



Soluções para Armazenamento de Dados

Bootcamp Arquiteto(a) de Soluções

Marcelo Leite

2022

Soluções para Armazenamento de Dados

Bootcamp Arquiteto(a) de Soluções

Marcelo Leite

© Copyright do Instituto de Gestão e Tecnologia da Informação.

Todos os direitos reservados.

Sumário

Capítulo 1. Conceitos.....	5
1.1 Os diferentes tipos de dados.....	5
1.2 Tipos de servidores de armazenamento de dados.....	9
1.3 Serviços de armazenamento de dados na nuvem	10
Capítulo 2. Bancos de dados transacionais	12
2.1 O que são bancos de dados SQL, NoSQL ou NewSQL?	12
2.2 Bancos de dados em nuvem	14
2.3 Bancos de dados em tempo real e para cache de dados	15
2.4 Operações em bancos de dados transacionais.....	16
2.5 DataOps	18
Capítulo 3. Bancos de dados Analíticos	19
3.1 Diferenças entre Big Data e DataWarehouse.....	19
3.2 Arquiteturas de dados modernas	19
3.3 Modelos de Arquitetura de dados.....	22
3.4 Organização de um Datalake	29
3.5 Governança e Segurança de dados	32
3.6 Principais soluções de dados modernas	38
Capítulo 4. Outras plataformas de dados	40
4.1 Soluções para extração, ingestão, transformação e análise de dados	40
4.2 Pesquisas Indexadas/Cognitivas.....	41

Referências.....	43
------------------	----

Capítulo 1. Conceitos

Ao longo das últimas décadas, a quantidade de dados gerados por sistemas, aplicativos e dispositivos aumentou consideravelmente. Os dados estão em todos os lugares, disponíveis em diferentes estruturas e formatos. Compreender os dados, como armazenar e explorá-los, revela fatos interessantes e ajuda você a obter insights significativos.

1.1 Os diferentes tipos de dados

Dados são todos os registros gerados por sistemas como dados cadastrais, logs de transações, áudio de chamadas telefônicas, vídeos etc. Processar esses dados para que eles apoiem as tomadas de decisão é o chamado informação. Nos últimos anos, temos observado diversas empresas buscando dados não somente em seus sistemas internos, mas na internet, nas redes sociais e em parceiros de negócios que possam enriquecer suas informações e, conseqüentemente, seus insights.

Podemos classificar os tipos de dados como estruturados, semiestruturados ou não estruturados.

Dados estruturados são dados armazenados de maneira tabular, representados por linhas e colunas em um banco de dados. Os bancos de dados que armazenam tabelas dessa forma são chamados de bancos de dados relacionais (o termo matemático relação refere-se a um conjunto organizado de dados mantidos em formato de tabela). Cada linha de uma tabela tem o mesmo conjunto de colunas. A imagem abaixo traz um exemplo que ilustra uma tabela de um banco de dados de despesas pessoais.

Figura 1 – Representação de uma tabela de dados estruturados

MÊS	CATEGORIA	VALOR
Janeiro	Transporte	R\$ 74,00
Janeiro	Supermercado	R\$ 235,00
Janeiro	Despesas domésticas	R\$ 175,00
Janeiro	Entretenimento	R\$ 100,00
Fevereiro	Transporte	R\$ 115,00
Fevereiro	Supermercado	R\$ 240,00
Fevereiro	Despesas domésticas	R\$ 225,00
Fevereiro	Entretenimento	R\$ 125,00
Março	Transporte	R\$ 90,00
Março	Supermercado	R\$ 260,00
Março	Despesas domésticas	R\$ 200,00
Março	Entretenimento	R\$ 120,00

Fonte: Captura própria

Dados semiestruturados não residem em um banco de dados relacional, mas ainda possuem alguma estrutura. Um dos formatos mais utilizados são os documentos JSON (JavaScript Object Notation). O exemplo a seguir mostra um documento JSON que representa informações dos padrões e termos.

Figura 2 – Representação de dados em um arquivo JSON

```

1  {
2    "patterns": [
3      {
4        "patternName": "pattern",
5        "instances": [
6          {
7            "roles": [
8              {
9                "name": "roleName",
10               "element": "org.example.entity"
11             }
12           ]
13         }
14       ]
15     },
16     "terms": [
17       {
18         "entityName": "org.example.entity",
19         "termsWithCounter": {
20           "term1": 1,
21           "term2": 4,
22           "termN": 1
23         }
24       }
25     ]
26   }
27 }
```

Fonte: Captura própria

Também há outros tipos de dados semiestruturados. Os exemplos incluem repositório de chave-valor e bancos de dados de grafo.

Um banco de dados de valor-chave armazena matrizes associativas. Nessas matrizes, uma chave serve como um identificador exclusivo para recuperar um valor específico. Esses valores podem ser qualquer coisa, desde um número ou cadeia de caracteres até um objeto complexo, como um arquivo JSON.

Muito utilizado no desenvolvimento de microserviços modernos, um banco de dados de chave-valor armazena dados como uma coleção única, sem estrutura ou relação. Isso o torna diferente de um banco de dados relacional, em que as tabelas são compostas por linhas e colunas com tipos de dados predefinidos.

A imagem abaixo mostra um exemplo de dados de chave-valor:

Figura 3 – Representação de dados tipo chave-valor

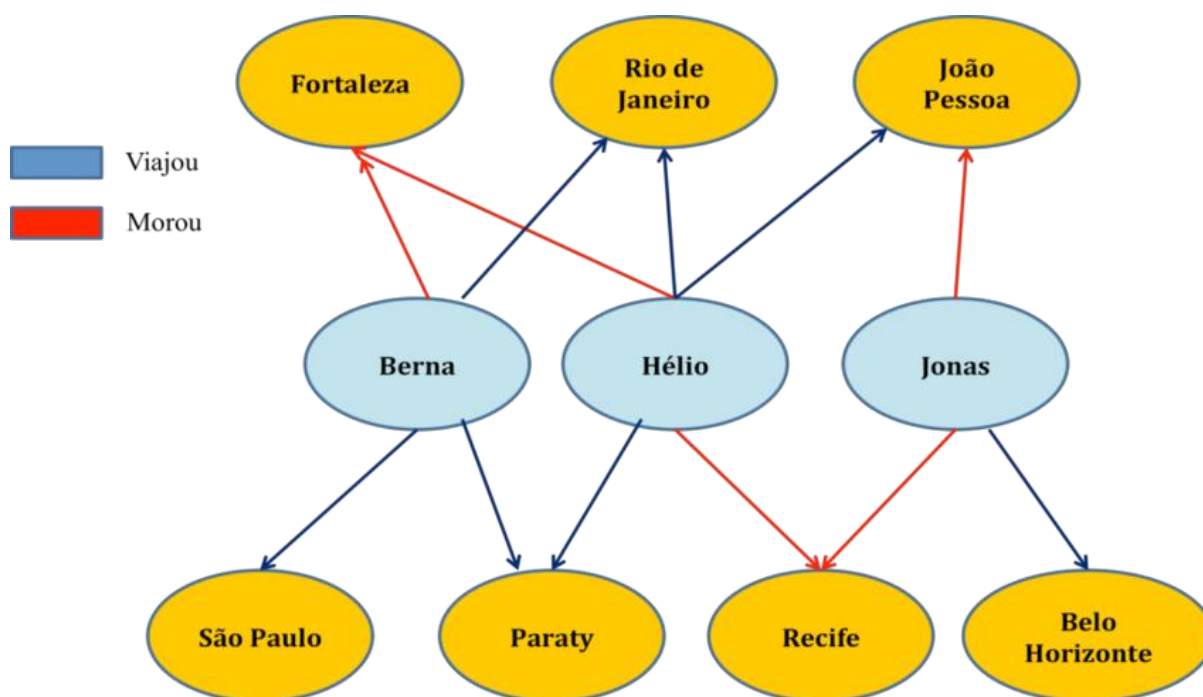
ID da Pessoa	Tipo	Atributo	Atributo	Atributo	Atributo
1	ID do Presidente	Washington	Adams	Jefferson	Madison
2	ID do Monarca	Henrique VIII	Ricardo III	Elizabeth I	

Fonte: Captura própria

Outro formato de organização de dados são os chamados bancos de dados grafos. Eles podem ser utilizados para armazenar e consultar informações sobre relações complexas. Um exemplo de implementação de grafos são as timelines das redes sociais, onde a sua lista de contatos e seus interesses são correlacionados sendo transformados em uma série de resultados de consultas apresentados na tela.

Uma base grafo contém “nós” (informações sobre objetos) e bordas (informações sobre as relações entre os objetos). A imagem abaixo mostra um exemplo de como você pode estruturar os dados em um banco de dados de grafo:

Figura 4 – Representação de dados em formato grafo



Fonte: Captura própria

Nem todos os dados são estruturados ou até mesmo semiestruturados. Por exemplo, arquivos de áudio e vídeo e arquivos de dados binários podem não ter uma estrutura específica. Eles são chamados de dados não estruturados.

São fontes importantíssimas de informação, mas, na maioria dos casos, é necessário estruturar os dados de áudios e vídeos em texto extraído para tabelas para, então, utilizá-los.

O que é um banco de dados transacional?

Um banco de dados transacional é onde os sistemas corporativos registram suas transações. Uma transação pode ser em um controle de estoque, como uma venda que faz com que um produto saia do estoque; financeira, como a movimentação de dinheiro entre contas em um sistema bancário; ou pode fazer parte de um sistema de varejo, como acompanhar pagamentos de bens e serviços efetuados pelos clientes. Toda transação gera uma coleção de dados, normalmente

já armazenada em formato tabular. O trabalho executado por sistemas transacionais é conhecido como OLTP (Processamento de Transações Online).

O que é uma base de dados analítica?

Uma base analítica é organizada para arranjar os dados de diferentes fontes e disponibilizar formatos de consulta para usuários que transformarão esses dados em informação para tomada de decisões.

1.2 Tipos de servidores de armazenamento de dados

O Storage é um equipamento que armazena os dados.

Divididos em três grandes grupos, os Storages são classificados como DAS, NAS e SAN. Diferenciados pela forma de conexão com seu computador host ou servidor, as unidades de armazenamento possuem diferentes finalidades e recursos.

DAS (*Direct Attached Storage* ou Armazenamento anexado direto) é o tipo de Storage conectado diretamente ao servidor ou computador. Esse tipo de Storage é muito comum em notebooks e computadores pessoais, mas é menos utilizado em servidores nas ofertas de computação em nuvem. São exemplos de DAS desde Pen Drives, HDs, SSDs e outras unidades diretamente conectadas ao computador de processamento.

NAS (*Network Attached Storage* ou Armazenamento anexado via rede) surgiu de maneira mais abrangente em Datacenters, onde o Storage era centralizado e compartilhado por vários servidores e computadores na rede. É utilizado, por exemplo, como servidores de arquivos corporativos.

Já o SAN (*Storage Area Network*) é uma rede de armazenamento dedicada e composta por servidores e Storages interligados através de conexões IP (iSCSI) ou Fibre Channel (FC). Buscando simplificar e consolidar a produção de dados, uma SAN centraliza e melhora o gerenciamento das informações, proporcionando mais segurança e velocidade no acesso aos dados.

1.3 Serviços de armazenamento de dados na nuvem

Com a modernização dos Datacenters para uso de provedores de nuvem pública, esses conceitos foram encapsulados em serviços de armazenamento PaaS, que serão abordados abaixo:

Armazenamento de Discos

Armazenamento em bloco durável e de alto desempenho para Máquinas Virtuais. Pode ser HD, SSD e outras variações que aumentam ainda mais a performance do disco, mas sempre como um disco tradicional de armazenamento de dados.

Armazenamento de Blobs

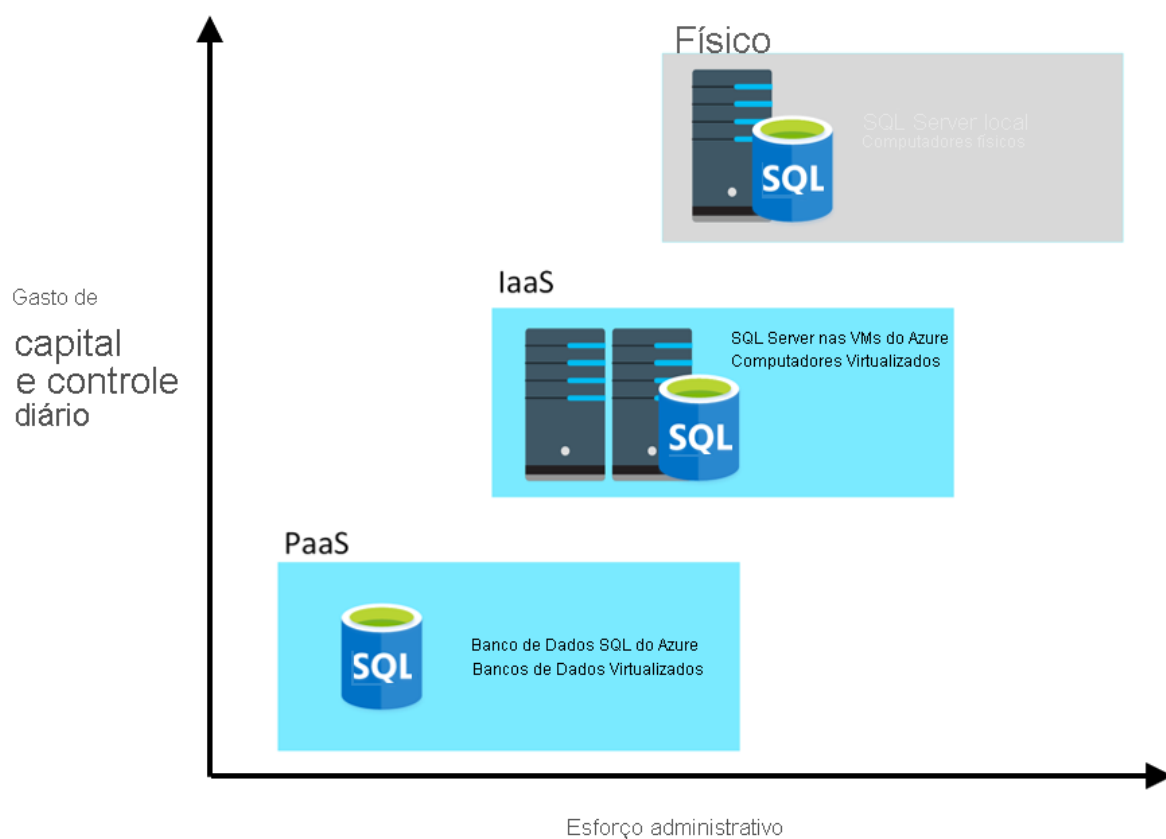
Armazenamento de objetos altamente escalonável e seguro para cargas de trabalho, arquivos, Data Lakes, computação de alto desempenho e aprendizado de máquina nativos de nuvem

Armazenamento Files

Compartilhamentos de arquivo em nuvem de nível empresarial simples, seguro e sem servidor. Várias opções para uso pessoal também como Dropbox, Google Drive e Microsoft OneDrive.

Para projetos que nascem em nuvem, os bancos de dados PaaS trazem uma série de benefícios que os fazem ser boas opções. Os serviços PaaS cuidam da configuração básica, do gerenciamento diário, das atualizações de software, da segurança dos bancos de dados que hospeda e, muitas vezes, até de alta disponibilidade, backup e outros temas importantes para a continuidade dos negócios.

Figura 5 – Diferentes tipos de ofertas de bancos de dados em nuvem



Fonte: Microsoft

A imagem acima da documentação do Microsoft Azure demonstra as diferentes ofertas de bancos de dados, sendo que o PaaS (*Platform As a Service*) seria o banco com menor esforço administrativo devido às operações automatizadas da nuvem.

Capítulo 2. Bancos de dados transacionais

2.1 O que são bancos de dados SQL, NoSQL ou NewSQL?

A sigla SQL significa “Structured Query Language”, ou seja, “Linguagem de Consulta Estruturada”. Ela é a linguagem de consulta a bancos de dados mais conhecida e usada mundialmente.

Existem comandos SQL para criar, alterar, gerenciar, consultar, entre outras operações em seus dados no banco de dados.

NoSQL (*Not Only SQL*) é o termo utilizado para banco de dados não relacionais onde, geralmente, a linguagem SQL não é a linguagem de consulta.

NoSQL foi criado para atender a demandas específicas de acesso e controle de dados, provendo, para esses casos, uma performance melhor e uma escalabilidade ideal para suprir necessidades onde os bancos relacionais tradicionais não eram eficazes.

Os 5 tipos de bancos de dados NoSQL mais utilizados no mercado são:

Tipo Documento

Os dados são armazenados como documentos. Os documentos podem ser descritos como dados no formato de chave-valor, como o padrão JSON.

Um exemplo de banco de dados neste formato é o MongoDB.

Tipo Colunas

Os dados são armazenados em linhas particulares de tabela no disco, podendo suportar várias linhas e colunas, além de permitir subcolunas.

Um banco de dados baseado em Coluna é o Cassandra.

Tipo Grafo

Os dados são armazenados na forma de grafos (vértices e arestas).

O Neo4j e o Gremling são exemplos de bancos de dados grafos.

Chave-valor

Essa família de bancos NoSQL é a que aguenta a maior carga de dados, pois o conceito dela é que um determinado valor seja acessado através de uma chave identificadora única.

Um exemplo é o banco de dados Riak.

Existem, ainda, os chamados bancos de dados **NewSQL**, que buscam promover a mesma melhoria de desempenho e escalabilidade dos sistemas NoSQL, não abrindo mão dos benefícios dos bancos de dados tradicionais da linguagem SQL, nem das propriedades ACID.

Mike Stonebreaker, fundador do VoltDB (um dos bancos de dados desse novo modelo), destaca que a principal vantagem dos bancos de dados NewSQL é que eles proporcionam consultas em tempo real, além de maior capacidade de processamento.

Segundo Mike, há um custo grande em não usar SQL, no caso de bancos NoSQL, sendo exigido trabalho excessivo dos desenvolvedores para compensar sua ausência.

Diferente dos sistemas de gerenciamento de bancos de dados tradicionais, que eram considerados soluções para qualquer tipo de aplicação, os NewSQL utilizam uma estratégia diferente, onde cada novo sistema desenvolvido visa atender a uma necessidade específica em um projeto.

As características de um banco de dados **NewSQL**:

- ✓ Controle de concorrência não bloqueante, para que as leituras e escritas não causem conflitos entre si;
- ✓ Linguagem SQL como meio de interação entre o *SGBD* e a aplicação;
- ✓ Suporte para transações *ACID*;
- ✓ Arquitetura que forneça um maior desempenho por nó de processamento;












- ✓ Arquitetura escalável, com memória distribuída e com capacidade de funcionar em um aglomerado com um grande número de nós.

2.2 Bancos de dados em nuvem

Cada provedor de nuvem pública tem ofertas diferentes de bancos de dados gerenciados. Algumas dessas ofertas são bancos de dados nativos em nuvem, que foram projetados com recursos de automação de gerenciamento e escalabilidade, enquanto outros formatos são bancos de dados tradicionais, como Microsoft SQL Server e MySQL, em plataformas gerenciadas pelos provedores.

O mais importante como arquiteto de soluções é identificar o caso de uso para determinado banco de dados e, então, avaliar as diferentes opções no provedor de nuvem que o projeto rodará.

Figura 6 – Diferentes casos de usos de dados na AWS

Tipo de banco de dados	Casos de uso	Serviço da AWS
Relacional	Aplicações tradicionais, ERP, CRM, comércio eletrônico	 Amazon Aurora  Amazon RDS  Amazon Redshift
Chave-valor	Aplicações da Web de alto tráfego, sistemas de comércio eletrônico, aplicativos de jogos	 Amazon DynamoDB
Em memória	Armazenamento em cache, gerenciamento de sessões, placares de jogos, aplicações geoespaciais	 Amazon ElastiCache for Memcached  Amazon ElastiCache for Redis
Documento	Gerenciamento de conteúdo, catálogos, perfis de usuários	 Amazon DocumentDB (com compatibilidade com o MongoDB)
Coluna ampla	Aplicativos industriais de grande escala para manutenção de equipamentos, gerenciamento de frota e otimização de rotas	 Amazon Keyspaces (para Apache Cassandra)
Grafo	Deteção de fraudes, redes sociais, mecanismos de recomendação	 Amazon Neptune
Séries temporais	Aplicativos IoT, DevOps, telemetria industrial	 Amazon Timestream
Ledger	Sistemas de registro, cadeia de suprimentos, registros, transações bancárias	 Amazon QLDB

Fonte: AWS

É comum os provedores de nuvem documentarem casos de aplicação de seus diferentes tipos de estruturas de dados. A imagem acima, proveniente da documentação da AWS, mostra os bancos de dados possíveis por caso de uso.

Conforme o exemplo acima, os provedores de Cloud documentam casos de uso para utilização de cada formato de banco de dados.

[Bancos de dados do Google Cloud](#)

[AWS banco de dados na nuvem - Amazon Web Services](#)

[Bancos de dados do Azure – Tipos de bancos de dados do Azure | Microsoft Azure](#)

No caso de sistemas multicloud, onde a mesma aplicação rodará em um provedor de nuvem e utilizará um segundo provedor de nuvem como contingência, atente-se aos bancos de dados que possam estar rodando nos dois provedores, com ferramentas de sincronismo de dados que garantam a contingência de um desastre.

2.3 Bancos de dados em tempo real e para cache de dados

Com a modernização dos padrões de desenvolvimento de software, novas necessidades de armazenamento de dados foram tratadas pela comunidade Open Source, especializando bancos de dados para atendimento a microsserviços específicos. Dois tipos de bancos de dados bastante utilizados em softwares modernos são bancos de dados em tempo real e bancos de dados para cache de dados.

Um **banco de dados em tempo real** se refere a uma configuração de banco de dados que depende do processamento em tempo real para gerenciar cargas de trabalho com estados em constante mudança. É diferente de um banco de dados convencional que geralmente tem a tarefa de lidar com dados persistentes sem ser afetado pelo tempo.

Os bancos de dados em tempo real podem ser utilizados para vários casos de uso, como registros médicos, contabilidade, legislação, controle de processos, análise de dados, multimídia e sistemas de reserva.

São exemplos de bancos de dados em tempo real o PubNub, o Back4App, Firebase e o Pusher.

Um **banco de dados cache** é um banco de dados de alta velocidade, que guarda um conjunto de dados, geralmente de forma temporária, para que futuras solicitações referentes a esses dados sejam atendidas de modo mais rápido do que direto ao local de armazenamento principal de dados. Normalmente, as bases de dados de cache são implementadas como camada de acesso antes do banco de dados transacional. O armazenamento em cache permite reutilizar com eficiência dados recuperados ou computados anteriormente.

Os dados em um cache geralmente são armazenados no hardware de acesso rápido, como a memória RAM de servidores. O principal objetivo de um cache é aumentar a performance da recuperação de dados ao reduzir a necessidade de acessar a camada subjacente mais lenta de armazenamento em uma consulta.

Exemplos de bancos de dados de cache são o Redis Cache e o Amazon ElastiCache.

2.4 Operações em bancos de dados transacionais

Os Bancos de Dados PaaS em nuvem trazem uma série de operações administrativas importantes para a manutenção e continuidade do uso de dados como a atualização automática do software de banco de dados e a aplicação de patches de segurança para garantir que você sempre esteja executando a versão mais recente e mais segura do serviço.

Os recursos de escalabilidade de bancos de dados PaaS em nuvem garantem que você possa aumentar os recursos disponíveis para armazenar e processar dados sem precisar executar uma atualização manual cara.

Os serviços podem fornecer garantias de alta disponibilidade “build-in”, isto é, já configurada no serviço para garantir que seus bancos de dados estejam disponíveis com base em SLA (*Service Level Agreement*) que podem chegar até 99,9999% do tempo de uso.

Outro fator interessante é que os bancos de dados PaaS normalmente contam com recursos de backup com restauração pontual, o que permite recuperar um banco de dados para o estado em que estava em qualquer ponto no passado.

Ainda é possível replicar bancos de dados em regiões de nuvem diferentes para fornecer garantia adicional e recuperação de desastre.

Também presentes em ofertas PaaS de bancos de dados, camadas de segurança, auditoria e criptografia podem ser consideradas em projetos para elevar ainda mais a confiança e a continuidade da aplicação. Como esses recursos não estão presentes em todas as ofertas de bancos de dados PaaS, é importante avaliar quais deles são recursos para o seu projeto:

A **proteção avançada contra ameaças** fornece recursos de segurança avançados, como avaliações de vulnerabilidade, para ajudar a detectar e corrigir possíveis problemas de segurança com seus bancos de dados. A proteção contra ameaças detecta atividades que indicam tentativas incomuns de acesso ao banco de dados. Algumas proteções têm monitoramento contínuo do banco de dados com relação a atividades suspeitas e fornecem alertas de segurança imediatos sobre possíveis vulnerabilidades, ataques do tipo “SQL Injection” e padrões de acesso anormal ao banco de dados.

Recursos de **auditoria** rastreiam eventos ocorridos no banco de dados e os gravam em um log de auditoria. A auditoria pode ajudar a manter uma conformidade regulatória, a entender a atividade do banco de dados e a obter informações sobre discrepâncias e anomalias que poderiam indicar preocupações de negócios ou suspeitas de violações de segurança.

Outro recurso importante é de **criptografia**, sendo que existem dois tipos de habilitação: em movimento (entre as transações com o banco de dados) e em repouso (quando os dados estão armazenados sem uma operação no banco de dados).

2.5 DataOps

DataOps é a derivação da metodologia DevOps aplicada a ambientes de dados. Segundo Michele Goetz, vice-presidente e analista principal da Forrester, é a capacidade de habilitar soluções, desenvolver produtos de dados e ativar dados para valor de negócios em todas as camadas de tecnologia, da infraestrutura à experiência. Em outras palavras, é a técnica que visa atender ao trabalho de dados de forma estruturada, alinhada às necessidades de negócios e o mais automatizada possível.

Para implementar uma estratégia de DataOps eficiente, é necessário um conhecimento básico sobre como funciona a metodologia DevOps no desenvolvimento de software e realizar o paralelo aos ambientes de dados.

Normalmente, citamos “DataOps” quando nos relacionamos com ambientes Analíticos, pois bancos de dados transacionais ficam, muitas vezes, encapsulados em estratégias de automação e governança de DevOps, junto às camadas de aplicação.

As principais áreas de especialização para práticas de DataOps incluem:

- Bancos de dados;
- Integração;
- Dados para processar orquestração;
- Implantação de política de dados;
- Integração de dados e modelo;
- Segurança de dados e controles de privacidade.

Capítulo 3. Bancos de dados Analíticos

3.1 Diferenças entre Big Data e DataWarehouse

Big Data: Big Data refere-se aos dados que estão em grande volume e possuem conjuntos de dados complexos. Essa grande quantidade de dados pode ser estruturada, semiestruturada ou não estruturada e não pode ser processada por softwares e bancos de dados tradicionais de processamento de dados. Várias operações, como análise, manipulação, alterações etc., são realizadas nos dados e, em seguida, são utilizadas pelas empresas para uma tomada de decisão inteligente. Big Data é um ativo muito poderoso no mundo de hoje. Big Data também pode ser usado para resolver problemas de negócios, fornecendo tomada de decisão inteligente.

Data Warehouse: Data Warehouse é a centralização de dados de várias fontes heterogêneas. É o principal componente do sistema de inteligência de negócios, onde a análise e o gerenciamento de dados são feitos, o que é usado posteriormente para melhorar a tomada de decisões. Envolve o processo de extração, carregamento e transformação para fornecer os dados para análise. Os armazéns de dados também são usados para realizar consultas em uma grande quantidade de dados. Ele usa dados de vários bancos de dados relacionais e arquivos de log do aplicativo.

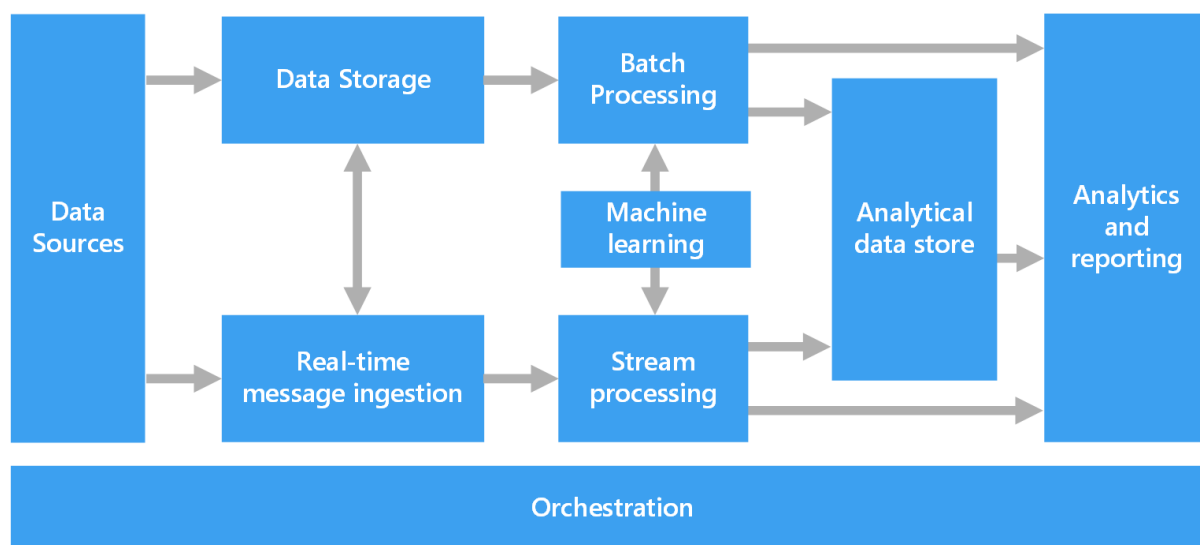
3.2 Arquiteturas de dados modernas

Devido às limitações de escalabilidade dos ambientes de Data Warehouse e à complexidade na administração de ambientes Big Data tradicionais, novas arquiteturas analíticas modernas de dados que derivam o melhor das duas técnicas.

Componentes de uma arquitetura moderna de dados

O diagrama a seguir mostra os componentes lógicos que se inserem em uma arquitetura moderna de dados. As soluções individuais podem não conter todos os itens neste diagrama.

Figura 7 – Arquitetura conceitual analítica de dados



Fonte: Microsoft

A maioria das arquiteturas de dados modernas inclui alguns ou todos os seguintes componentes:

Fontes de dados (Data Sources). As fontes de dados são quaisquer repositórios onde os dados são originados. Alguns exemplos são:

- 1) Armazenamentos de dados de aplicativo, como bancos de dados relacionais.
- 2) Arquivos produzidos por aplicativos, como arquivos de log de transação da aplicação.
- 3) Fontes de dados em tempo real, como dispositivos IoT e telemetria GPS de um aplicativo de celular.

Armazenamento de dados (Data Storage): Os dados das fontes de dados são copiados para um repositório estruturado, normalmente com uma organização hierárquica de dados similar a pastas de arquivos. Esse tipo de repositório em uma arquitetura moderna é chamado de **Data Lake**.

Processamento em lote (Batch Processing): Como os conjuntos de dados são muito grandes, geralmente, uma solução de dados precisa processar arquivos de

dados usando lotes de processamento para filtrar, agregar e, de outro modo, preparar os dados para análise. Normalmente, esses trabalhos envolvem ler arquivos de origem, processá-los e gravar a saída para novos arquivos, utilizando o Data Lake como o repositório de armazenamento em todos estes estados.

Ingestão em tempo real (*Real time message ingestion*): Dependendo do caso de uso, é necessária uma ingestão de dados próximo ao tempo real. Para atender a esse requisito, utilizamos uma técnica baseada em eventos, que são ingeridos e processados antes mesmo de serem armazenados no Data Lake.

Machine Learning: Em arquiteturas modernas, é necessário prever cenários de utilização de técnicas de Machine Learning para enriquecer as análises de dados. Modelos estatísticos de Machine Learning podem ser treinados e processados utilizando dados em lote (Batch) ou near real-time (Stream) conforme o case implementado.

Processamento de fluxo (*Stream Processing*): Depois de capturar mensagens em tempo real, a solução precisa processá-las filtrando, agregando e preparando os dados para análise. Os dados de fluxo processados podem ser apresentados em Dashboards e, também, gravados no Data Lake.

Armazenamento de dados analíticos (*Analytical Data Store*): Com os dados armazenados no Data Lake, muitas vezes é necessário criar modelos de consumo de dados que façam com que esses dados tenham sentido para os seus consumidores.

Esses modelos são implementados em uma camada Gold do Data Lake (baseado em Delta Lake) ou baseado em tabelas relacionais em uma plataforma de Data Warehouse.

Com esse formato estruturado nos dados, eles se tornam informação para serem consultados com ferramentas analíticas.

Análise e relatórios (*Analytics & Reporting*): O grande resultado do ambiente analítico é o consumo dos dados. Normalmente, são utilizadas ferramentas

de relatórios chamadas de ferramentas de BI (*Business Intelligence*) e ferramentas de Analytics.

Orquestração (*Orchestration*): Para a camada de orquestração, é importante considerar as operações que governam toda a arquitetura de dados com componentes que criem pipelines de carga de dados, garantem automação no fluxo de dados e mantenham o controle da linhagem de dados (rastreabilidade do dado desde a fonte de dados até o consumo do mesmo, com o registro completo de todas as transformações realizadas).

3.3 Modelos de Arquitetura de dados

Arquitetura Lambda

Ao trabalhar com conjuntos de dados muito grandes, pode levar muito tempo para executar a classificação de consultas de que os clientes precisam. Essas consultas não podem ser executadas em tempo real e geralmente exigem algoritmos como [MapReduce](#), que operam em paralelo em todo o conjunto de dados. Os resultados são, então, armazenados separadamente dos dados brutos e usados para consulta.

Uma desvantagem dessa abordagem é que ela introduz latência — se o processamento levar algumas horas, uma consulta poderá retornar resultados de várias horas atrás. O ideal é que você obtenha alguns resultados em tempo real (talvez com alguma perda de precisão) e combine esses resultados com os resultados da análise de lote.

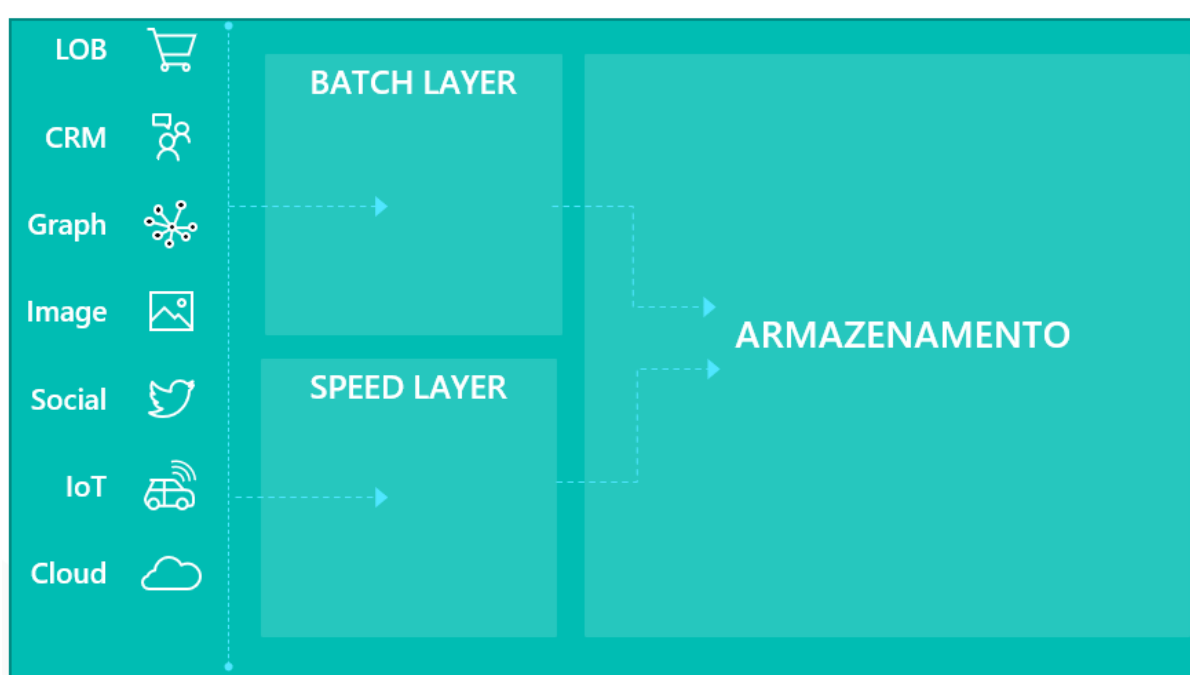
A **arquitetura lambda**, primeiramente proposta por Nathan Marz, resolve esse problema criando dois caminhos para o fluxo de dados. Todos os dados recebidos pelo sistema passam por esses dois caminhos:

Uma **camada de lote** (caminho frio) armazena todos os dados de entrada em sua forma bruta e executa o processamento em lotes nos dados. O resultado desse processamento é armazenado como uma **exibição de lote**.

Uma **camada de velocidade** (caminho quente) analisa os dados em tempo real. Essa camada foi projetada para baixa latência em detrimento da precisão.

A camada de lote alimenta uma **camada de serviço** que indexa a exibição de lote para uma consulta eficiente. A camada de velocidade atualiza a camada de serviço com atualizações incrementais de acordo com os dados mais recentes.

Figura 8 – Arquitetura Lambda



Fonte: Captura Própria

Os dados que fluem para o caminho quente são restritos por requisitos de latência impostos pela camada de velocidade, de modo que ela possa ser processada o mais rapidamente possível. Geralmente, isso exige uma desvantagem de algum nível de precisão em favor dos dados que estão prontos o mais rapidamente possível. Por exemplo, considere um cenário de IoT em que muitos sensores de temperatura enviam dados telemétricos. A camada de velocidade pode ser usada para processar uma janela de tempo deslizante dos dados de entrada.

Os dados que fluem para o caminho frio, por outro lado, não estão sujeitos aos mesmos requisitos de baixa latência. Isso permite uma computação de alta precisão em conjuntos de dados grandes, o que pode ser muito demorado.

Em última análise, os caminhos quente e frio convergem no aplicativo cliente de análise. Se o cliente precisar exibir dados em tempo hábil, mas potencialmente menos precisos em tempo real, ele adquirirá seu resultado do caminho quente. Caso contrário, ele selecionará resultados do caminho frio para exibir dados em menos tempo hábil, mas mais precisos. Em outras palavras, o caminho quente contém dados para uma janela relativamente pequena de tempo, após o qual os resultados podem ser atualizados com os dados mais precisos do caminho frio.

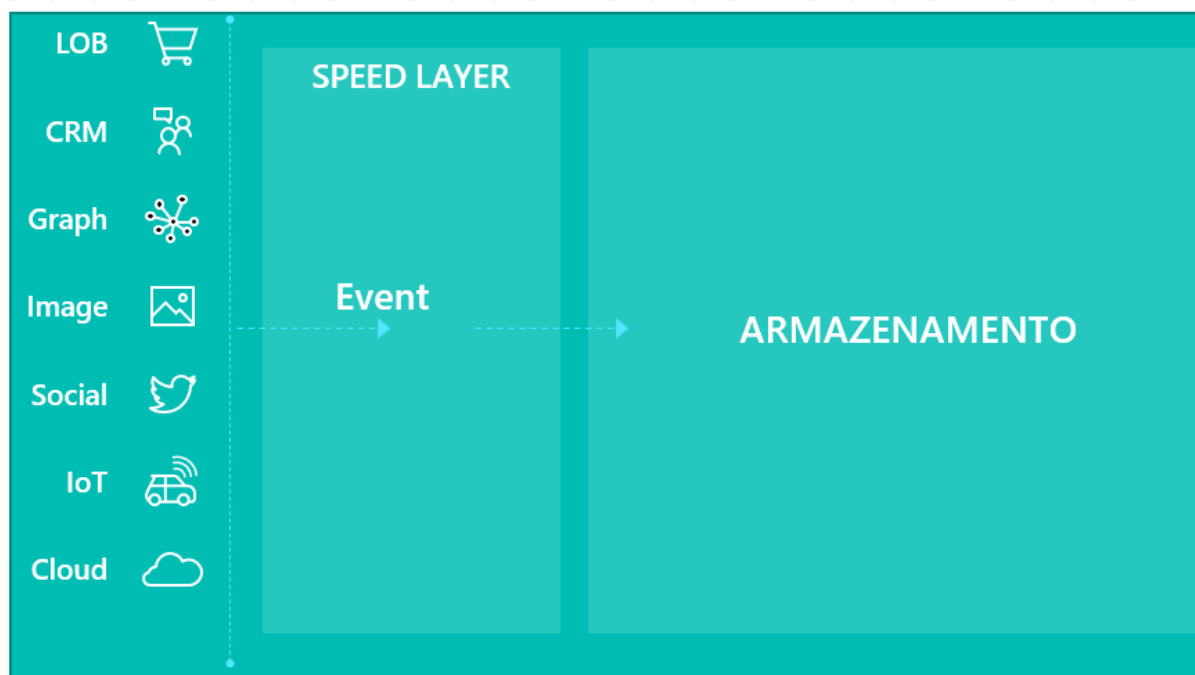
Os dados brutos armazenados na camada de lote são imutáveis. Os dados de entrada sempre são acrescentados aos dados existentes, e os dados anteriores nunca são substituídos. As alterações no valor de um dado específico são armazenadas como um novo registro de evento com carimbo de data/hora. Isso permite o recálculo em qualquer ponto no tempo no histórico dos dados coletados. A capacidade de recalcular a exibição de lote dos dados brutos originais é importante, pois permite que novas exibições sejam criadas conforme o sistema evolui.

Arquitetura Kappa

Uma desvantagem da arquitetura de lambda é sua complexidade. A lógica de processamento aparece em dois lugares diferentes – os caminhos frio e quente – usando estruturas diferentes. Isso leva a uma lógica de cálculo duplicada e à complexidade de gerenciar a arquitetura para os dois caminhos.

A **arquitetura de kappa** foi proposta por Jay Kreps como uma alternativa à arquitetura de lambda. Ela tem as mesmas metas básicas da arquitetura de lambda, mas com uma diferença importante: todos os dados fluem por um único caminho, usando um sistema de processamento de fluxo.

Figura 9 – Arquitetura Kappa



Fonte: Captura Própria

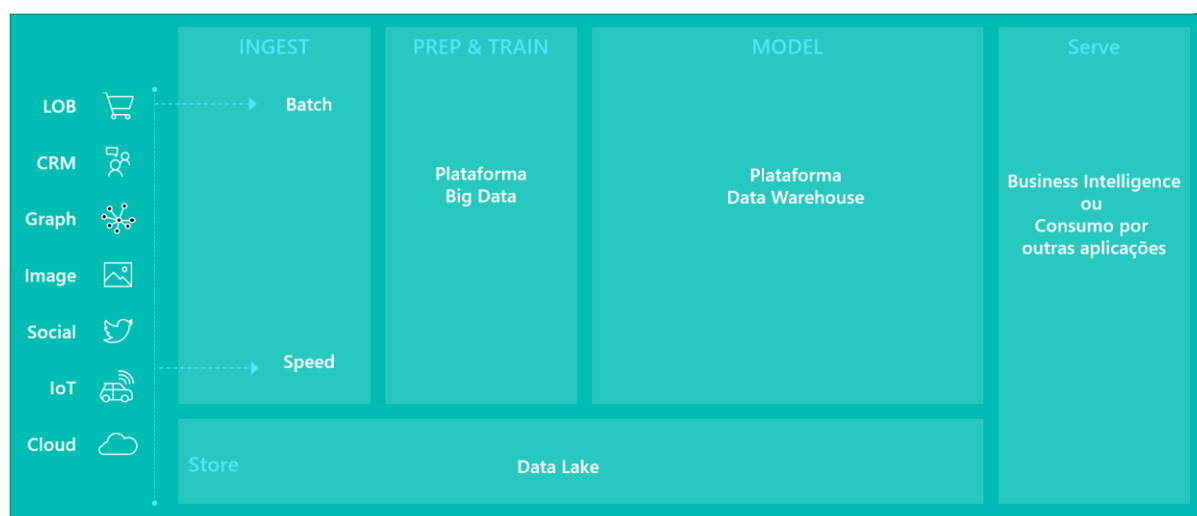
Há algumas semelhanças na camada de lote da arquitetura de lambda, em que os dados do evento são imutáveis e todos eles são coletados, em vez de um subconjunto. Os dados são ingeridos como um fluxo de eventos em um log unificado distribuído e tolerante a falhas. Esses eventos são ordenados e o estado atual de um evento é alterado somente por um novo evento que está sendo acrescentado. Semelhante à camada de velocidade da arquitetura de uma lambda, todo o processamento de eventos é feito no fluxo de entrada e persistido como uma exibição em tempo real.

Se você precisar recalcular todo o conjunto de dados (equivalente ao que a camada de lote faz no lambda), basta reproduzir o fluxo, normalmente usando o paralelismo para concluir o cálculo em tempo hábil.

Arquitetura Modern Data Warehouse ou Enterprise Data Warehouse

Uma Data Warehouse empresarial permite reunir todos os seus dados em qualquer escala facilmente e obter informações por meio de painéis analíticos, relatórios operacionais ou análises avançadas para todos os seus usuários.

Figura 10 – Arquitetura Modern Data Warehouse



Fonte: Captura Própria

Arquitetura Lakehouse

Um Lakehouse é uma nova arquitetura que permite aos usuários fazer tudo, desde BI, análise SQL, ciência de dados e ML em uma única plataforma. O Lakehouse tem uma abordagem opinativa para construir Data Lakes, adicionando atributos de Data Warehouses – confiabilidade, desempenho e qualidade, mantendo a abertura e a escala de Data Lakes. Suporta:

Transações ACID: Toda operação é transacional, isso significa que cada operação ou é totalmente bem-sucedida ou é abortada. Quando abortada, é registrada e qualquer resíduo é limpo para que você possa tentar novamente mais tarde. A modificação dos dados existentes é possível, porque as transações permitem que você faça atualizações refinadas. As operações em tempo real são consistentes e versões de dados históricos são armazenadas automaticamente. O Lakehouse também fornece instantâneos (snapshots) de dados para permitir que os desenvolvedores facilmente acessem e revertam para versões anteriores para auditorias ou reproduções de experimentos.

Manipulação de grandes metadados: Lakehouse trata da arquitetura metadados, assim como dados, aproveitando a capacidade de processamento

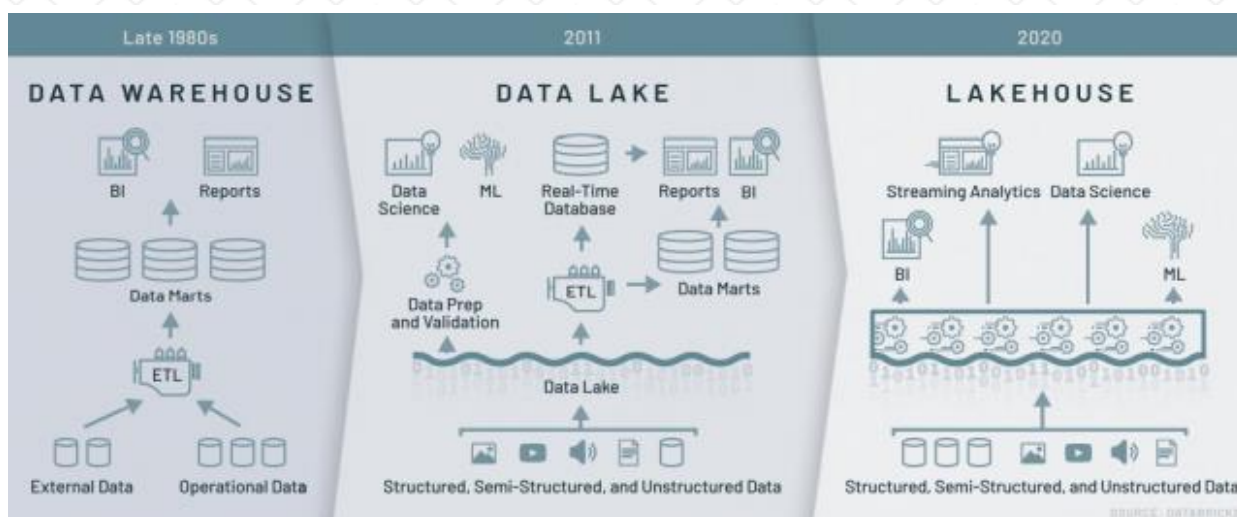
distribuído do Apache Spark utilizada para lidar com todos os seus metadados. Como resultado, ele pode lidar com tabelas em escala de petabyte com bilhões de partições e arquivos com facilidade.

Indexação: Junto com o particionamento de dados, a arquitetura do Lakehouse inclui várias técnicas estatísticas, como filtros bloom (bloom filters) e data skipping para evitar a leitura de grandes porções dos dados no total e, portanto, oferecem acelerações massivas.

Validação de esquema: Todos os seus dados que vão para uma tabela devem aderir estritamente a um esquema definido. Se os dados não satisfizerem o esquema, ele é movido para uma quarentena em que você pode examiná-lo mais tarde e resolver os problemas.

No passado, a tomada de decisão era baseada principalmente em dados estruturados de sistemas operacionais. É essencial para um sistema de gerenciamento de dados de hoje ser muito mais flexível e suportar dados não estruturados em basicamente qualquer formato, permitindo técnicas avançadas de ML. Com a abordagem Lakehouse, essa flexibilidade é alcançada simplificando profundamente a infraestrutura de dados para permitir a aceleração e inovação. Isso é especialmente importante em um momento em que o Machine Learning vem revolucionando todas as áreas da indústria e exige uma infraestrutura elástica suportando velocidade e eficiência operacional.

Figura 11 – Arquitetura Data Lakehouse



Fonte: Databricks

Arquitetura Data Mesh

O Data Mesh é uma nova abordagem para projetar e desenvolver arquiteturas de dados em ambientes complexos. Ao contrário de uma arquitetura centralizada e monolítica baseada em um Data Warehouse e/ou Data Lake, o Data Mesh é uma arquitetura de dados altamente descentralizada.

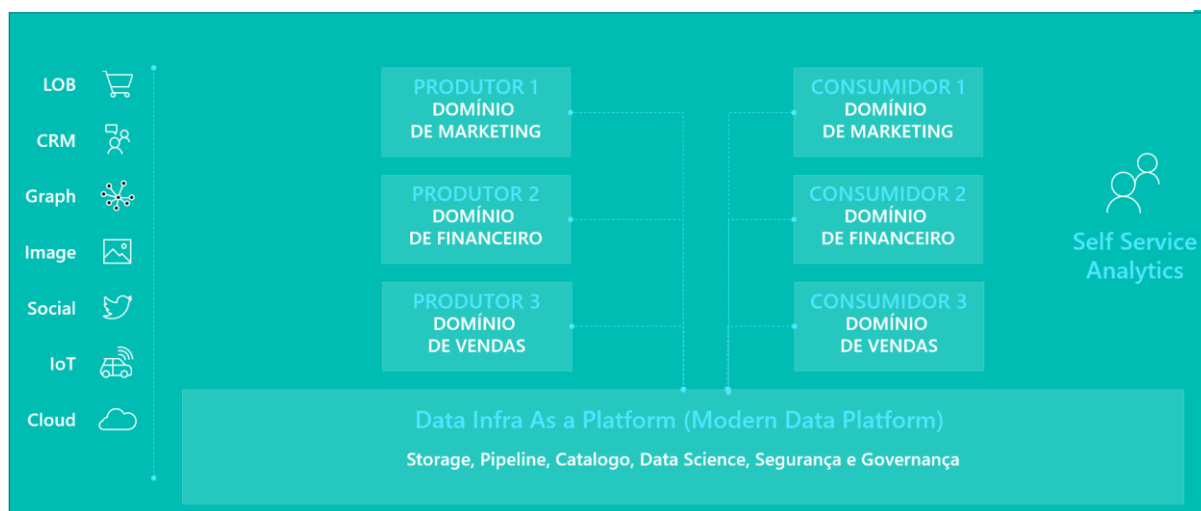
Dessa forma, o Data Mesh desafia a perspectiva tradicional de que o Big Data deve ser centralizado para alavancar seu potencial analítico. Indo de encontro à suposição de que todos os dados precisam estar armazenados em um mesmo local e gerenciados centralmente para entregar seu valor real, o Data Mesh afirma que o Big Data pode alimentar a inovação apenas e somente quando é distribuído entre os proprietários dos dados de domínio, que, então, fornecem os dados como um produto.

Para facilitar esse processo, um novo modelo de governança de TI deve ser implantado com auxílio da automação, garantindo a interoperabilidade necessária. A democratização dos dados é a premissa principal sobre a qual se baseia o conceito do Data Mesh e não pode ser alcançada sem descentralização e priorização da experiência dos consumidores de dados.

Como um paradigma arquitetônico, o Data Mesh oferece a promessa de potencializar a análise em escala, fornecendo acesso rápido a conjuntos de domínios

distribuídos de rápido crescimento. Isso é ainda mais verdade no caso de cenários de proliferação de consumo, como análise, aprendizado de máquina ou desenvolvimento e implantação de aplicativos centrados em dados.

Figura 12 – Arquitetura Data Mesh



Fonte: Captura Própria

3.4 Organização de um Datalake

Data Lake é um conceito que surgiu em 2010 como “um sistema ou repositório de dados armazenados em seu formato natural/bruto” (DIXON, 2010).

Dentro de um projeto de Big Data, deparamo-nos com questões que são conceituais e não existem ferramentas que entregam essas respostas prontas. Uma dessas questões está relacionada à organização das informações no repositório de dados (Data Lake).

Dentro de um bom projeto de Data Lake, as zonas permitem a separação lógica/física dos dados que mantêm o ambiente seguro, organizado e facilmente compreensível.

Não existe formato certo ou errado, o importante é que seja documentado um esquema de controle dentro do repositório Data Lake para que esse esquema seja seguido. Um dos formatos que está crescendo no mercado é a utilização de camadas

Bronze, Silver e Gold, por meio de utilização de tabelas de metadados chamados Delta Tables.

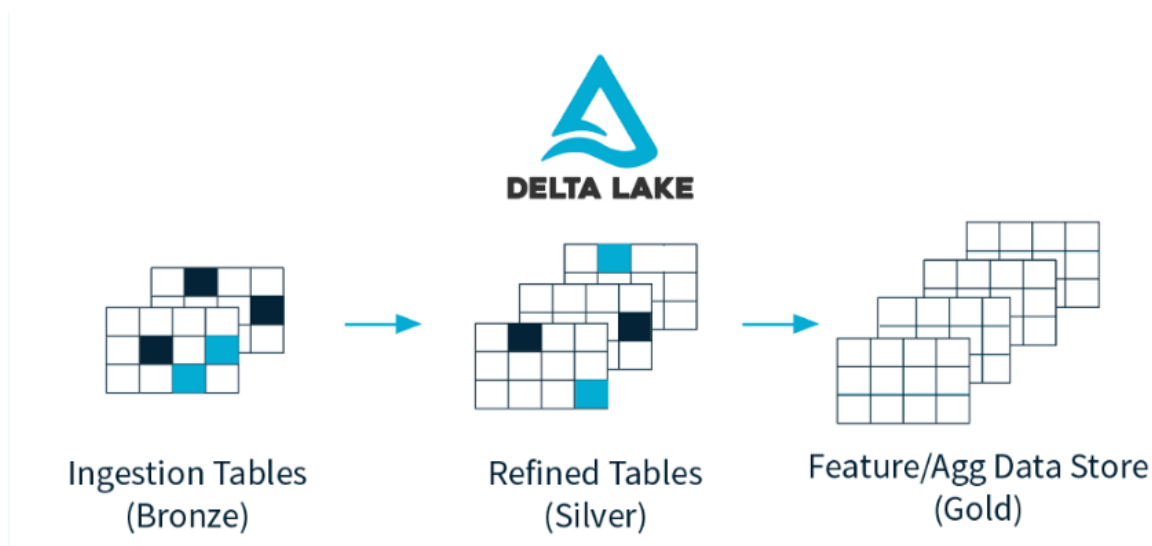
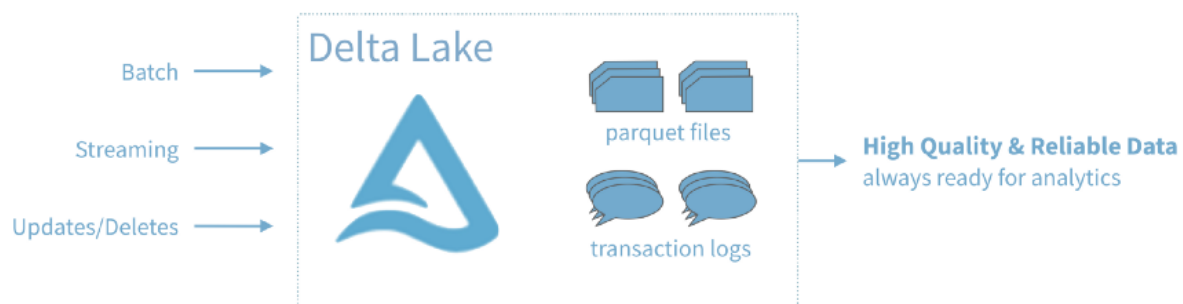
Considere os dados se movendo por diferentes tabelas de controle denominadas Bronze, Silver e Gold. Os dados dos jobs de Streaming e Batch são primeiro capturados na tabela Bronze em seus formatos brutos, depois nas tabelas Silver são limpos e o processamento é feito para torná-los "consultáveis". As tabelas Gold contêm métricas de negócios chaves, agregadas, que são consultadas com frequência.

Bronze: Contêm os dados brutos ingeridos de várias fontes, como arquivos json, dados de sistemas RDBMS, dados IoT etc.

Silver: Fornecem uma visão mais refinada dos dados Bronze.

Gold: Fornece agregados de nível de negócios geralmente usados para relatórios e painéis.

Figura 13 – Estrutura Delta Lake/ Delta Tables



Fonte: Databricks

O Delta Lake fornece transações ACID, tratamento de metadados escalonáveis e unifica o processamento de dados batch e streaming. O Delta Lake é executado sobre o Data Lake existente e é totalmente compatível com as APIs do Apache Spark.

Delta Lake é construído principalmente sobre três pilares que abordam os desafios de confiabilidade, desempenho e engenharia:

Dados limpos e de qualidade;

Visualizações consistentes em cargas de trabalho de dados batch e stream;

Otimizado e fácil de adotar.

Especificamente, o Delta Lake oferece:

- ✓ Transações ACID no Spark: Níveis de isolamento serializáveis asseguram que os leitores nunca vejam dados inconsistentes.
- ✓ Manipulação de metadados escaláveis: Aproveita a capacidade de processamento distribuído do Spark para lidar com todos os metadados para tabelas de escala de petabytes com bilhões de arquivos com facilidade.
- ✓ Unificação de stream e batch: Uma tabela no Delta Lake pode ser atualizada via streaming. A ingestão de dados streaming permite as consultas interativas imediatamente.
- ✓ Flexibilidade de esquema: Manipula automaticamente as variações de esquema para evitar a inserção de registros inválidos durante a ingestão.
- ✓ Dados históricos: O controle de versão de dados permite reversões, trilhas de auditoria de histórico completo e experimentos de aprendizado de máquina reproduzíveis.
- ✓ Upserts e Delete: Dá suporte a operações de upsert, atualização e exclusão para habilitar casos de uso complexos, como Change-Data-Capture, operações SCD (slowly-Changing-Dimension), streaming Upserts e assim por diante.

As otimizações do mecanismo Delta possibilitam as operações do Delta Lake serem realizadas com alto desempenho, dando suporte a uma variedade de cargas de trabalho que variam de processamento de ETL em larga escala até consultas interativas e ad hoc.

Delta Lake usa arquivos Parquet com versão para armazenar seus dados em nuvem. Além das versões, Delta Lake também armazena um log de transações para manter o controle de todos os commits feitos na tabela ou diretório de armazenamento de blob para fornecer transações ACID.

3.5 Governança e Segurança de dados

O modelo DAMA-DMBoK® V2 está estruturado em 17 capítulos, sendo que 11 estão relacionados com as áreas de conhecimento (houve acréscimo de uma com

relação ao DMBOK1-Integração e Interoperabilidade), além de outros assuntos relevantes, como Ética no tratamento de dados, Big Data&Ciência de dados; Avaliação de maturidade em gestão de dados; Possíveis papéis e organização da gestão de dados e Gerência de mudanças na gestão de dados. Alguns desses capítulos, pelo aspecto temporal, são praticamente novidades quando comparados com o DMBOK1 (Big Data&Ciência de dados e Modelos de maturidade) e outros foram expandidos e enriquecidos, transformando o DAMA-DMBoK® V2 numa fonte, quase incomparável de referência em Gestão de Dados (DAMA, 2012).

Figura 14 – Modelo DAMA DMBOK de Governança de dados



Fonte: Captura Própria

O DMBok apresenta essas 10 atividades importantes para se manter uma governança de dados em uma organização.

1. Governança de dados

Exercício da autoridade e controle, que quer dizer o planejamento, monitoramento e engajamento sobre o gerenciamento de ativos de dados.

2. Gestão de Arquitetura de Dados

Definição dos dados necessários da organização e o desenho do diagrama mestre para atingir as necessidades

3. Modelagem e desenho de dados

Projetar, implementar e manter soluções para atender às necessidades de dados da organização

4. Gestão de integração de dados e interoperabilidade

Controle do ciclo de vida de dados, também chamada de linhagem do dado, desde a criação até o arquivamento e possível eliminação

5. Gestão de segurança de dados

A regulamentação brasileira da LGPD, que derivou da GDPR, que são leis de proteção a dados pessoais, fez a camada de segurança se tornar ainda mais importante nas estruturas de dados das organizações. Às vezes até é estruturada uma área específica dentro dos times de dados nas empresas para a segurança dos dados ou a segurança da informação.

Mas a responsabilidade é de planejar, desenvolver e executar procedimentos e políticas de segurança para prover autenticação, autorização, acesso e auditoria de dados e informação, atendendo às necessidades de regulações e compliance.

6. Gestão de dados mestres e referência

Garantir consistência com padrão ouro ou "golden record" de valores de dados contextuais. Como você armazena, na sua organização de dados, campos como CPF, telefone, gênero se é M ou F ou se é número, então define padrões para toda a organização de dados.

7. Gestão de DW e BI

Prover dados de suporte para a decisão e suportar trabalhadores do conhecimento engajados em análises, queries e relatórios.

8. Gestão da documentação e conteúdo

Armazenar, proteger e permitir o acesso a dados dentro de arquivos eletrônicos e registros físicos, como arquivos de texto, gráficos, imagens, áudios e vídeos.

9. Gestão de metadados

Definição dos metadados integrados dos dados e manter esse rótulo para um entendimento geral do uso da informação.

Exemplo: Tem um dado na minha base que é se um aluno está matriculado aqui no IGTI, mas o que isso quer dizer? Que ele está cursando um curso? Que ele fez apenas a matrícula e trancou? Então, o metadado é utilizado para explicar o dado que estará presente naquele campo.

10. Gestão da qualidade de dados

Medir, avaliar, otimizar e garantir dados adequados para uso, por exemplo, cadastros incompletos de clientes.

Se a sua organização vai acumular responsabilidades de gestão de dados, isto é, ter um gerente de governança que é responsável por todas essas atividades, ou se será um time com vários profissionais, fica a cargo da empresa, mas são atividades importantes para se manter uma boa governança de dados.

Figura 15 – Planilha de maturidade Governança de dados

Funções de Gestão de dados	Metas e princípios	Atividades	Entregas Primárias	Papéis e Responsabilidades	Tecnologia	Práticas e Técnicas	Organização e Cultura
Governança de dados							
Gestão de Arquitetura de Dados							
Desenvolvimento de dados							
Gestão de operações de dados							
Gestão de segurança de dados							
Gestão de dados mestres e referência							
Gestão de DW e BI							
Gestão da documentação e conteúdo							
Gestão de metadados							
Gestão da qualidade de dados							

Fonte: Captura Própria

Tabela de mapeamento para definir metas e melhorar o nível de maturidade em cada atividade da governança de dados.

Segurança de dados

Quatro medidas de segurança para ambientes de Big Data, segundo Mary E. Shacklett, presidente do Transworld Data, em 2016:

1- Realizar revisões regulares do acesso do usuário aos dados

Em uma base semestral ou anual, a TI deve sentar-se com as partes interessadas corporativas que acessam dados do Data Lake e revisar as permissões de acesso a dados para todos os funcionários autorizados. As permissões de acesso podem ser ajustadas para cima ou para baixo com base em responsabilidades de trabalho de funcionários/contratantes. Quando os funcionários/contratados não estão mais empregados na empresa, eles devem ser imediatamente removidos do acesso.

2- Mascaramento de dados

Em alguns casos, o mascaramento pode ser usado para dados confidenciais (por exemplo, números de CPF, telefones, nomes) para que esses dados não sejam

compartilhados com outros fora da empresa. O mascaramento deve ser especialmente considerado se a empresa quiser vender dados a terceiros.

3- Criptografar dados

Se o Big Data for armazenado em um único repositório de dados que todos os funcionários com autorizações apropriadas são capazes de acessar, a criptografia pode ser usada nos dados. A ideia por trás da criptografia de dados é que você dá a todos a máxima flexibilidade para obter os dados de que precisam, mas eles podem fazê-lo com segurança.

4- Monitore o comportamento

Monitorar continuamente os hábitos de acesso de cada usuário, desenvolver modelos comportamentais para como o usuário acessa os dados e emitir um alerta se, por qualquer razão, houver uma anomalia de acesso ou um padrão de uso que não concorde com a forma como o usuário normalmente usa os dados.

Quando ocorre uma anomalia de uso para um usuário, o acesso do usuário é imediatamente suspenso e uma investigação ocorre. Em alguns casos, uma falha de segurança em estágio inicial pode ser detectada. Em outros casos, é simplesmente uma situação em que um funcionário recebeu um novo cargo ou responsabilidade que exige diferentes formas de acesso e compartilhamento de dados.

3.6 Principais soluções de dados modernas

Em um mercado em constante transformação, é importante sempre estar atento às novas tecnologias e à evolução das tecnologias líderes de cada segmento. Devido à complexidade destas análises, na área de tecnologia, é bastante comum analisar estudos produzidos por institutos de pesquisas globais, como Forrester e Gartner, para selecionar uma lista de soluções de tecnologia que possam atender às necessidades da sua empresa.

[Gartner Magic Quadrant for Data Management Solutions for Analytics](#)

[Gartner Magic Quadrant for Cloud Database Management Systems](#)

Gartner Magic Quadrant for Data Science and Machine Learning Platforms

Vamos avaliar algumas das soluções líderes citadas nos estudos:

Azure Synapse Analytics é um serviço da Microsoft que reúne integração de dados, armazenamento (Data Lake), processamento de Data Warehouse (SQL Pool) e análise de Big Data (Spark).

O **AWS Redshift** é um serviço de DW, que interage com dados no Data Lake do S3 para habilitar análises adicionais por outros serviços AWS

O **GCP BigQuery** é um Data Warehouse que usa consultas SQL super-rápidas usando o poder de processamento da infraestrutura do Google.

<https://azure.microsoft.com/pt-br/services/synapse-analytics/>

<https://aws.amazon.com/pt/redshift>

<https://cloud.google.com/bigquery>

Capítulo 4. Outras plataformas de dados

Para que uma estratégia de uso de dados esteja completa, pode ser necessário adicionar alguns outros softwares em uma arquitetura de dados. Nesta seção, vamos abordar algumas dessas possíveis soluções.

4.1 Soluções para extração, ingestão, transformação e análise de dados

ETL significa Extrair, Transformar e Carregar. É um processo convencional gerenciar pipelines de dados há anos. No entanto, a popularidade crescente de Data Warehouses contemporâneos baseados em nuvem está mudando a ideia habitual de ETL na direção de ELT.

ETL

A Processo ETL inclui três etapas importantes, incluindo Extração, Transformação e Carregamento. Ferramentas ETL buscam dados de um banco de dados e os coloca em outro após a transformação e verificações de qualidade.

A primeira etapa, chamada Extração, envolve retirar dados de uma fonte de dados. Durante essa fase, os dados são lidos e coletados, geralmente de várias fontes, como bancos de dados locais e em nuvem, aplicativos corporativos, sistemas de arquivos e muito mais.

Durante a Transformação, os dados extraídos são então convertidos em um formato aceitável para outro banco de dados. A transformação de dados é feita usando expressões, regras ou tabelas de pesquisa ou mesclando um conjunto de dados com outro nesta fase.

O último passo é Carregando, que é o procedimento para gravar ou empilhar os dados no banco de dados ou no armazém de dados de destino.

O ETL é um componente essencial nos processos modernos de inteligência de negócios. Ele torna possível integrar dados, sejam estruturados ou não, de fontes diferentes em um local para extrair insights de negócios. Algumas pessoas costumam

fazer a pergunta: "ETL está desatualizado?" A resposta a essa pergunta amplamente formulada é que depende das necessidades de uma organização. No entanto, o ETL tem um lugar no legado do Data Warehouse e ainda é usado popularmente por empresas que não sentem a necessidade de migrar para a nuvem.

ELT

ELT é um acrônimo para Extract, Load e Transform. É um processo que extrai dados de um sistema de origem para um sistema de destino e as informações são, então, transformadas para aplicativos downstream.

Ao contrário do ETL, onde a transformação de dados ocorre em uma área de preparação antes de serem carregados no sistema de destino, o ELT carrega os dados brutos diretamente no sistema de destino e os converte lá.

Desta forma, O ELT é mais benéfico para lidar com enormes conjuntos de dados e use-os para inteligência de negócios e análise de dados.

Em comparação com o processo ETL, o ELT reduz consideravelmente o tempo de carregamento. Além disso, o ELT, em comparação com o ETL, é um método mais eficiente em termos de recursos, pois aproveita a capacidade de processamento desenvolvida em uma configuração de armazenamento de dados, diminuindo o tempo gasto na transferência de dados.

4.2 Pesquisas Indexadas/Cognitivas

Com a quantidade de informações disponíveis em documentos, bancos de dados e demais arquivos em nossas empresas, encontrar o que você precisa é sempre muito complexo. Os sistemas de pesquisas indexadas e pesquisas cognitivas organizam centenas de seus conteúdos, criando índices de Pesquisas para fornecer os resultados mais úteis e relevantes em uma fração de segundo e para apresentá-los de uma maneira que ajude você a encontrar o que está procurando.

Esses sistemas de classificação são compostos não por um, mas por uma série de algoritmos. Para fornecer as informações mais úteis, os algoritmos da

pesquisa analisam vários fatores, inclusive palavras da consulta, relevância e usabilidade das páginas, conhecimento das fontes, bem como seu local e configurações. O peso aplicado a cada fator varia de acordo com a natureza da consulta. Por exemplo, a atualização do conteúdo desempenha um papel mais importante na resposta a consultas sobre tópicos de notícias atuais do que sobre definições de dicionário.

Referências

MICROSOFT. Azure documentation. [S. l.], 2021. Disponível em:
<https://docs.microsoft.com/en-us/azure/>. Acesso em: 15 dez. 2021.

LORICA, Ben *et al.* What is a Data Lakehouse. **Databricks**, [S. l.], 2020. Disponível em: <https://databricks.com/blog/2020/01/30/what-is-a-data-lakehouse.html>. Acesso em: 15 dez. 2021.

REAL-TIME DATABASE. *In*: WIKIPEDIA. [S. l.], 2021. Disponível em:
https://en.wikipedia.org/wiki/Real-time_database. Acesso em: Acesso em: 15 dez. 2021.