# DATABRICKS
# Delta Lake

Grimaldo Oliveira

# O que é o DATABRICS Delta Lake

1. Databricks é uma poderosa plataforma de colaboração entre os profissionais da área de dados. É uma plataforma fácil de usar para aqueles que desejam executar consultas em seu Data Lake.
2. O Delta Lake é um projeto de código aberto  que permite construir uma arquitetura chamada Lakehouse sobre o Data Lake no Databricks.
3. O Delta Lake fornece processamento em batch, streaming em lote, além de controle de transações sobre os dados, como um banco de dados.

**Grimaldo Oliveira – grimaldo_lopes@hotmail.com**

# Sites Importantes

**DataBricks**

https://databricks.com/

**Documentação**

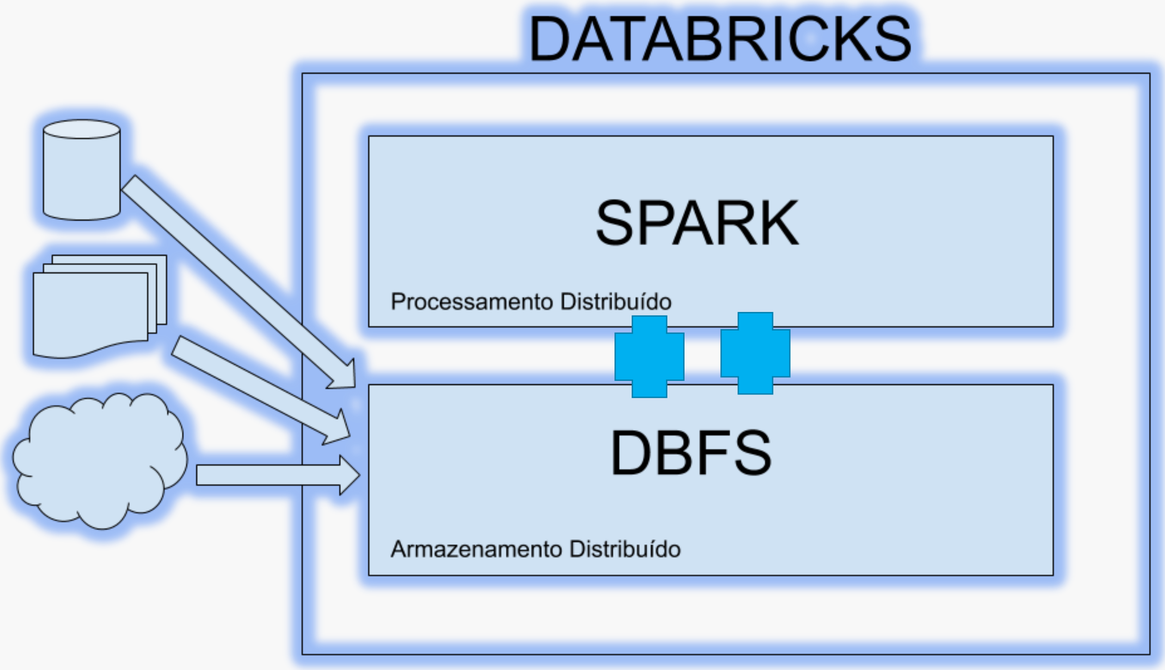
https://docs.databricks.com/

**Databricks Community**

https://community.cloud.databricks.com/login.html

# Ecossistema

Como funciona a leitura dos dados para serem carregados no Databricks Store e gerenciado pelo SPARK.



# Databricks

Uma plataforma aberta para armazenar e gerenciar todos os seus dados para que você possa utilizar em seus projetos com dados.

## Databricks

Uma plataforma aberta destinada para armazenar e gerenciar todos os seus dados para todas as suas cargas de dados.

É dividida em :

**NOTEBOOKS**:  Permite que os analistas e cientistas de dados construam seus scripts nas linguagens Python, SQL, R, Scala, fazendo consulta aos dados.

**ANALYTICS**: Use a poderosa interface SQL Analytics para consultar e visualizar dados e preparar dashboards.

**DELTA LAKE**: Permite que sejam combinados os diversos tipos de dados inseridos no Databricks, pra que ajustes, tratamentos nos dados possam ser executados.

**MACHINE LEARNING**: É possível desenvolver modelos matemáticos e estatísticos para a geração de informação para seus negócios.

# SPARK

Faz todo o gerenciamento e distribuição do processamento executado no Databricks.

## Apache Spark

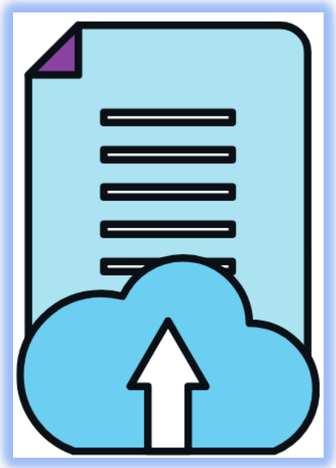
Apache Spark é o principal **mecanismo** de análise unificado para Big Data e aprendizado de máquina que existe no mundo, sendo utilizado pelas grandes corporações. Explorando nas suas execuções o uso de memória e outras otimizações. Anteriormente as empresas utilizavam o Hadoop.

# DBFS

Onde os dados são armazenados e gerenciados.

## DBFS

Databricks File System (DBFS) é um sistema de arquivos distribuído montado em um espaço de trabalho Databricks e disponível em clusters Databricks.

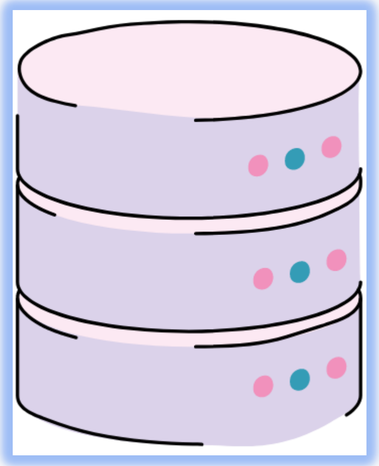


# Cluster no Databricks

É  onde os recursos para operacionalização são criados. Precisa criar um Cluster para que o Databricks seja operacional.

## Cluster

É um conjunto de recursos e configurações em que você cria os seus projetos, dentro dos chamados notebooks, é possível dentro de um cluster executa cargas de trabalho e analisar seus dados.



# Delta Lake

O Delta Lake fornece transações ACID, manuseio de metadados escaláveis e unifica o processamento de dados em lote e de streaming.

## Delta Lake

Tem como principal objetivo de melhorar o suporte e resolver os problemas de atualização concorrentes dos Data Lakes.

É composto por:

**ACID**:  Inclusão de características **ACID (Atomicidade, Consistência, Isolamento, Durabilidade)** aos Data Lakes permitindo um controle para evitar inconsistências nos dados.

**ARMAZENAMENTO**: O Delta Lake pode ser visto como uma camada adicional em cima do sistema de arquivo do Data Lake (HDFS,S3, etc).

**Versionamento dos dados**: Permite que os dados sejam acessados e revertam para versões anteriores de dados, controle de históricos.

**Formato**: Todos os dados no Delta Lake são armazenados no formato Apache Parquet, já que o Parquet é extremamente eficiente em compactação.

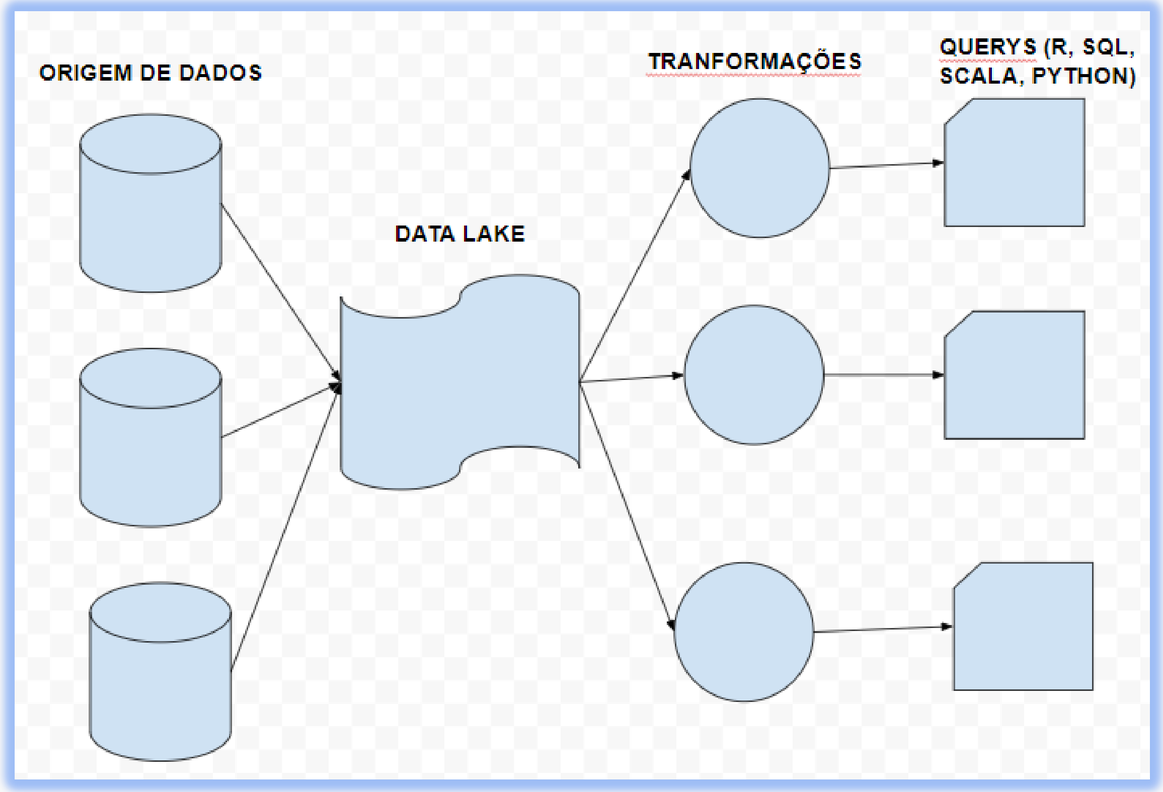
**Processamento em lote (batch) e/ou Processamento de Fluxos Contínuos (Streams)** : Permite a ingestão de dados tanto em batch quanto em streaming.

# Ecossistema

Vamos entender as diferenças entre

Data Lake X Delta Lake

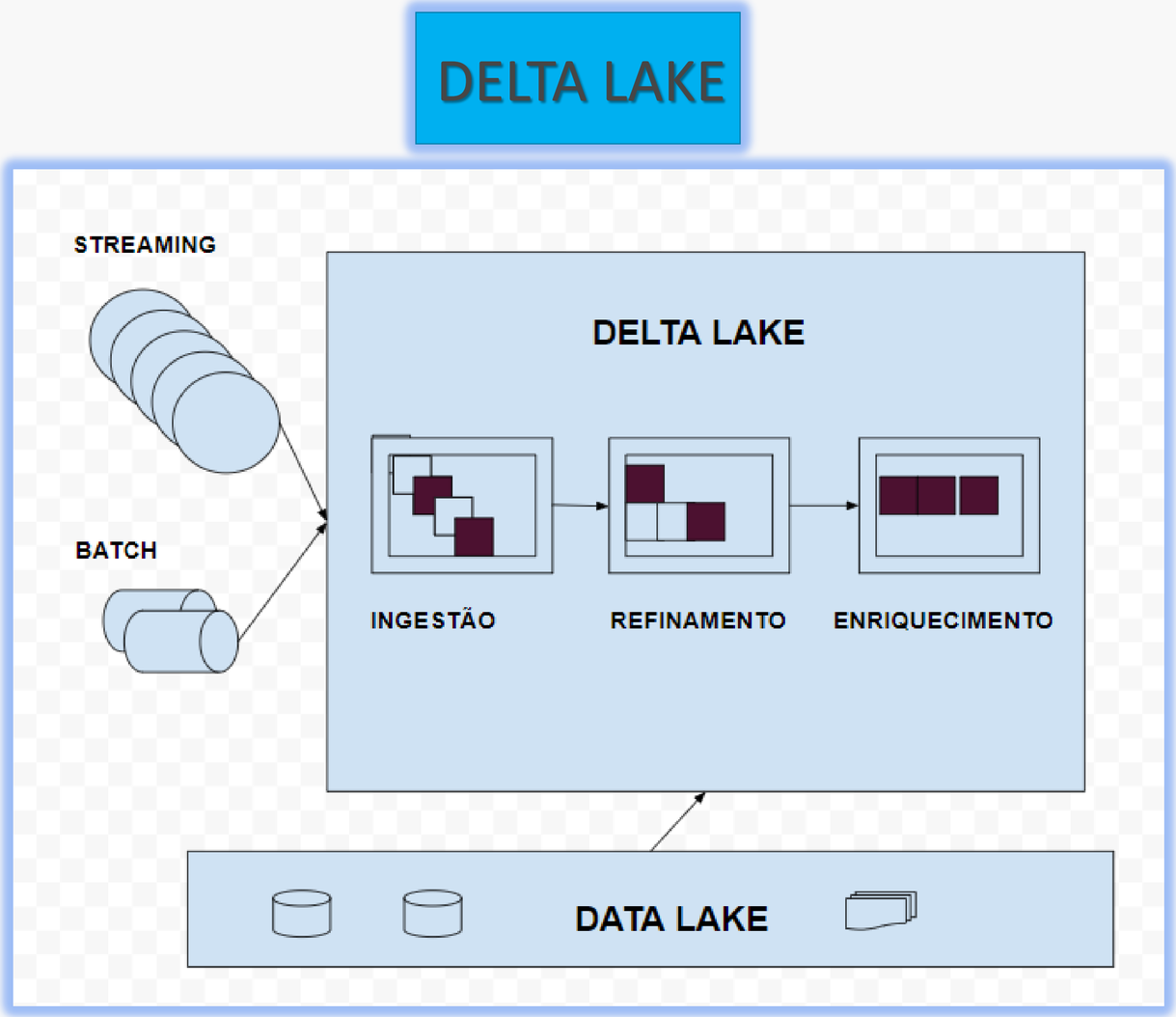
## DATA LAKE



# Ecossistema

Vamos entender as diferenças entre

Data Lake X Delta Lake



# Diferenças

Delta Lake e Data Lake , entenda quais são as diferenças.

## Data Lake

O objetivo principal  é na criação de uma estrutura  que possa centralizar os dados da  organização, sejam eles estruturados ou não ( banco de dados, arquivos texto, vídeos, imagens, etc), com foco na garantia da qualidade e segurança dos dados coletados e armazenados. Fundamental para a tomada de decisões baseada em dados, os Data Lakes precisam ter a certeza de que todas as áreas estão se baseando no mesmo nível de informação.

## Delta Lake

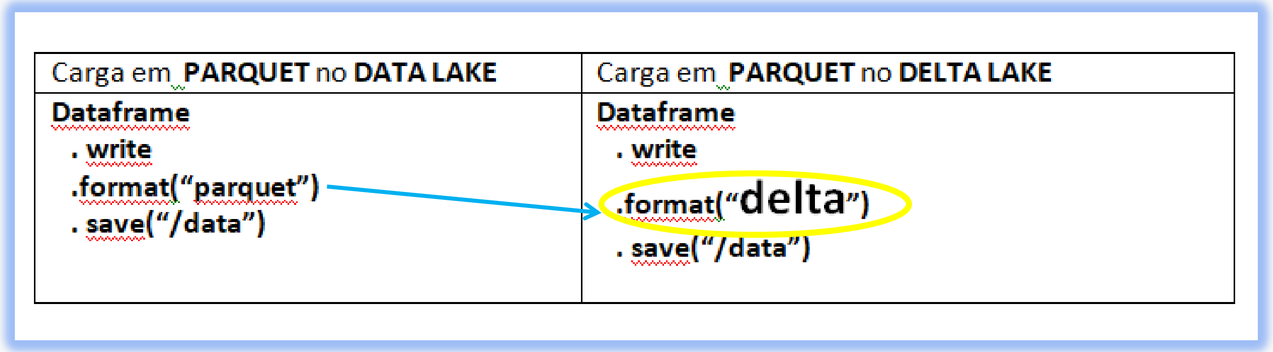
O objetivo principal  é otimização dos processos de coleta e tratamento dos dados, reduzindo o tempo de processamento e descartando o que não for útil para também economizar no armazenamento.

# Delta é simples

Entendimento do funcionamento do Delta Lake

## Delta Lake carga

A criação de uma tabela delta é muito simples, seja a origem destes dados em formatos variados CSV, JSON ou algum outro formato, podemos carregá-los para um dataframe no Spark e então persistí-los como Delta. Este é o formato armazenado em separado ao sistema de armazenamento do Databricks (DBFS).



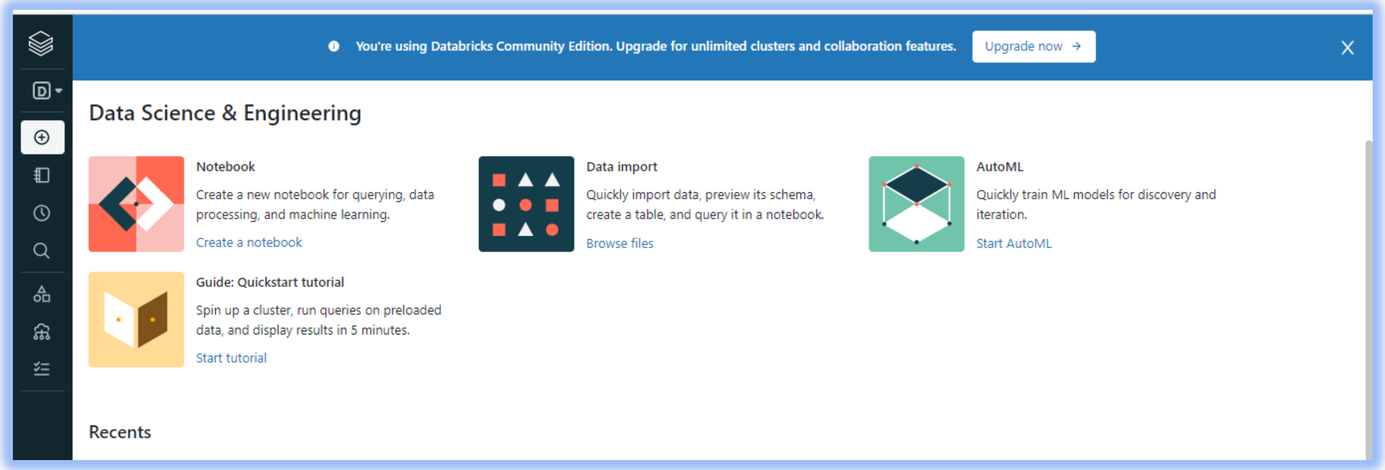

Ao criar uma tabela delta, além dos arquivos parquets que representam os seus dados ou tabela, há a criação de um controle de transações sobre os dados que possibilita explorar todas as features oferecidas pelo Delta Lake.

# Delta Lake

É  local de trabalho dentro do Databricks.

## Workspace

Um espaço de trabalho Databricks é um ambiente para acessar todos os seus ativos Databricks. O espaço de trabalho organiza objetos, e fornece acesso a dados e recursos computacionais, como o clusters.

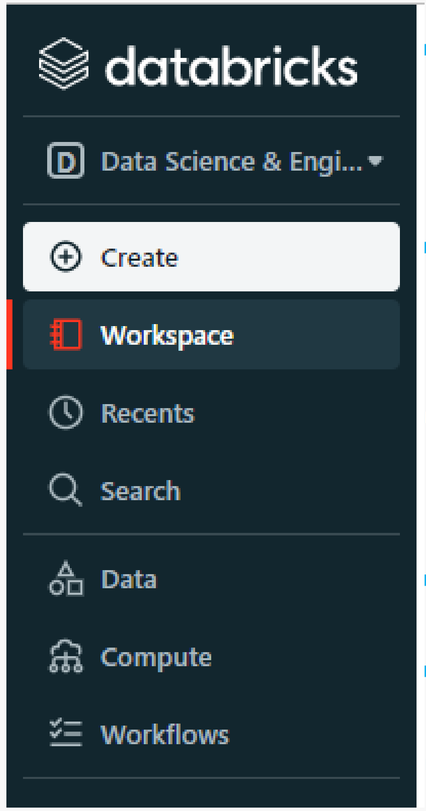



# Workspace

É  local de trabalho dentro do Databricks.

## Workspace

Um espaço de trabalho Databricks é um ambiente para acessar todos os seus ativos Databricks. O espaço de trabalho organiza objetos, e fornece acesso a dados e recursos computacionais, como o clusters.




Retorna a tela de trabalho principal.

Permite a criação de notebooks, tabelas e clusters. Dentre as tarefas permite criar, copiar, deletar.

Permite a criação de tabelas, upload de arquivos, conexões com aws, cassandra, elasticsearch e integrações com ferramentas de dados

Exibe os notebooks recentes criados.

Permite buscar os notebooks criados de forma rápida.

Gerenciamento dos cluster, local para definição e configuração dos clusters, além de informações sobre as tabelas.

Informações sobre o gerenciamento dos clusters

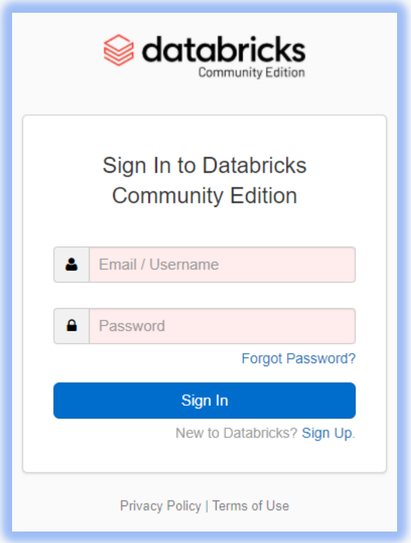
Local de agendamento dos notebooks.

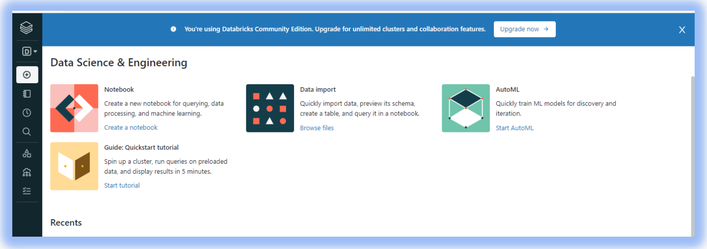
# Criar sua conta

Vamos trabalhar na versão Community gratuita.

## Criando sua conta na Community

Você deverá criar um acesso gratuito na Community (https://community.cloud.databricks.com/login.html)




- Os usuários terão acesso a clusters de 15GB, um gerenciador de clusters e o ambiente de notebook para protótipos de aplicações simples.

- O cluster com os dados fica disponível por 2 horas.

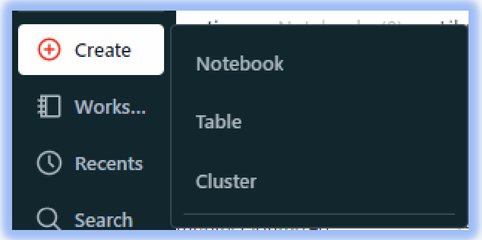
- Limitado a 3 usuários colaborativos.

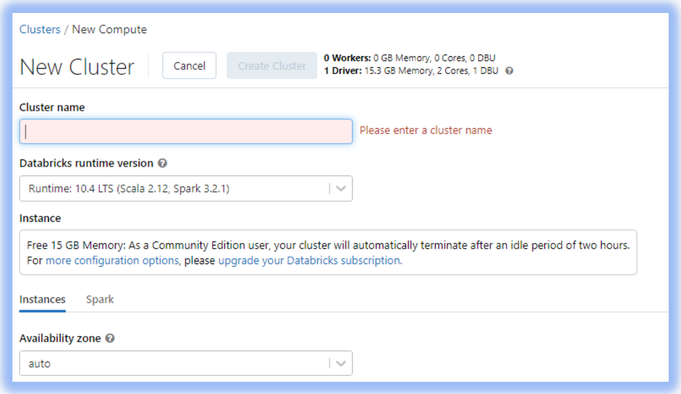
# Começando a trabalhar

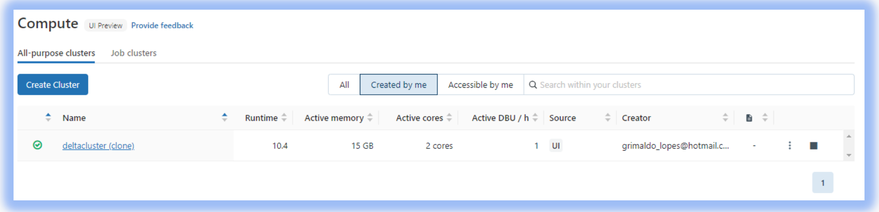
Vamos criar o cluster para iniciarmos nosso trabalho.

## Criando o cluster

Primeiro precisaremos criar uma área para carregar os dados e gerar os nossos notebooks para análise.



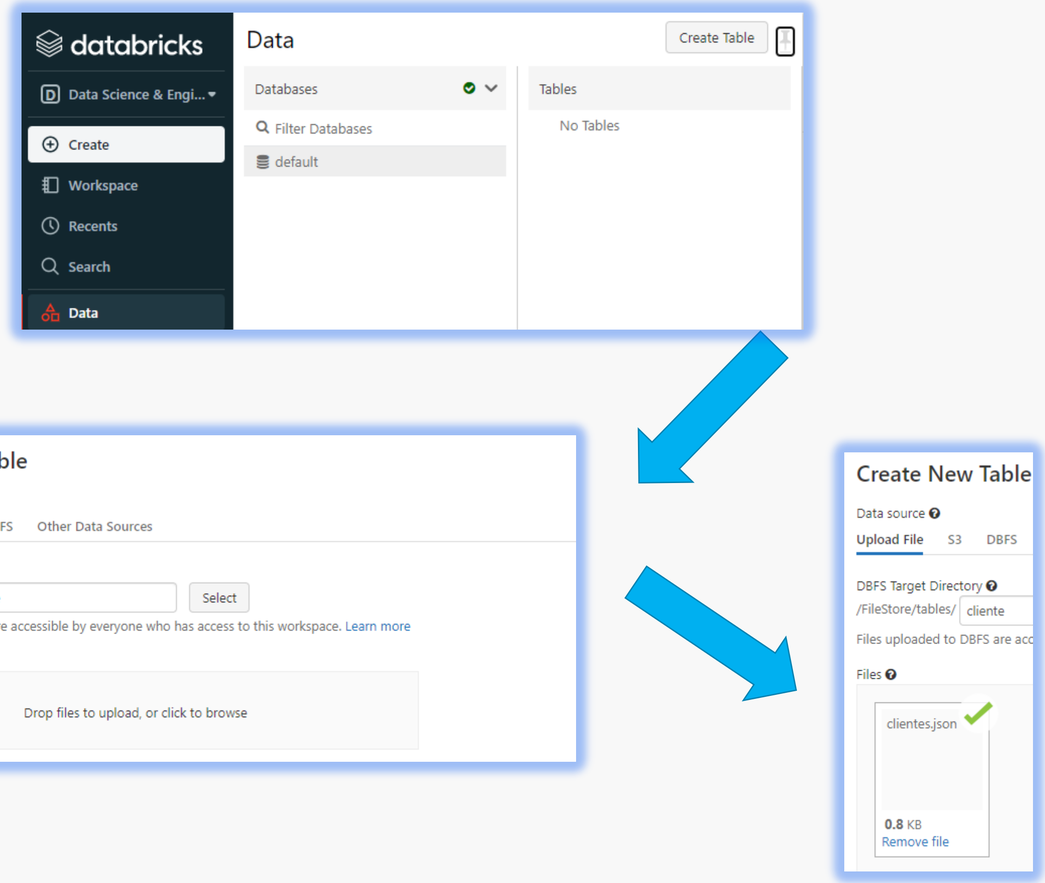



# Começando criar um notebook de trabalho

Local onde analisaremos os dados.

## Carregando os dados

Vamos carregar um pequeno conjunto de dados de cliente e suas compras em .json. Leremos o arquivo **clientes.json**

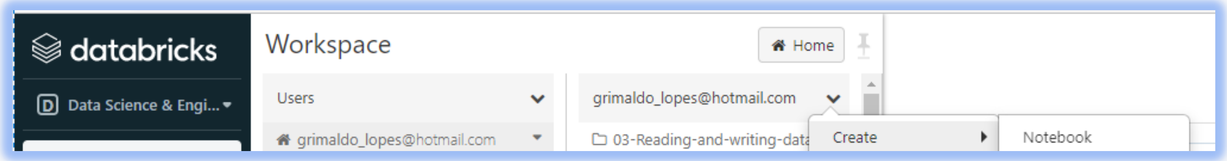


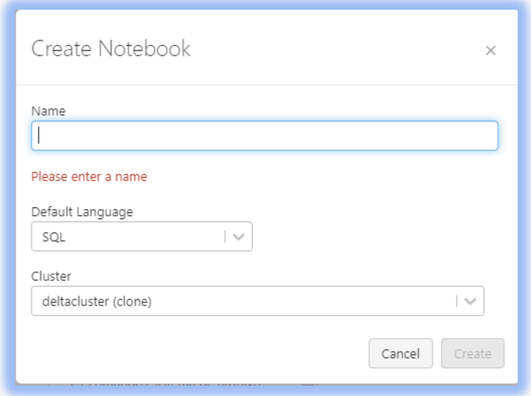
# Começando criar um notebook de trabalho

Local onde analisaremos os dados.

## Criando um notebook

Local que leremos os dados e faremos análises nos dados, podem serem gerados gráficos. Os notebook podem ser em R, Python,Scala e SQL.



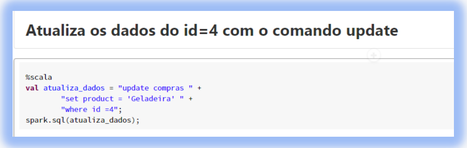
# Começando criar um notebook de trabalho

Local onde analisaremos os dados.

## Criando um notebook

Vamos criar um notebook completo de leitura, carga e transações dos dados no Delta Lake



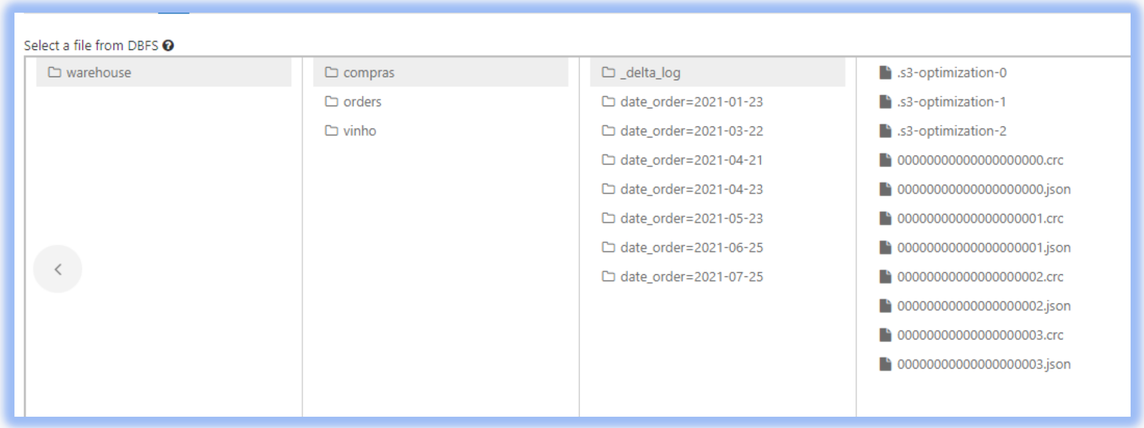



# Começando criar um notebook de trabalho

Local onde analisaremos os dados.

## Log do Delta Lake

Para cada transação ACID é criado um arquivo parquet de controle



# Utilizando comandos utilitários

Podemos agora trabalhar com históricos e comandos para visualização dos dados e suas versões.

## Comandos utilitários

Como ACID, os dados ficam gravados e podemos trabalhar com suas versões

```
%sql
DESCRIBE HISTORY '/user/hive/warehouse/compras'
```

▸ (1) Spark Jobs

| | version | timestamp | userId | userName | operation | operationParameters |
|---|---|---|---|---|---|---|
| 1 | 3 | 2022-09-02T18:51:11.000+0000 | 6174905436003004 | grimaldo_lopes@hotmail.com | DELETE | {"predicate": "[\"(CAST(spark_catalog.default.co |
| 2 | 2 | 2022-09-02T18:49:32.000+0000 | 6174905436003004 | grimaldo_lopes@hotmail.com | UPDATE | {"predicate": "(cast(id#6147 as int) = 4)"} |
| 3 | 1 | 2022-09-02T18:48:14.000+0000 | 6174905436003004 | grimaldo_lopes@hotmail.com | MERGE | {"predicate": "(CAST(spark_catalog.default.comp [{\"actionType\":\"update\"}]", "notMatchedPredic |
| 4 | 0 | 2022-09-02T18:45:59.000+0000 | 6174905436003004 | grimaldo_lopes@hotmail.com | CREATE OR REPLACE TABLE | {"isManaged": "true", "description": null, "partiti |

```
%sql
 SELECT * FROM delta. `/user/hive/warehouse/compras` VERSION AS OF 3
```

▸ (3) Spark Jobs

| | id | date_order | customer | product | unit | price |
|---|---|---|---|---|---|---|
| 1 | 2 | 2021-03-22 | Ana | AR-CONDICIONADO | 6 | 121.6 |
| 2 | 4 | 2021-04-23 | Sandra | Geladeira | 8 | 313 |
| 3 | 5 | 2021-05-23 | Tereza | FRIGOBAR | 3 | 412 |

# Otimização de consulta

Vamos carregar dados sobre  viajantes

## Carregando dados

Vamos carregar registros em csv do arquivo **Datafiniti_Hotel_Reviews_Jun19.csv**

*/FileStore/tables/hotel/Datafiniti_Hotel_Reviews_Jun19.csv*

```
%sql
DROP TABLE IF EXISTS hotel;

-- Carregaato de arquivos Parquet sobre dados de hotel
CREATE TABLE hotel
USING parquet
PARTITIONED BY (Categoria)
SELECT _c0 as Ordem, _c1 as Tipo, _c2 as Situacao, _c3 as Tip_local, _c4 as Categoria, _c5 as Tip_Local2, _c6 as Tip_local3, _c7 as Acomodacao, _c8 as Cidade,
_c9 as Pais, _c10 as Endereco, _c11 as Latitude, _c12 as Longitude, _c13 as Provincia, _c14 as CEP, _c15 as UF, _c16 as Data_estadia, _c17 as Revisao_texto,
_c17 as Revisao_titulo, _c19 as Revisao_cidade, _c20 as Endereco_web, _c21 as Comentario_usuario, _c22 as Resumo, _c23 as Comentario2, _c24 as Comentario3,
_c25 as Comentario4
FROM csv.`dbfs:/FileStore/tables/hotel/Datafiniti_Hotel_Reviews_Jun19.csv`
```

```
%sql
-- OOtimizando a consulta da tabela Delta com o campo Pais, é importante que busque o campo que melhor otimiza
OPTIMIZE hotel ZORDER BY (Pais);
```

# Delta Time Travel

A introdução de recursos de viagem no tempo no Databricks Delta Lake.

## Viagem no tempo (Delta Time Travel )

O mecanismo de análise unificado construído sobre o Apache Spark. Com este novo recurso, a Delta Lake visualiza automaticamente o Big Data onde você armazena seus dados, e você pode acessar qualquer versão histórica desses dados.

O que compõem:

**FACILIDADE**: Esse gerenciamento de dados temporais simplifica seu pipeline de dados, facilitando a auditoria, a reversão de dados em caso de gravações ou exclusões de falhas acidentais e a reprodução de experimentos.

**AUDITORIA**: Organizações que trabalham com sistemas de dados tradicionais para tecnologias de Big Data sempre tem necessidade de auditar  os dados, fundamental tanto em termos de conformidade de dados quanto de depuração simples para entender como os dados mudaram ao longo do tempo.

**REVERSÕES**: Os pipelines de dados ao serem executados,  podem escrever dados não adequados (precisando de refinamento, ajustes), atualizações e exclusões, pode se tornar muito complicado, e os engenheiros de dados normalmente têm que projetar um pipeline complexo, quando não contam com este recurso do Delta Time Travel.

**REPRODUZIR EXPERIMENTOS**: analistas ou cientistas de dados projetam práticas recomendadas criando várias cópias dos dados, levando a um aumento dos custos de armazenamento. Tudo isso para simular a história dos dados.

# Delta Time Travel

Vamos mostrar a passagem no tempo com os  dados sobre  viajantes.

## Carregando dados

Veja como funcionará o controle no tempo

**Contando a quantidade de registros na terceira versão via SQL - Outra forma de realizar a tarefa**

```
%sql
SELECT * FROM delta.`/user/hive/warehouse/compras@v3`
```

**Vamos reinserir o registro com ID=1 que eliminamos, uma forma de realizar o Delta Time Travel**

```
INSERT INTO compras
SELECT * FROM compras VERSION AS OF 1
WHERE Id = 1
```

**Verificando quantos registro é a diferença da versão atual, para a versão 3**

```
%sql
SELECT count(distinct ID) - (SELECT count(distinct ID) FROM compras VERSION AS OF 3) as `Diferença de registros`
FROM compras
```

# Tabelas e comandos

Vamos mostrar que é possível realizar a criação de tabelas com restrições.

## Carregando dados

Como criar tabelas, fazer consultas e colocar restrições.

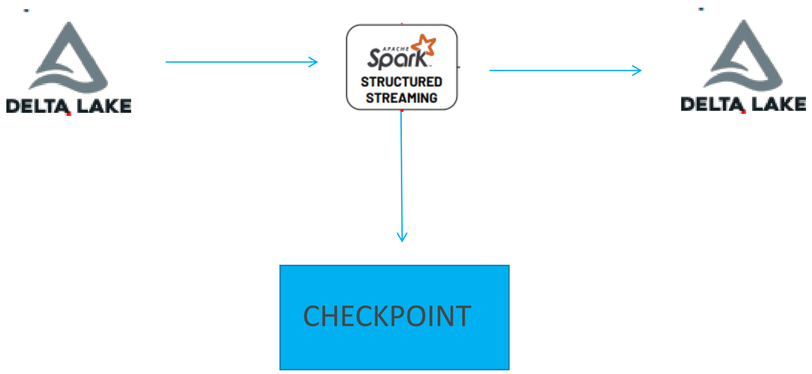
**Criação das tabelas TB_AUTOR,TB_LIVRO e a associativa TB_LIVRO_AUTOR**

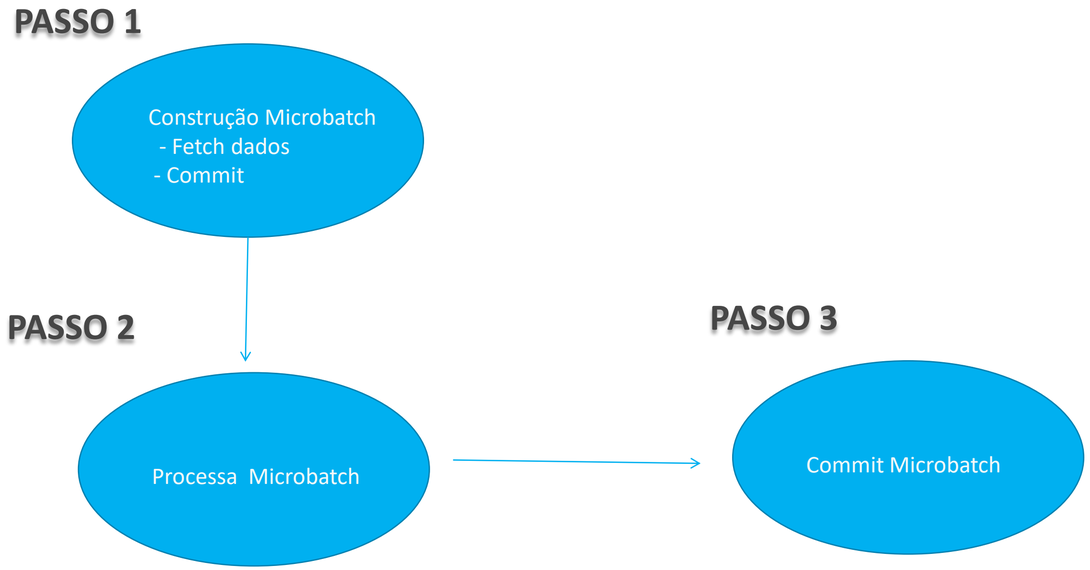
```
%scala
val Tscrisql = "CREATE OR REPLACE TABLE TB_AUTOR (ID_AUTOR DOUBLE,NOME STRING,SEXO STRING,DATA_NASCIMENTO STRING) USING DELTA ;"
spark.sql(Tscrisql);
val Tscrisql1 =  "CREATE OR REPLACE TABLE TB_LIVRO (ID_LIVRO DOUBLE,ISBN STRING,TITULO STRING,EDICAO DOUBLE,PRECO DOUBLE,QTDE_ESTOQUE DOUBLE) USING DELTA;"
spark.sql(Tscrisql1);
val Tscrisql3 = "CREATE OR REPLACE TABLE TB_LIVRO_AUTOR (ID_LIVRO_AUTOR DOUBLE,ID_LIVRO DOUBLE,ID_AUTOR DOUBLE) USING DELTA;"
spark.sql(Tscrisql3);
```

**Selecionando os registros, ligando todas as tabelas**

```
%sql
SELECT TB_LIVRO.TITULO, TB_LIVRO.ISBN,TB_LIVRO.PRECO,TB_AUTOR.NOME,TB_AUTOR.DATA_NASCIMENTO FROM TB_LIVRO
  INNER JOIN TB_LIVRO_AUTOR ON TB_LIVRO.ID_LIVRO = TB_LIVRO_AUTOR.ID_LIVRO
  INNER JOIN TB_AUTOR ON TB_AUTOR.ID_AUTOR = TB_LIVRO_AUTOR.ID_AUTOR
```

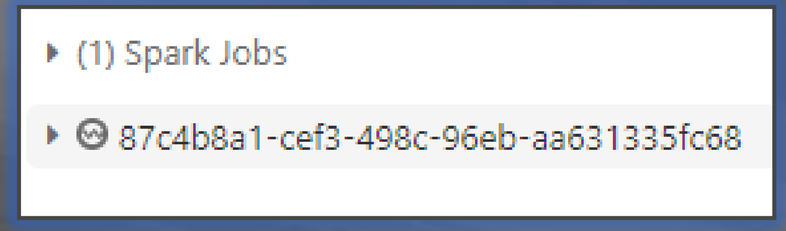
```
%sql
ALTER TABLE TB_AUTOR DROP CONSTRAINT validIds;
ALTER TABLE TB_AUTOR ADD CONSTRAINT validIds CHECK (ID_AUTOR> 0 and ID_AUTOR < 5);
SHOW TBLPROPERTIES TB_AUTOR;
```




# Pontos Principais carga Streaming

**Status no Notebook**



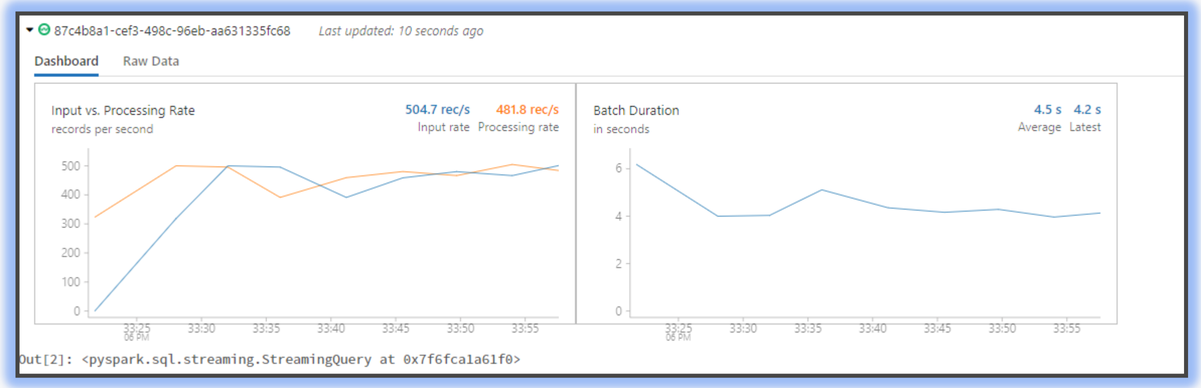
# Streaming em tabelas Delta

Vamos mostrar como podemos trabalhar com streaming de dados e armazenar em tabelas Delta.

## Streaming de Dados

Executaremos um pipeline para armazenamento de dados oriundos de streaming.

```
%python
#Lendo um dos arquivos JSON
dataf3 = spark.read.json("/databricks-datasets/structured-streaming/events/file-1.json")
dataf3.show()
```




```
%sql select distinct action, count(*) from db_stream.tab_stream
 group by action
```

(3) Spark Jobs

| | action | count(1) |
|---|---|---|
| 1 | Open | 14499 |
| 2 | Close | 13501 |

# Schema Evolution em tabelas Delta

Vamos entender como podemos acompanhar as mudanças de campos em tabelas Delta.

## Schema Evolution

Ajustes de campos em tabelas Delta

O que compõem:

**Schema de Tabelas**: Cada Dataframe contém um esquema, que define a forma dos dados, como tipos de dados e colunas, e metadados. Com Delta Lake, o esquema da tabela é salvo no formato JSON dentro do registro de transações.

**Schema Evolution**: É um recurso que permite que os usuários alterem facilmente o esquema atual de uma tabela para acomodar dados que estão mudando com o tempo. Geralmente utilizado quando queremos acrescentar uma ou mais colunas(campos).

**Como Aplicar**: A evolução do esquema é ativada adicionando ao seu script o comando**. .option('mergeSchema', 'true').**

**Por que utilizar**: Pode ser usada sempre que você pretende alterar o esquema da sua tabela. Engenheiros de dados e cientistas podem usar essa opção para adicionar novas colunas sem quebrar modelos existentes que dependem das colunas antigas.

# Schema Evolution

Veja como funciona

## Schema Evolution

| Funcionario | Salario |
|---|---|
| Joao Santos | 2000 |
| Carlos Fernandez | 3400 |

| Setor | Comissao |
|---|---|
| Financeiro | 240 |
| Marketing | 540 |

| Funcionario | Salario | Setor | Comissao |
|---|---|---|---|

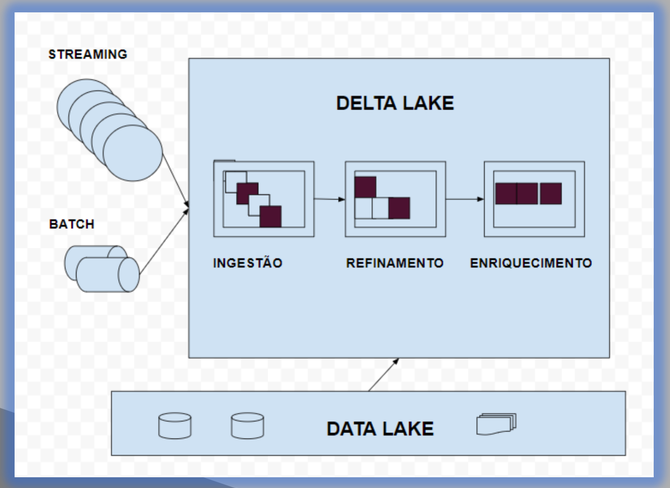
# PRÁTICA

**ENVIE AO PROFESSOR**
**grimaldo_lopes@hotmail.com**

**PREPARE UM ESTUDO**

Carregar qualquer arquivo do site **kaggle.com** e criar uma tabela Delta e demonstrar como funciona o Delta Time Travel, gerando insert, delete, update e os arquivos do delta_log.

Muito boa sorte e conte comigo!




**Selecionando os registros, ligando todas as tabelas**

```
%sql
SELECT TB_LIVRO.TITULO, TB_LIVRO.ISBN,TB_LIVRO.PRECO,TB_AUTOR.NOME,TB_AUTOR.DATA_NASCIMENTO FROM TB_LIVRO
  INNER JOIN TB_LIVRO_AUTOR ON TB_LIVRO.ID_LIVRO = TB_LIVRO_AUTOR.ID_LIVRO
  INNER JOIN TB_AUTOR ON TB_AUTOR.ID_AUTOR = TB_LIVRO_AUTOR.ID_AUTOR
```

```
%sql
DROP TABLE IF EXISTS hotel;

-- Carregaato de arquivos Parquet sobre dados de hotel
CREATE TABLE hotel
USING parquet
PARTITIONED BY (Categoria)
SELECT _c0 as Ordem, _c1 as Tipo, _c2 as Situacao, _c3 as Tip_local, _c4 as Categoria, _c5 as Tip_Local2, _c6 as Tip_local3, _c7 as Acomodacao, _c8 as Cidade,
_c9 as Pais, _c10 as Endereco, _c11 as Latitude, _c12 as Longitude, _c13 as Provincia, _c14 as CEP, _c15 as UF, _c16 as Data_estadia, _c17 as Revisao_texto,
_c17 as Revisao_titulo, _c19 as Revisao_cidade, _c20 as Endereco_web, _c21 as Comentario_usuario, _c22 as Resumo, _c23 as Comentario2, _c24 as Comentario3,
_c25 as Comentario4
FROM csv.`dbfs:/FileStore/tables/hotel/Datafiniti_Hotel_Reviews_Jun19.csv`
```