Data Lake

Embora *Data Warehouses* sejam ainda muito relevantes e muito poderosos para dados estruturados, não se pode dizer o mesmo para dados semi-estruturados e não estruturados (KHINE e WANG, 2018; SAWADOGO e DARMONT, 2021). Como a maioria dos dados provenientes de *Big Data* são dados não estruturados (MILOSLAVSKAYA e TOLSTOY, 2016), pode-se concluir que *Data Warehouses* tradicionais não mostram-se alternativas viáveis como fonte de dados analíticos de Big Data. Neste sentido, o conceito de *Data Lake* desafia os tradicionais *Data Warehouses* no armazenamento de dados heterogêneos e complexos (KHINE e WANG, 2018).

As definições para *Data Lake* têm variado ao longo do tempo (KHINE e WANG, 2018; SAWADOGO e DARMONT, 2021). Alguns autores definem *Data Lake* apenas como um novo rótulo para soluções como Apache Hadoop por exemplo, o que, segundo Sawadogo e Darmont (2021) não estaria correto. Outros autores afirmam que *Data Lakes* podem ser utilizados apenas por alguns profissionais específicos, como estatísticos e cientistas de dados, pois os dados armazenados neste tipo de repositório requer certos cuidados que usuários de negócio poderiam negligenciar. Para Sawadogo e Darmont (2021), a definição mais correta para Data Lake é:

“um sistema escalável de armazenamento e análise de dados de qualquer tipo, retido em seu formato nativo e utilizado principalmente por especialistas em dados (estatísticos, cientistas de dados ou analistas) para extração de conhecimento. Suas características incluem:

- um catálogo de dados que reforça a qualidade dos dados;
- ferramentas e políticas de governança de dados;
- acessível por vários tipos de usuários;
- integração de qualquer tipo de dado;
- organização física e lógica;
- escalabilidade em termos de armazenamento e processamento.“.

Para Sawadogo e Darmont (2021), *Data Lakes* não necessariamente precisam ser restritos a estatísticos e cientistas de dados. Na visão destes autores, especialistas de negócio podem acessar Data Lakes através de um *software* de navegação ou de análise de dados apropriados.

De acordo com a Amazon (2022), um *Data Lake* é um repositório centralizado que permite armazenar dados estruturados e não estruturados, em qualquer escala. Os dados podem ser armazenados como estão, sem precisar primeiro estruturá-los. Uma vez armazenados, podem ser executadas diferentes tipos de análises sobre os dados, desde painéis e visualizações até processamento de *big data*, análise em tempo real e aprendizado de máquina.

Para Khine e Wang (2018), *Data Lake* consiste em um repositório em que todos os dados de uma organização, ou seja, dados estruturados, semi estruturados e não estruturados, são armazenados juntos, independentemente dos tipos, formato ou estrutura. Em um *Data Lake*, o entendimento dos dados, da sua estrutura e natureza, é delegada ao consumidor do dado, no momento da recuperação (ou seja, no momento da consulta). Os dados são transformados pelo usuário, ou aplicação, de acordo com o setor da organização, no intuito de adquirir *insights* de negócios.

Alguns *Data Lakes* possuem recursos que possibilitam a criação de uma camada semântica sobre os dados, que permitem a construção de uma camada de contexto, significado e relacionamento entre os dados armazenados. (KHINE e WANG, 2018).

No contexto de um *Data Lake*, todos os dados são primeiramente coletados de suas fontes (Extract - E), depois carregados no *Data Lake* (Load - L), sem modificação de seus formatos originais, e por fim são transformados (Transform - T), de acordo com o interesse do cliente consumidor (KHINE e WANG, 2018). Ou seja, com *Data Lake* temos o acrônimo ELT, ao invés do ETL utilizado no *Data Warehouse*.

Sawadogo e Darmont (2021) apresentam diversas arquiteturas de *Data Lakes*, e concluem que tanto as arquiteturas funcionais quanto as arquiteturas baseadas em maturidade de dados são limitadas, e expõem a necessidade de uma arquitetura que aborde simultaneamente, funcionalidade e maturidade de dados.

A maioria dos *Data Lakes* são construídos sobre o ecossistema Apache Hadoop (HDFS, MapReduce, Spark, etc.), mas como exposto por Sawadogo e Darmont (2021), esta não é a única alternativa, visto que a construção de um *Data Lake* exige diversas partes básicas, tais como:

- **Ingestão de dados:** esta etapa envolve a transferência de dados para o *Data Lake*. Alguns exemplos de ferramentas utilizadas são: Flink, Samza, Flume, Kafka e Sqoop.
- **Armazenamento:** de acordo com Sawadogo e Darmont (2021), existem duas abordagens para armazenamento de dados em *Data Lakes*. A primeira é por meio da utilização de SGBDs tradicionais, como MySQL, PostgreSQL e Oracle. Geralmente, estes SGBDs são utilizados para armazenamento de dados estruturados e possuem também alguns recursos para armazenamento de dados não estruturados. Mas, geralmente, bancos de dados NoSQL são utilizados para o armazenamento de dados estruturados e não estruturados. A segunda forma é por meio do HDFS, utilizado em torno de 75% das vezes para armazenamento de dados em *Data Lakes*. No entanto, de acordo com Sawadogo e Darmont (2021), o HDFS sozinho é geralmente insuficiente para lidar com todos os formatos de dados, especialmente dados estruturados. Deste modo, segundo os autores, o ideal é que seja feita uma combinação do HDFS com bancos de dados relacionais e/ou NoSQL.
- **Processamento de dados:** em *Data Lakes*, o processamento de dados é geralmente feito com o MapReduce ou Apache Spark. O Spark funciona como o MapReduce, mas adota uma abordagem de processamento 100% em memória ao invés de utilizar armazenamento físico para resultados parciais. Deste modo, spark é preferível para processamento de dados em tempo real. O Apache Flink e o Apache Storm são também boas alternativas para o processamento de dados em tempo real. As duas abordagens podem ser utilizadas simultaneamente, com o MapReduce dedicado ao processamento de grandes volumes de dados, e outros motores para o processamento de dados em tempo real.
- **Acesso aos dados:** Em *Data Lakes*, o acesso a dados pode ser feito por SQL para bancos de dados relacionais, JSONiq para MongoDB, XQuery para bancos de dados XML ou SPARQL para recursos RDF. No entanto, isto não permite a execução de consultas sobre bancos de dados heterogêneos, visto que, geralmente, *Data Lakes* são construídos sobre

uma arquitetura heterogênea (SAWADOGO e DARMONT, 2021). Deste modo, uma alternativa consiste na adoção de soluções capazes de realizar consultas em múltiplas fontes simultaneamente. Por exemplo, Spark SQL, ou SQL++ podem ser utilizados para a recuperação tanto de dados em SGBDs relacionais quanto dados semi estruturados no formato JSON. Outro exemplo é o Apache Drill, que permite consultas e junções sobre dados provenientes de múltiplas fontes. Usuários de negócio têm utilizado ferramentas de visualização amigáveis para visualizar dados em Data Lakes, como o Microsoft Power BI e o Tableau.

Combinando *Data Lakes* com *Data Warehouses*

Sawadogo e Darmont (2021) citam duas abordagens de combinação de *Data Lakes* com Data Warehouses. A primeira utiliza o *Data Lake* como fonte de dados para o DW, para onde os dados susceptíveis ao rigor do processo de ETL são enviados, enquanto os dados remanescentes ficam disponíveis para análises complementares. A segunda abordagem considera a construção de um DW dentro do *Data Lake*. Algo que, segundo Sawadogo e Darmont (2021), seria possível com a abordagem de subdivisão do *Data Lake* em *data ponds*, proposta por Inmon (2016). *Data ponds* estruturados a partir de aplicações operacionais poderiam ser considerados equivalentes a *Data Warehouses* (INMON, 2016).

Oreščanin e Hlupić (2021) mencionam uma combinação entre *Data Wharesouse* e *Data Lake* que envolve a migração de dados históricos de um DW para um *Data Lake*. De acordo com os autores, nesta configuração, o DW teria a função de armazenar os dados (fatos) mais recentes, e, à medida em que o DW for aumentando de tamanho, os dados mais antigos, e portanto menos acessados, poderiam ser migrados para a *Foundation Area* de um *Data Lake*. Deste moto, o DW ganharia performance de consulta e espaço em disco. Nesta configuração, os dados armazenados no *Data Lake* continuariam acessíveis e prontos para serem analisados juntamente com os dados no DW.

Maturidade e segurança

Caso o *Data Lake* não seja gerenciado adequadamente, este pode deteriorar-se a tal ponto de tornar-se o que é chamado de data swamp (pantano de dados) (KHINE e WANG, 2018; OREŠČANIN e HLUPIĆ, 2021). Ninguém sabe exatamente o que será incluído em um *Data Lake*, não havendo procedimentos para prevenção de inclusão de dados inconsistentes ou repetidos, por exemplo. Se ninguém sabe exatamente o que há no *Data Lake* até a extração, dados corrompidos podem ser utilizados equivocadamente em tomadas de decisão, fazendo com que o erro seja descoberto tarde demais (KHINE e WANG, 2018). De acordo com Sawadogo e Darmont (2021), a gestão de metadados é fundamental para impedir que o *Data Lake* se torne um data swamp.

Neste mesmo sentido, Khine e Wang (2018) afirmam que a gestão de metadados é fundamental em um *Data Lake*. Como a forma de armazenamento não possui schemas pré-definidos (*schema-on-ready*), como em um *Data Warehouse* (que é schema-on-write), o *Data Lake* precisa fornecer metadados aos clientes durante o processo de análise e realização de consultas. Os metadados precisam ser adicionados no momento de armazenamento dos dados. Deste modo, a definição de uma arquitetura coerente, juntamente com a gestão de metadados, são cruciais para a qualidade de dados.

Begoli, Goethert e Knight (2021) afirmam que *Data Lakes* não impõem explicitamente um *schema* para armazenamento de dados. Este aspecto, em um cenário de crescimento exponencial de dados, provenientes de inúmeras fontes, pode transformar o *Data Lake* em um *Data Swamp*, ou seja, em um pântano cheio de detritos de dados.

De acordo com Sawadogo e Darmont (2021), temas como governança de dados e segurança em *Data Lakes* são atualmente pouco abordados pela literatura. De acordo com Khine e Wang (2018), *Data Warehouses* podem garantir governança, performance, segurança e controle de acesso aos dados, enquanto *Data Lakes* não podem garantir, de forma plena, nenhum destes aspectos. Neste sentido, Khine e Wang (2018) afirmam que gerar valor com *Data Lakes* é ainda algo muito difícil.

Revisão #22
Criado 2022-03-07 15:28:09 UTC por FLAVIO LOPES DE MORAIS
Atualizado: 2022-12-14 09:02:06 UTC por FLAVIO LOPES DE MORAIS