >= 202209
- incrementalDate <= 202211
- sourceFilter (cod_sector='MED' e cod_source_system='EDL')

Depois de executados os filtros e identificados os registos de cada tabela, os dados são escritos no Azure Storage Account na pasta da Persistent em formato delta, para a escrita é utilizado o método de *overwrite* onde é aplicada uma cláusula de *replaceWhere* com as cláusulas acima identificadas, este método é utilizado com o objetivo de apenas atualizar ou inserir novos registos.

Após carregar os registos identificados anteriormente na pasta Standardized da Azure Storarge Account, o *notebook* materializa a tabela como uma Databricks delta table.

Sendo esta a última camada da fase de transformação, os dados passaram por diversas etapas de metamorfose, estando agora mais refinados e organizados. Os dados encontram-se prontos a serem consumidos para análise e, posteriormente, para visualização.

4. RESULTADOS E DISCUSSÃO

O objetivo geral deste projeto consiste na migração dos dados do cliente que se encontravam em fontes *on-premesis* para um armazenamento único e organizado em *cloud.* Para tal, o desenvolvimento do projeto baseou-se em duas fases, o desenvolvimento de pipelines responsáveis pela extração dos dados e o desenvolvimento de *notebooks* Databricks responsáveis pela transformação dos dados.

De forma a garantir que a fase de extração foi concluída com sucesso e os dados foram extraídos na sua íntegra, o pipeline possui uma atividade de storeProcedure responsável por preencher a tabela de log, como foi anteriormente explicado no capítulo da metodologia.

Com o correto preenchimento da tabela de log podemos de uma forma rápida identificar qual o status da extração de cada tabela, caso tenha sido extraída com sucesso o *status* terá o valor de "Succeeded" caso contrário o valor será "Failed". Para além do *status,* podemos visualizar quantos registos foram carregados em cada extração (*RowCount*), a data e o tempo em que a extração se iniciou e terminou assim como a duração da extração de cada tabela em segundos.

A utilização de uma tabela de log traz bastantes vantagens ao processo, uma vez que, de uma forma prática e visível conseguimos agrupar o resultado de todas as extrações num só local. Caso alguma extração falhe, torna-se fácil identificar qual a tabela que não foi extraída, assim como a sua localização, esquema, modelo e fonte.

ModelName	Destination…	So…	SchemaName	TableName	Status	StartDate	RowCount	SourceR…	EndDate	Duration
YEARLY-01	raw/delivery_an…	EDL	clsd_core	d_calendar	Succeeded	2023-02-22 16:27:54.000	2	qa/edl/md/c…	2023-02-22 16:28:11.370	17
daily	raw/delivery_an…	EDL	clsd_core	clsd_core.pkg_wareh…	Succeeded	2022-11-04 11:06:15.000	478	qa/edl/md/c…	2022-11-04 11:10:22.607	247
daily	raw/delivery_an…	EDL	clsd_core	clsd_core.pkg_wareh…	Succeeded	2022-11-04 16:40:42.000	478	qa/edl/md/c…	2022-11-04 16:44:54.387	252
daily	raw/delivery_an…	EDL	clsd_core	clsd_core.pkg_wareh…	Succeeded	2022-11-08 11:34:14.000	478	qa/edl/md/c…	2022-11-08 11:38:14.990	240
daily	raw/delivery_an…	EDL	clsd_core	pkg_customer_servic…	Succeeded	2022-11-23 11:42:26.000	479	qa/edl/md/c…	2022-11-23 11:45:38.757	192
DAILY-01	raw/delivery_an…	EDL	clsd_core	pkg_warehouse_perfo…	Succeeded	2022-11-29 16:11:02.000	479	qa/edl/md/c…	2022-11-29 16:14:48.620	226
DAILY-01	raw/delivery_an…	EDL	clsd_core	pkg_warehouse_perfo…	Succeeded	2022-11-29 17:17:40.000	479	qa/edl/md/c…	2022-11-29 17:21:31.210	231
DAILY-01	raw/delivery_an…	EDL	clsd_core	pkg_warehouse_perfo…	Succeeded	2022-12-05 12:22:47.000	479	qa/edl/md/c…	2022-12-05 12:26:25.260	218
DAILY-01-DEV	raw/delivery_an…	EDL	clsd_core	pkg_warehouse_perfo…	Succeeded	2022-12-19 15:47:13.000	62	dev/edl/md/…	2022-12-19 15:47:52.743	39
Test_TMZONE	raw/DeliveryAna…	EDL	clsd_core	pkg_warehouse_perfo…	Succeeded	2023-01-10 09:38:23.000	62	dev/edl/md/…	2023-01-10 09:39:28.700	65

Tabela 5 - Tabela de Log – Extração

Este projeto encontra-se em constante evolução e está previsto, em desenvolvimentos futuros, a criação de uma tabela de log materializada em Databricks que registe e monitorize o estado dos dados entre as diferentes camadas de transformação de dados.

Uma vez que, os dados têm de ser mantidos em sigilo, torna-se complicado representar a transformação dos dados entre as diferentes camadas. A criação de um protótipo de tabela de log foi a melhor forma que encontrei para representar os resultados da segunda fase do projeto.

SourceAzureStorageAcount	SourceDeltaTable	TableName	Layer	RowCount
raw/delivery/edl/pck_warehouse	Null	pck_warehouse	Standardized	150
standardized/delivery/edl/pck_warehouse	standardized.pck_warehouse	pck_warehouse	Stage	150
stage/delivery/edl/pck_warehouse	stage.pck_warehouse	pck_warehouse	Persistent	150

(Continua)

StartDate	EndDate	Duration	DestAzureStorageAccount	DestDeltaTable	Status
2023-02-22 16:27:54	2023-02-22 16:28:11	17	standardized/delivery/edl/pck_warehouse	standardized.pck_warehouse	Succeeded
2023-02-22 16:28:15	2023-02-22 16:32:11	247	stage/delivery/edl/pck_warehouse	stage.pck_warehouse	Succeeded
2023-02-22 16:33:54	2023-02-22 16:34:56	62	persistent/delivery/edl/pck_warehouse	persistent.pck_warehouse	Succeeded

Tabela 6 - Protótipo tabela de Log - Transformação

Com este protótipo de tabela de log, podemos monitorizar as transformações de dados entre as diferentes camadas de uma forma dinâmica e agregada, podendo analisar a fonte, camada, número de registos por tabela, data e tempo de começo e término da execução, duração do processo, destino e *status.*

Uma das vantagens de representar as transformações dos dados tendo como base uma tabela de log é a possibilidade de fazer queries à tabela, desta forma, é possível retirar conclusões sobre a performance de cada camada assim como de cada tabela.

Em suma, com a utilização destas duas tabelas de log, uma para o processo de extração e outra para o processo de transformação, conseguimos acompanhar e monitorizar todo o percurso dos dados desde a sua fonte até estarem aptos a ser consumidos por ferramentas de análise e visualização, conseguindo de uma forma mais ágil encontrar erros e vulnerabilidades.

5. CONCLUSÃO E DESAFIOS

Ao longo deste relatório de estágio apresentei argumentos em como o crescimento exponencial de dados traz vantagens para que as indústrias se concentrem mais na adoção de processos digitalmente sofisticados e orientados por dados. Ter grandes quantidades de dados não dá às empresas uma vantagem competitiva, a menos que esses dados possam ser usados para fornecer *insights* valiosos, e esse era exatamente o principal objetivo deste estágio. A construção desta solução de *business intelligence* permitirá que o cliente faça um melhor uso dos seus dados e, consecutivamente, esteja melhor preparado na gestão quotidiana do negócio e para os desafios do futuro.

O estágio na BI4ALL foi uma ótima experiência, tanto profissional quanto pessoalmente. Os objetivos do estágio estavam diretamente alinhados com o programa do mestrado. A maioria das tarefas desenvolvidas estava relacionada a processos ETL, análise de dados e *business intelligence*. A maior parte do conhecimento que foi aprendido durante o programa de mestrado, tanto teórico quanto prático, foi aplicado. O estágio deu-me a capacidade de trabalhar num problema do mundo real e entender a complexidade e a arquitetura por detrás da construção de uma solução, assim como uma compreensão mais profunda da importância da análise de dados e *business intelligence* no mundo moderno.

Como o estágio foi dividido em duas fases distintas, deu-me a possibilidade de trabalhar e melhorar em diferentes vertentes e tecnologias. A primeira parte mais focada na extração dos dados e a segunda mais focada na transformação. Comunicação, organização e trabalho em equipa foram as principais competências interpessoais desenvolvidas; existiram diversas reuniões de planeamento ao longo de todo o projeto e muitas tarefas a serem atribuídas, assim sendo, as competências acima mencionadas foram essenciais para uma boa entrega. As competências técnicas desenvolvidas durante a primeira fase do projeto foram principalmente desenvolvimento de pipelines no Azure Data Factory, uma vez que a *framework* foi construída utilizando estas tecnologias na extração dos dados. A segunda fase me permitiu-me expandir as minhas competências técnicas, usando Databricks e a linguagem Python. Em ambas as fases, a comunicação e a vontade de aprender foram essenciais.

A parte mais desafiante deste projeto, provavelmente, foi o desenvolvimento dos *notebooks* e a forma como as diferentes camadas se relacionavam entre si. Mesmo com alguma experiência em Python e SQL, muito derivado ao mestrado, houve sempre espaço para aprender e consolidar conhecimentos. A arquitetura deste projeto fez-me perceber melhor como funciona todo o fluxo de dados e trabalhos, desde a sua fonte até às ferramentas de visualização. Embora o meu trabalho neste projeto tenha sido de desenvolvedor, tenho a consciência que as tecnologias e arquiteturas utilizadas podem ser aplicadas em diversos setores de negócio.

Em suma, posso dizer que o projeto foi bem-sucedido, tendo sido entregue com todos os requisitos e atendendo aos prazos. O estágio proporcionou-me uma excelente experiência prática, oportunidades e desafios no desenvolvimento de soluções de *business intelligence*, sentindo-me agora mais preparado para novos desafios que estejam para vir.

REFERENCIAS BIBLIOGRÁFICAS

Almeida, M. B. , & O. R. C. (2013). *Business Intelligence (BI): conceitos, tecnologias e aplicações*. Editora Atlas.

Amazon Web Services. (2023, February 7). *What is cloud computing?* https://aws.amazon.com/pt/what-is-cloud-computing/

Armbrust, M. , F. A. , G. R. , J. A. D. , K. R. , K. A. , . . . & Z. M. (2010). A view of cloud computing. *Communications of the ACM*.

Berkeley, E. C. (1964). *Giant Brains or Machines That Think*. John Wiley & Sons.

Brychcy, M. , & W. W. (2016). *Business intelligence and analytics systems* (NY: Springer).

Buyya, R. , & Y. C. S. (2008). Market-oriented cloud computing: vision, hype, and reality for delivering IT services as computing utilities. *Igh Performance Computing and Communications*, 1–12.

Chandarana, G. (2017). *Big Data and Business Intelligence*. Springer.

Chaudhuri, S. , & N. v. (1997). An efficient algorithm for mining association rules in large databases. *ACM SIGMOD Record*, 145–157.

de Haan, J. (2017). The economics of cloud computing. *Journal of Economic Perspectives*, 167–188.

Dutta, S. (2018). Data Lakes: A Brief Introduction. *International Journal of Emerging Technologies in Engineering Research*, 68–71.

Elvis Plesky. (2019). *IaaS vs PaaS vs SaaS – cloud service models compared*. https://www.plesk.com/blog/various/iaas-vs-paas-vs-saas-various-cloud-service-models-compared/

Fayyad, U. , P.-S. G. , & S. P. (1996). From data mining to knowledge discovery in databases. *AI Magazine*, 37–54.

Gandomi, A. , & H. M. (2015). Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 137–144.

Gartner. (2019). *Benefits of cloud computing*. https://www.gartner.com/en/information-technology/glossary/cloud-computing-benefits

Han, J. , K. M. , & P. J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers.

IBM. (2021). *What is big data?* https://developer.ibm.com/blogs/what-is-big-data-more-than-volume-velocity-and-variety/

IBM. (2023). *What is cloud computing?* . https://www.ibm.com/topics/cloud-computing

Inmon, W. H. (2005). *Building the data warehouse* (NJ:John Wiley & Sons).

Khan, A. R. , & T. M. (2015). Security and privacy issues in cloud computing. *Journal of Network and Computer Applications*, 121–128.

Kimball, R. (2002). *The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses*. John Wiley & Sons. https://aws.amazon.com/pt/what-is-cloud-computing/

Kimball, R. , & R. M. (2013). *The data warehouse lifecycle toolkit* (IN:John Wiley & Sons).

Kimball, R. , R. M. , T. W. , & M. J. (2013). *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. John Wiley & Sons.

Kumar, V. , D. B. , & Z. J. (2017). *The data warehouse lifecycle toolkit. Indianapolis* (NY:Springer).

Kwon, O. , & C. S. H. (2017). Big data applications: Recent developments and future prospects. *Journal of Management Information Systems*, 469–509.

Lee, H. , K. Y. , & K. J. (2018). A study on the effectiveness of business intelligence systems: Focusing on the decision-making process. *International Journal of Information Management*, 18–26.

Luhn, H. P. (1958). A Business Intelligence System. *IBM Systems Journal*, 107–117.

Manyika, J. , C. M. , B. B. , B. J. , D. R. , R. C. , & B. A. H. (2011). Big data: The next frontier for innovation, competition, and productivity. *McKinsey Global Institute*.

Meyer, M. , B. M. , & N. B. (2018). Big data analytics in business process management. *Business Process Management Journal*, 390–405.

Microsoft Azure. (2023). *What is cloud computing?* . https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-cloud-computing

O'Reilly, T. (2009). *What is cloud computing?* . O'Reilly Media.

Salesforce. (2023). *What is cloud computing?* https://www.salesforce.com/eu/learning-centre/tech/cloudcomputing/

Scheps, S. (2014). *Business Intelligence for Dummies*. Wiley.

Sharda, R. , D. D. , & T. E. (2018). *Business intelligence: A managerial perspective on analytics* (MA:Pearson Education).

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. *Proceedings of the IEEE Symposium on Visual Languages*, 336–343.

Smith, A. , & M. A. (2019). The impact of business intelligence on organizational performance: A systematic review. *Journal of Business Research*, 341–351.

Srikanth Varma. (2023). *Architecture of Data Warehouse*. https://www.scaler.com/topics/architecture-of-data-warehouse/

Wang, R. Y. , & S. D. M. (1996). Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems*, 5–34.

Zhang, J. , L. X. , & Z. Y. (2017). Business intelligence and analytics: From big data to big impact. *MIS Quarterly*, 1165–1188.

Zhang, Q. , C. L. , & W. Q. (2014). Cloud computing: state-of-the-art and research challenges. *Journal of Internet Services and Information Security*, 7–18.

Zikopoulos, P. , D. T. , & E. C. (2011). *Understanding big data: Analytics for enterprise class hadoop and streaming data* (IN:John Wiley & Sons).