
FIAP

Data Engineering

Aula 3

Bancos de Dados

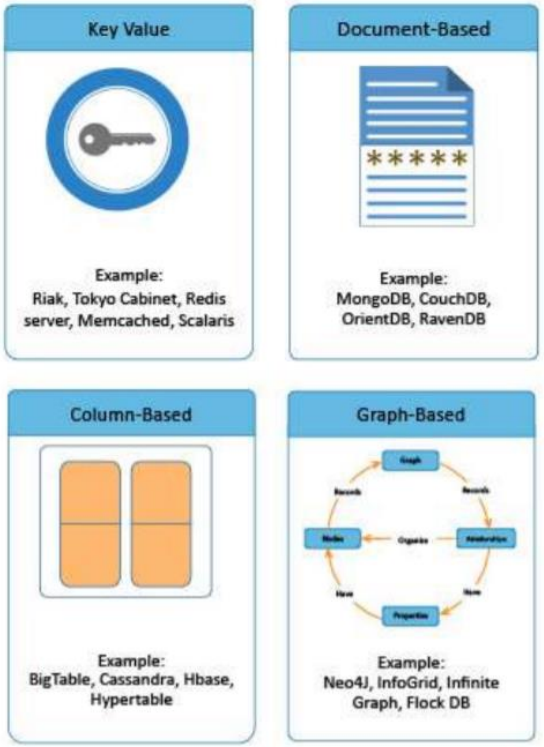
NoSQL x NewSQL

AULA 3 - OBJETIVOS

- Compreender conceitos relacionados aos Bancos de Dados, sua arquitetura e características.
- Diferenciar bancos relacionais de bancos não relacionais.
- Entender a diferença entre ACID e BASE.
- Conhecer o teorema CAP.
- Aplicar os conhecimentos adquiridos, na escolha do melhor banco de dados, conforme os cenários apresentados.

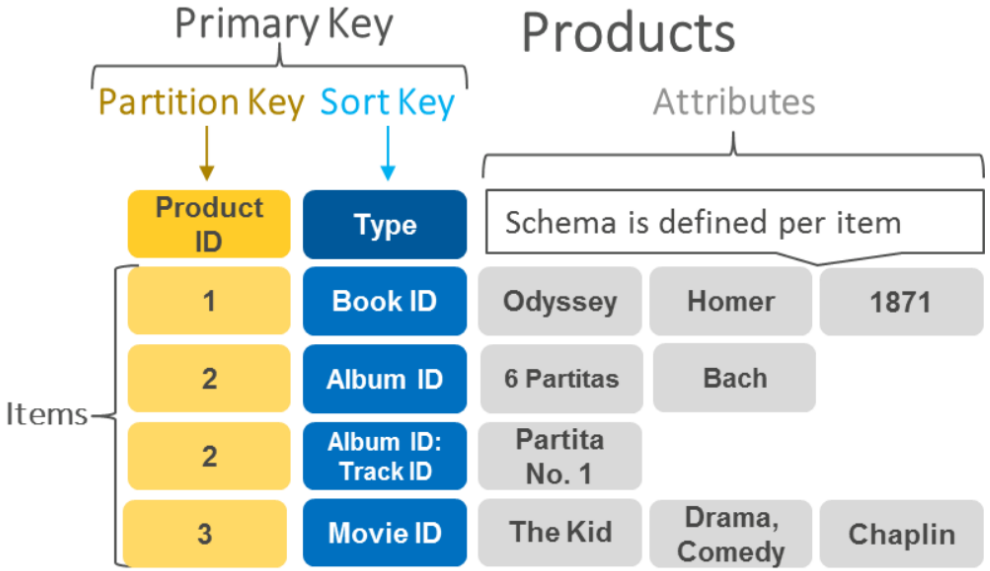
Famílias NoSQL

FAMÍLIAS NOSQL



- **Chave-Valor:** armazena a partir de uma chave, uma série de valores dentro da mesma estrutura. Ideal para simplificação do armazenamento.
- **Documentos:** armazena os dados dentro de uma formato muito próximo ao da aplicação, facilitando o armazenamento.
- **Colunar:** organiza os dados em colunas para uma recuperação mais rápida do mesmo.
- **Grafos:** utiliza a teoria de grafos para estabelecer relações mais naturais entre os assuntos e, portanto, mais fáceis de serem desenhadas.

CHAVE-VALOR



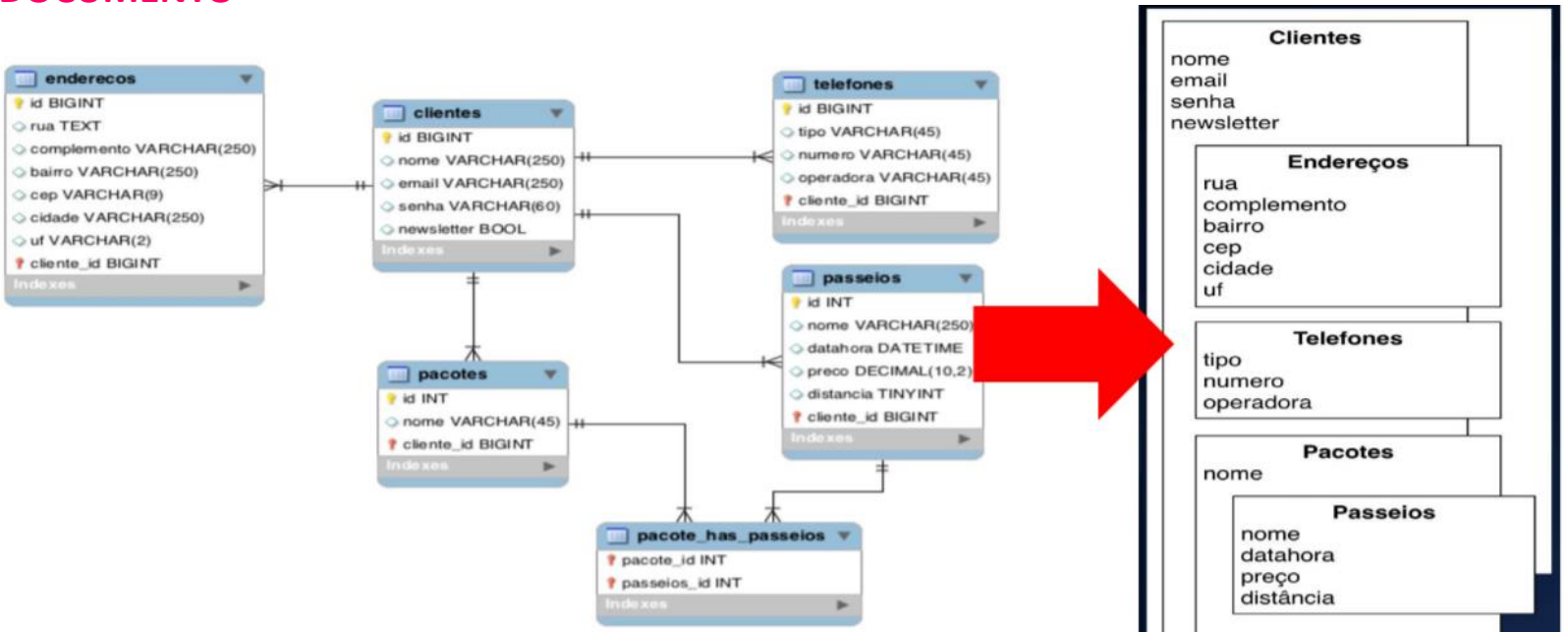
Muito utilizado para armazenar dados de sessões, tais como navegação web e carrinho de compras.;

DOCUMENTO

```Json
[
    {
        "year" : 2013,
        "title" : "Turn It Down, Or Else!",
        "info" : {
            "directors" : [ "Alice Smith", "Bob Jones"],
            "release_date" : "2013-01-18T00:00:00Z",
            "rating" : 6.2,
            "genres" : ["Comedy", "Drama"],
            "image_url" : "http://ia.media-imdb.com/images/N/O9ERWAU7FS797AJ7LU8HN09AMUP908RLlo5JF90EWR7LJKQ7@@._V1_SX400_.jpg",
            "plot" : "A rock band plays their music at high volumes, annoying the neighbors.",
            "actors" : ["David Matthewman", "Jonathan G. Neff"]
        }
    },
    {
        "year": 2015,
        "title": "The Big New Movie",
        "info": {
            "plot": "Nothing happens at all.",
            "rating": 0
        }
    }
]
```

Ideal para armazenamento de catálogos, e gerenciamento de conteúdo, como blogs e plataformas de vídeos.

DOCUMENTO



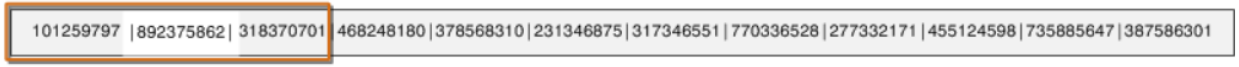
Observe que há uma simplificação no armazenamento, garantindo uma gravação dos dados mais rápida.

COLUNAR

SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL

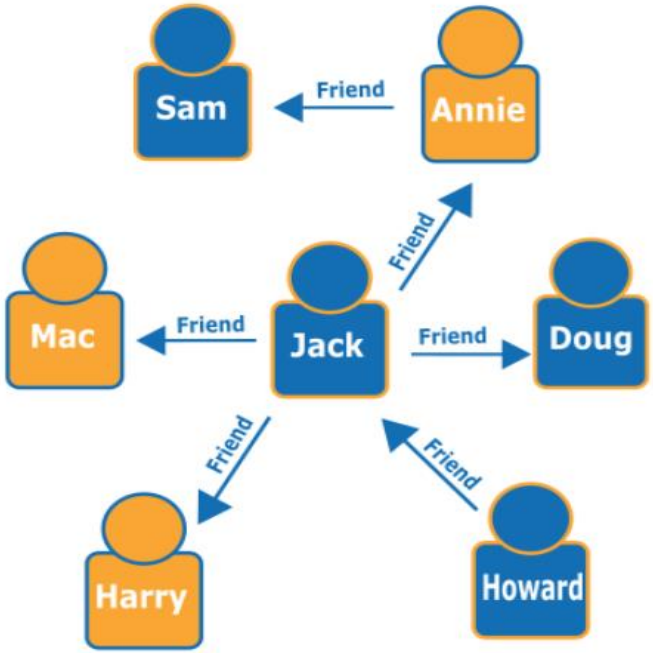
Block 1 **Block 2** **Block 3**

SSN	Name	Age	Addr	City	St
101259797	SMITH	88	899 FIRST ST	JUNO	AL
892375862	CHIN	37	16137 MAIN ST	POMONA	CA
318370701	HANDU	12	42 JUNE ST	CHICAGO	IL



Block 1

GRAFOS



Bastante útil para criar relações que apontem para fraudes, ou mesmo mecanismos de recomendação que reconheçam vínculos.

NOSQL POR FAMÍLIA

Column Store / Column Families	Document Store	Key Value / Tuple Store	Graph Databases
HBase	MongoDB	DynamoDB	Neo4J
Cassandra	Couchbase Server	Azure Table Storage	Infinite Graph
Hypertable	CouchDB	Riak	HyperGraphDB
Amazon SimpleDB	RethinkDB	Redis	GraphBase
Hypertable	Terrastore	Voldemort	Trinity
...	...	...	...

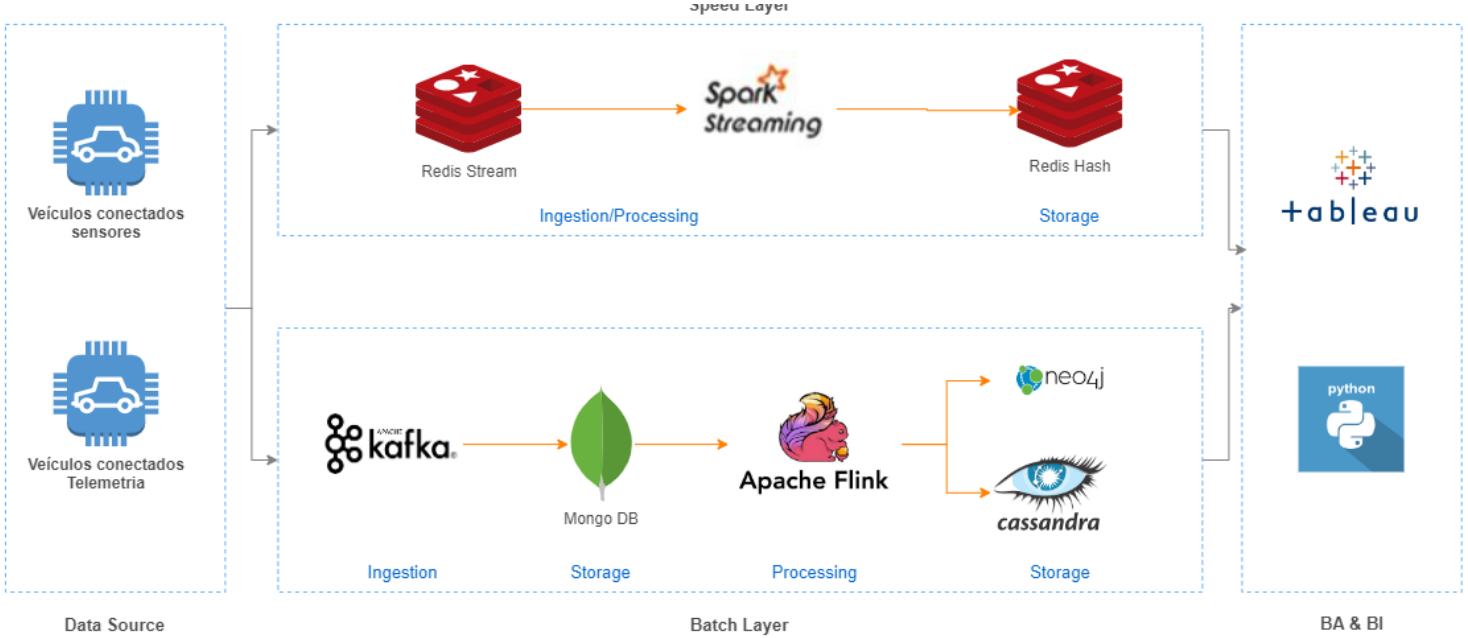
http://nosql-database.org/

397 systems in ranking, October 2022

Rank Oct 2022	Rank Sep 2022	Rank Oct 2021	DBMS	Database Model	Score Oct 2022	Score Sep 2022	Score Oct 2021
1.	1.	1.	Oracle	Relational, Multi-model	1236.37	-1.88	-33.98
2.	2.	2.	MySQL	Relational, Multi-model	1205.38	-7.09	-14.39
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	924.68	-1.62	-45.93
4.	4.	4.	PostgreSQL	Relational, Multi-model	622.72	+2.26	+35.75
5.	5.	5.	MongoDB	Document, Multi-model	486.23	-3.40	-7.32
6.	6.	6.	Redis	Key-value, Multi-model	183.38	+1.91	+12.03
7.	7.	↑ 8.	Elasticsearch	Search engine, Multi-model	151.07	-0.37	-7.19
8.	8.	↓ 7.	IBM Db2	Relational, Multi-model	149.66	-1.73	-16.30
9.	9.	↑ 11.	Microsoft Access	Relational	138.17	-1.87	+21.79
10.	10.	↓ 9.	SQLite	Relational	137.80	-1.02	+8.43
11.	11.	↓ 10.	Cassandra	Wide column	117.95	-1.17	-1.33
12.	12.	12.	MariaDB	Relational, Multi-model	109.31	-0.85	+6.71
13.	13.	↑ 18.	Snowflake	Relational	106.72	+3.22	+48.46
14.	14.	↓ 13.	Splunk	Search engine	94.66	+0.60	+4.04
15.	15.	↑ 16.	Amazon DynamoDB	Multi-model	88.35	+0.93	+11.80
16.	16.	↓ 15.	Microsoft Azure SQL Database	Relational, Multi-model	84.96	+0.54	+5.24
17.	17.	↓ 14.	Hive	Relational	80.60	+2.17	-4.14
18.	18.	↓ 17.	Teradata	Relational, Multi-model	66.07	-0.51	-3.76
19.	19.	19.	Neo4j	Graph	58.68	-0.79	+0.81
20.	20.		Databricks	Multi-model	57.61	+1.99	
21.	21.	↑ 22.	Solr	Search engine, Multi-model	53.50	-0.55	+2.34
22.	↑ 24.	↑ 25.	Google BigQuery	Relational	52.45	+2.33	+8.66
23.	23.	↓ 21.	FileMaker	Relational	52.41	+0.84	-0.44
24.	↓ 22.	↓ 20.	SAP HANA	Relational, Multi-model	52.08	+0.26	-3.20
25.	25.	↓ 23.	SAP Adaptive Server	Relational, Multi-model	42.97	+0.11	-5.62

https://db-engines.com/en/ranking

PERSISTÊNCIA POLIGLOTA



SQL x NoSQL x NewSQL

NEWSQL

- Oferecem a velocidade e escalabilidade do mundo NoSQL, sem abrirem mão das garantias ACID dos bancos de dados SQL. Portanto, buscam aproveitar o melhor dos dois mundos, relacional e não relacional.

Datastore

O Datastore é um banco de dados NoSQL altamente escalonável, ideal para aplicativos da Web e de dispositivos móveis.

Faça uma avaliação gratuita

Veja a documentação do produto.

Banco de dados NoSQL altamente escalonável

Firestore é a próxima geração do Datastore. Saiba mais sobre como atualizar para o Firestore.

O Datastore é um banco de dados NoSQL altamente escalonável para seus aplicativos. Ele controla automaticamente a fragmentação e a replicação para fornecer um banco de dados altamente disponível e com muita durabilidade, que é escalonado de forma automática para administrar a carga dos seus aplicativos. O Datastore fornece uma série de recursos, como transações ACID, consultas semelhantes a SQL, índices e muito mais.

NEWSQL

Cloud Spanner

Banco de dados relacional totalmente gerenciado com escala ilimitada, consistência forte e até 99,999% de disponibilidade.

Novos clientes ganham US$ 300 em créditos para gastar no Spanner. Todos os clientes podem criar uma instância de teste gratuito de 90 dias do Spanner com 10 GB de armazenamento, sem cobranças de créditos.

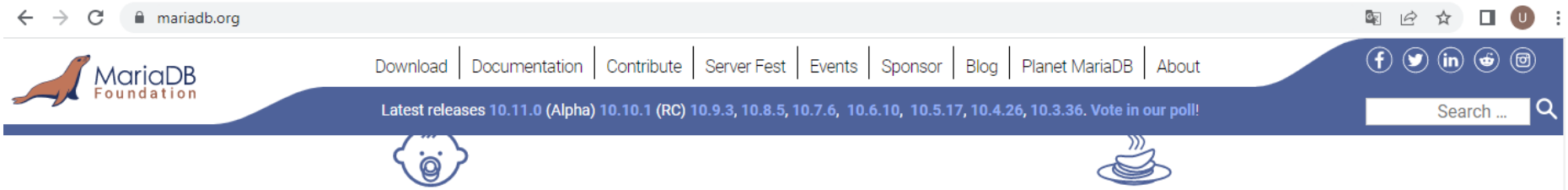
Faça uma avaliação gratuita do Spanner

- Tenha todos os benefícios da semântica relacional e do SQL com escalonamento ilimitado
- Comece com qualquer tamanho e ajuste o escalonamento sem limites de acordo com o aumento das suas necessidades. Faça um teste gratuito.
- Aproveite a alta disponibilidade com nenhuma inatividade programada e usando mudanças de esquema on-line
- Faça transações de alto desempenho com consistência forte em todas as regiões e continentes
- Foque na inovação e elimine tarefas manuais com recursos como a fragmentação automática

VÍDEO

Como o Uber é escalonado para milhões de solicitações simultâneas?

20:03



New to MariaDB Server?

MariaDB Server is one of the most popular open source relational databases. It's made by the original developers of MySQL and guaranteed to stay open source. It is part of most cloud offerings and the default in most Linux distributions.

It is built upon the values of performance, stability, and openness, and MariaDB Foundation ensures contributions will be accepted on technical merit. Recent new functionality includes advanced clustering with Galera Cluster 4, compatibility features with Oracle Database and Temporal Data Tables, allowing one to query the data as it stood at any point in the past.

Blog | Documentation

Latest releases

Development versions

10.11.0 26 Sep 2022 Release notes

RC versions

10.10.1 22 Aug 2022 Release notes

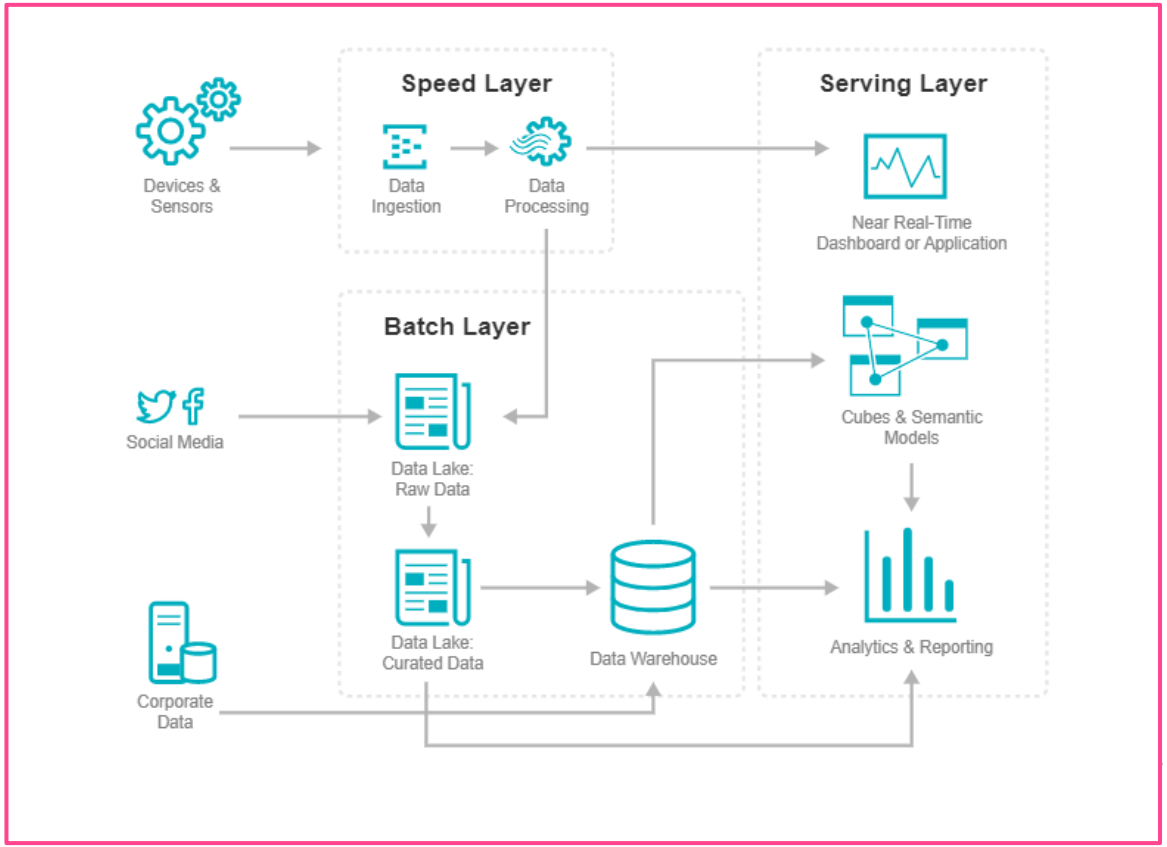
Stable versions

10.9.3 19 Sep 2022 Release notes
10.8.5 19 Sep 2022 Release notes
10.7.6 19 Sep 2022 Release notes
10.6.10 19 Sep 2022 Release notes
10.5.17 16 Aug 2022 Release notes
10.4.26 16 Aug 2022 Release notes
10.3.36 16 Aug 2022 Release notes

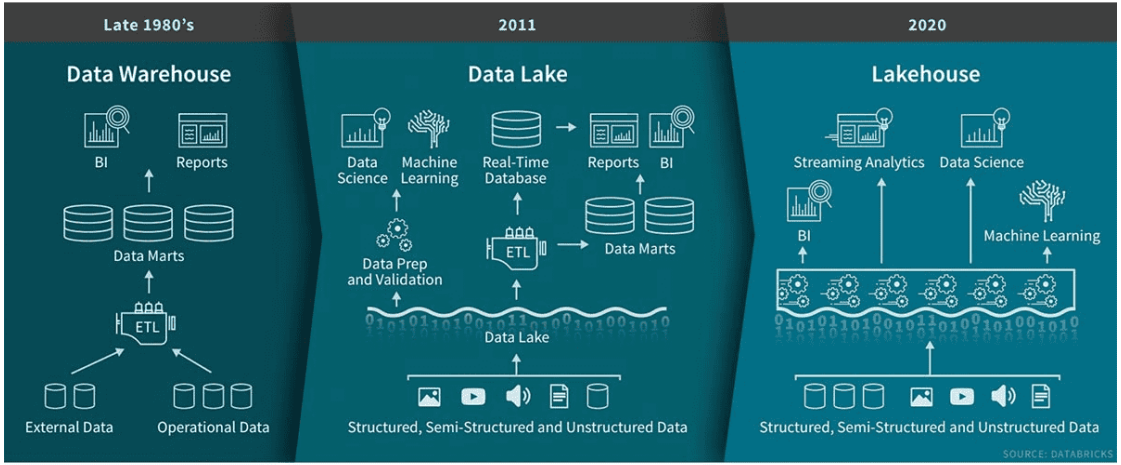
All releases

DATA LAKEHOUSE

ARQUITETURA LAMBDA – BI + BIG DATA

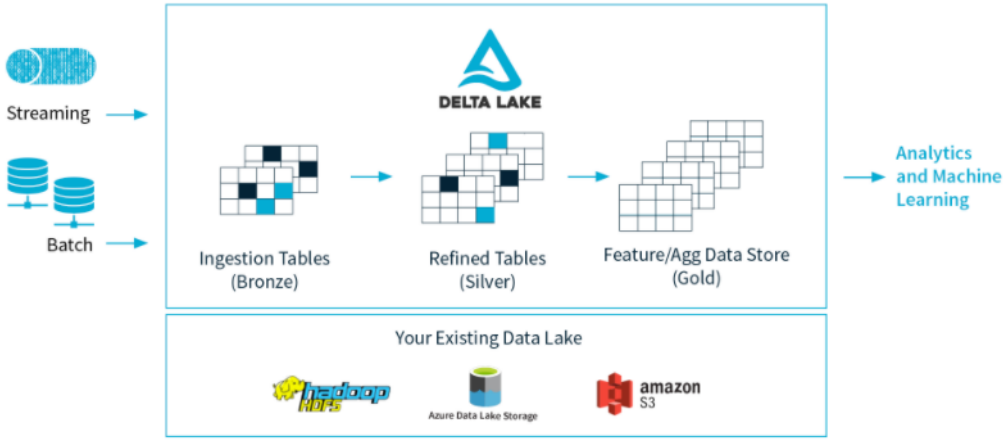


DATA LAKE HOUSE

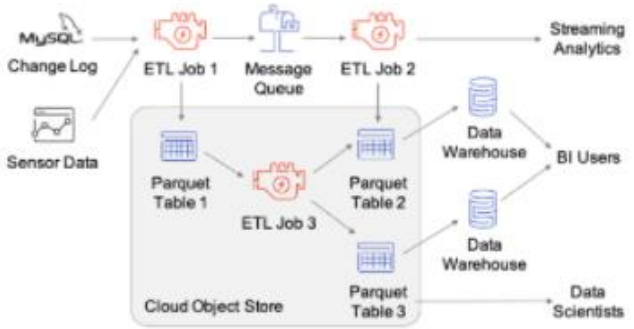


DELTA LAKE

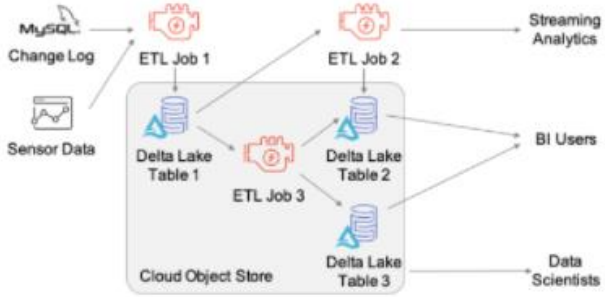
Delta Lake is an open-source storage layer that brings ACID transactions to Apache Spark™ and big data workloads.



ARQUITETURA LAMBDA COM DELTA LAKE



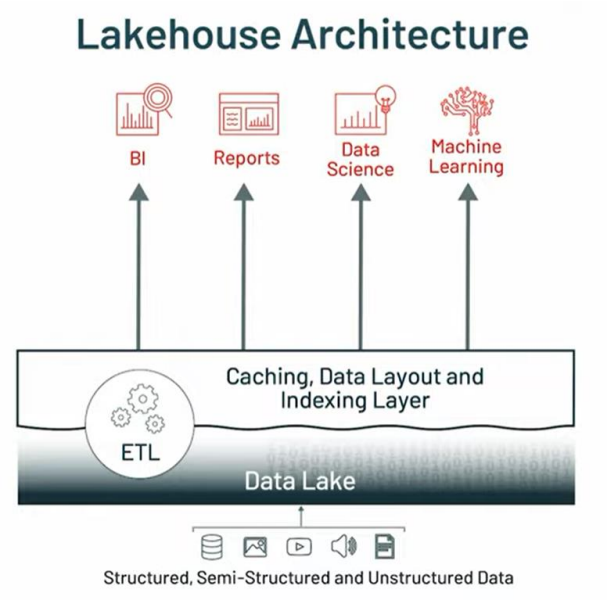
(a) Pipeline using separate storage systems.



(b) Using Delta Lake for both stream and table storage.

FORMATO DELTA

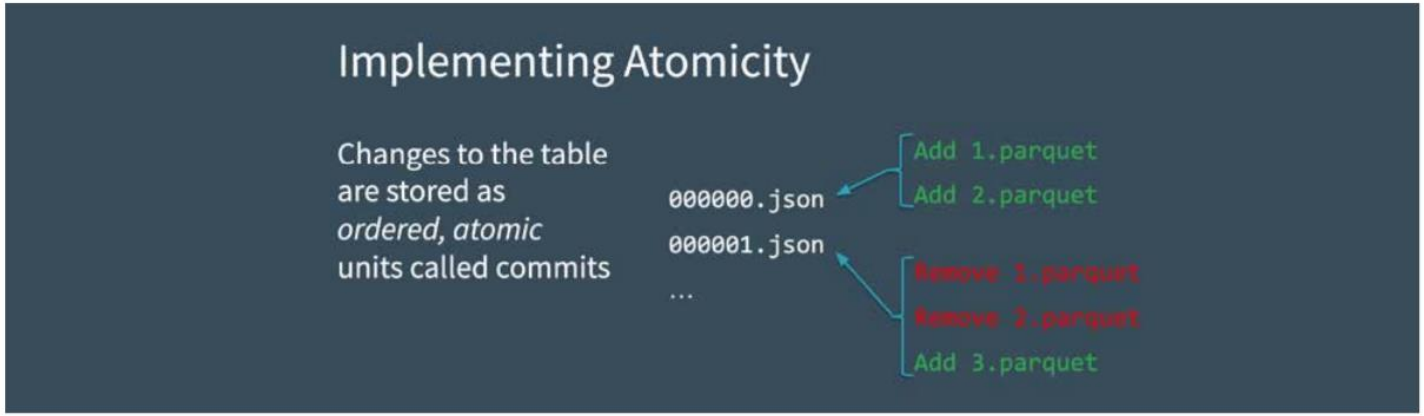
DOS MESMOS CRIADORES DE...



DELTA LAKE

O QUE É O DELTA LAKE?

- Delta Lake é um sistema de gerenciamento de dados unificado.
- Ele cria uma camada sobre os Data Lakes e permite a manipulação dos dados via Spark.
- Essa nova camada delta traz a flexibilidade de modificação e exclusão dos dados(CRUD). Assim como, oferece controles transacionais (ACID) característicos do universo SQL.
- Com o Delta Lake é possível fazer todo o gerenciamento do ciclo de vida dos dados.



- **Atomicidade:** O Delta Lake registra em seu log somente transações que sejam executadas completamente. Se não estiver registrado no log, nunca aconteceu! Transações precisam ter começo, meio e fim!

- **Isolamento:** DataBricks emprega um controle otimista de concorrência, ou seja, prevê no melhor cenário que todos podem fazer operações simultâneas porque não estarão manipulando o mesmo dado ao mesmo tempo. Contudo, caso isso ocorra, o protocolo aplicado é o de exclusão mútua, no qual nenhuma das operações recebe commit.

AUDITORIA E GOVERNANÇA

Review Delta Lake Table History for Auditing & Governance

All the transactions for this table are stored within this table including the initial set of insertions, update, delete, merge, and inserts with schema modification

```
%sql
DESCRIBE HISTORY loans_delta
```

	version	timestamp	userId	userName	operation	operationParameters
1	111	2021-01-04T20:23:00.000+0000	101001	Brenner.Heintz@databricks.com	WRITE	{"mode": "Append", "partition
2	110	2021-01-04T20:20:57.000+0000	101001	Brenner.Heintz@databricks.com	STREAMING UPDATE	{"outputMode": "Append", "qu
3	109	2021-01-04T20:20:56.000+0000	101001	Brenner.Heintz@databricks.com	STREAMING UPDATE	{"outputMode": "Append", "qu
4	108	2021-01-04T20:20:51.001+0000	101001	Brenner.Heintz@databricks.com	STREAMING UPDATE	{"outputMode": "Append", "qu
5	107	2021-01-04T20:20:51.000+0000	101001	Brenner.Heintz@databricks.com	STREAMING UPDATE	{"outputMode": "Append", "qu
6	106	2021-01-04T20:20:47.000+0000	101001	Brenner.Heintz@databricks.com	STREAMING UPDATE	{"outputMode": "Append", "qu
7	105	2021-01-04T20:20:46.000+0000	101001	Brenner.Heintz@databricks.com	STREAMING UPDATE	{"outputMode": "Append", "qu
8	104	2021-01-04T20:20:43.000+0000	101001	Brenner.Heintz@databricks.com	STREAMING UPDATE	{"outputMode": "Append", "qu

Showing all 112 rows.

TIME TRAVEL

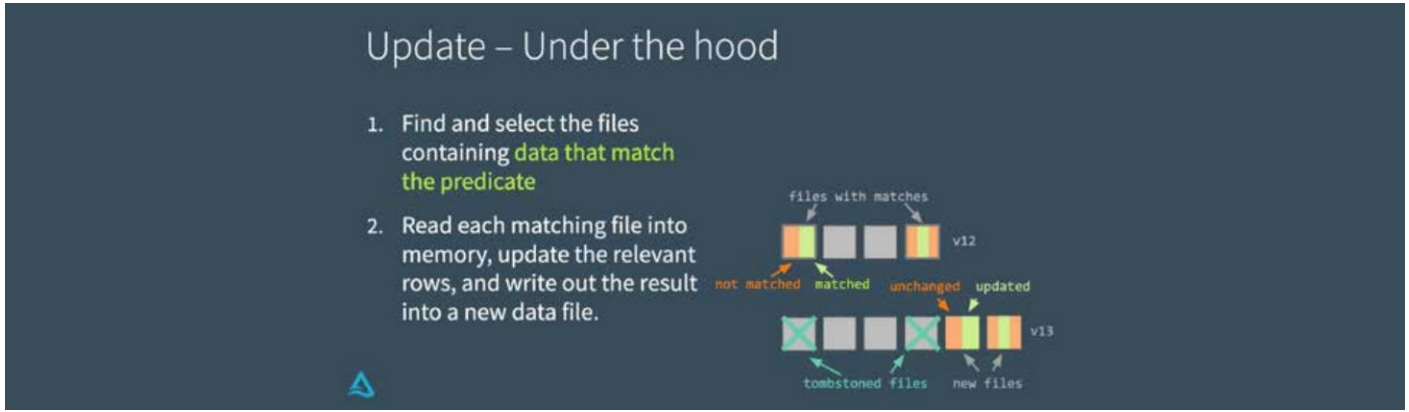
Use time travel to select and view the original version of our table (Version 0).

As you can see, this version contains the original 14,705 records in it.

```
spark.sql("SELECT * FROM loans_delta VERSION AS OF 0").show(3)
spark.sql("SELECT COUNT(*) FROM loans_delta VERSION AS OF 0").show()
```

```
+-------+-----------+---------+----------+-----+--------------------+
|loan_id|funded_amnt|paid_amnt|addr_state| type|           timestamp|
+-------+-----------+---------+----------+-----+--------------------+
|      0|       1000|   182.22|        CA|batch|2021-01-04 20:15:...|
|      1|       1000|   361.19|        WA|batch|2021-01-04 20:15:...|
|      2|       1000|   176.26|        TX|batch|2021-01-04 20:15:...|
+-------+-----------+---------+----------+-----+--------------------+
only showing top 3 rows

+--------+
|count(1)|
+--------+
|   14705|
+--------+
```



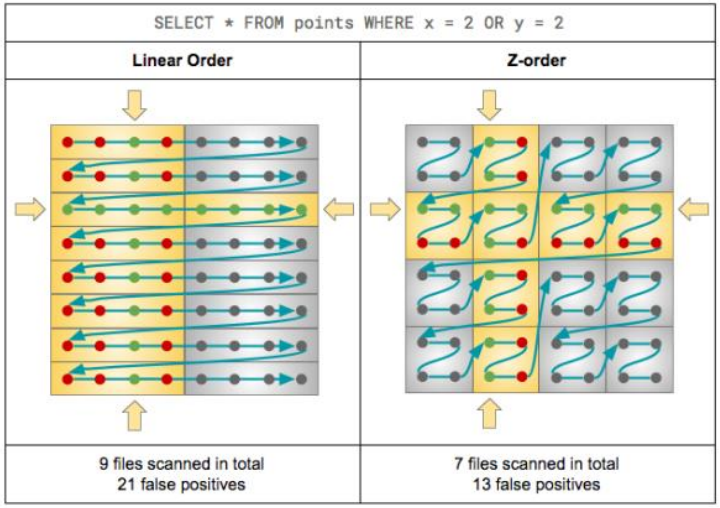
- Update/Delete: O Delta Lake carrega na memória os dados que serão atualizados/apagados. Faz um match entre as alterações e o que permaneceu intacto. E gera um novo arquivo, apenas marcando o anterior como “inativo” para que ele possa continuar sendo utilizado dentro do time travel.

- **Delete + Vacuum:** Com o comando vacuum é possível excluir permanentemente os arquivos de dados mais antigos (que já ultrapassaram o limite de retenção) ou que não estejam ativos.

- É possível também configurar o tempo de retenção, que por padrão é de 7 dias:

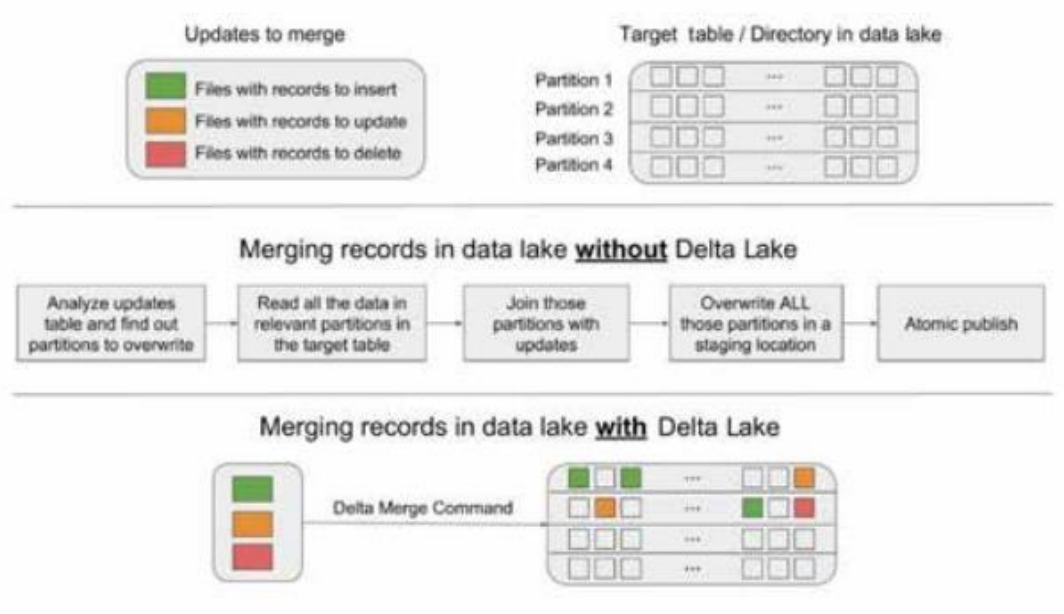
```
from delta.tables import * deltaTable.
# vacuum files older than 30 days(720 hours)
deltaTable.vacuum(720)
```

- **Optimize + Z-Order:** Optimize permite que arquivos pequenos sejam agrupados em arquivos maiores. E junto do ZORDER faz uma organização de co-localidade deixando informações que possuem alguma relação juntas, o que traz maior velocidade às consultas!




```
OPTIMIZE <table> [WHERE <partition_filter>]
ZORDER BY (<column> [, …])
```

MERGE


Updates to merge
Files with records to insert
Files with records to update
Files with records to delete
Target table / Directory in data lake
Partition 1
Partition 2
Partition 3
Partition 4
Merging records in data lake without Delta Lake
Analyze updates table and find out partitions to overwrite
Read all the data in relevant partitions in the target table
Join those partitions with updates
Overwrite ALL those partitions in a staging location
Atomic publish
Merging records in data lake with Delta Lake
Delta Merge Command

MERGE

```
MERGE INTO users
USING (
SELECT userId, latest.address AS address, latest.deleted AS deleted FROM
(
SELECT userId, MAX(struct(TIME, address, deleted)) AS latest
FROM changes GROUP BY userId
)
) latestChange
ON latestChange.userId = users.userId
WHEN MATCHED AND latestChange.deleted = TRUE THEN
DELETE
WHEN MATCHED THEN
UPDATE SET address = latestChange.address
WHEN NOT MATCHED AND latestChange.deleted = FALSE THEN
INSERT (userId, address) VALUES (userId, address)
```

DEMOS

Formato Delta

https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/4202364302108746/4093297341796551/6137813644088450/latest.html

Batch + Streaming

https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/4202364302108746/1675557929204906/6137813644088450/latest.html


Hands On

VAMOS APLICAR?

Construa uma estrutura em streaming no formato delta para lidar com os dados de e-commerce.

Preencha o notebook: DeltaLogStreamingHandsOn

https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/4202364302108746/2164175383573387/6137813644088450/latest.html

Use como referência o note: Demo Lambda + 2 speed + 1 batch

https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/4202364302108746/1675557929204906/6137813644088450/latest.html

Como foi a sua experiência com a aula de hoje?

https://fiap.me/AutoglassArquiteturadeSoftware

Foi bom estar com vocês!

https://www.linkedin.com/in/tassianarugoni/

AGORA JÁ SABEMOS UM POUQUINHO MAIS ;)

FIAP


FIAP