Aula 2 - Databricks

Lucio Monteiro

O que é?

O **Databricks**, desenvolvido pelos mesmos criadores do Apache Spark (framework que discutiremos no futuro), é uma plataforma de análise de dados unificada para engenharia de dados, machine learning e ciência de dados colaborativa. Se organiza através de workspaces, que são ambientes de software como serviço (SaaS) utilizados para acessar suas funcionalidades e objetos, como notebooks, em pastas e recursos computacionais, como clusters.

Por que utilizar Databricks?

Workspace colaborativo

Uma primeira razão para utilizar **Databricks** está em seus workspaces colaborativos, onde usuários diferentes podem compartilhar seus notebooks ou até mesmo trabalhar em um de forma simultânea, experiência similar à do Google Docs ou Word Online.

Para Cientistas, Analistas e Engenheiros de Dados

Databricks permite que cientistas de dados manipulem dados e criem modelos para Machine Learning; que analistas usem tabelas estruturadas, arquivos semi-estruturados e se conecte a ferramentas de visualização nativamente; para engenheiros de dados é possível extrair, transformar e carregar dados em ambientes diferentes ambientes. Sendo a plataforma então poderosa e agradável de se utilizar para diferentes profissionais.

Pluralidade de Linguagens

Os notebooks **Databricks**, similares ao do Jupyter, não se limitam a uma só linguagem de programação: neles é possível utilizar *Python*, *SQL*, *R* e *Scala*. Sendo possível, inclusive, mesclar em um mesmo notebook células em linguagens diferentes e combiná-las posteriormente, como por exemplo criando uma função em Python em uma primeira e a utilizando em um código SQL em uma segunda.

A pluralidade de linguagens resolve diversos problemas que times enfrentam ao definir arquitetura e escolher produtos. Se parte dos cientistas for mais familiarizado com Python e outros com R, tudo certo; se alguns engenheiros preferem utilizar Spark em Scala e outros em Python, tudo certo também. Evita-se assim uma necessidade de adaptar e treinar elementos do time a novas skills.

Multi-cloud

Databricks é uma ferramenta que pode ser considerada agnóstica por ser multicloud, em outras palavras, se conecta e se complementa muito bem a diferentes plataformas de computação em nuvem, como Amazon Web Services (AWS), Google Cloud Plataform (GCP) e Microsoft Azure. Sendo assim, independentemente da plataforma escolhida para sediar a arquitetura de uma solução, é possível encaixar produtos do Databricks em uma ou várias etapas.

Versão Community

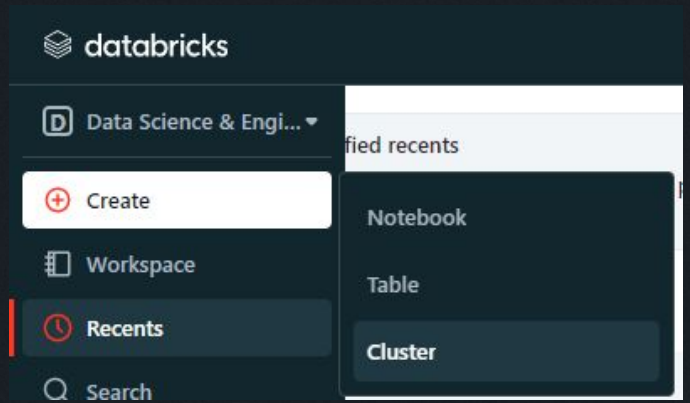
O **Databricks** possui uma versão Community para que estudantes e interessados na plataforma possam degustar e praticar em uma versão gratuita, ainda que com limitações. Outro aspecto importante é que diferente de suas parceiras, AWS Azure e GCP, a versão community é totalmente gratuita, o que está disponível apenas em versões pagas, simplesmente não funcionam. Ou seja, não é preciso inserir um cartão de crédito e fazer um rígido controle dos produtos utilizados para evitar gastos inesperados enquanto está apenas experimentando.

Primeiro Acesso

https://community.cloud.databricks.com/login.html?next_url=%2Frecents%3Fo%3D4084783085078408

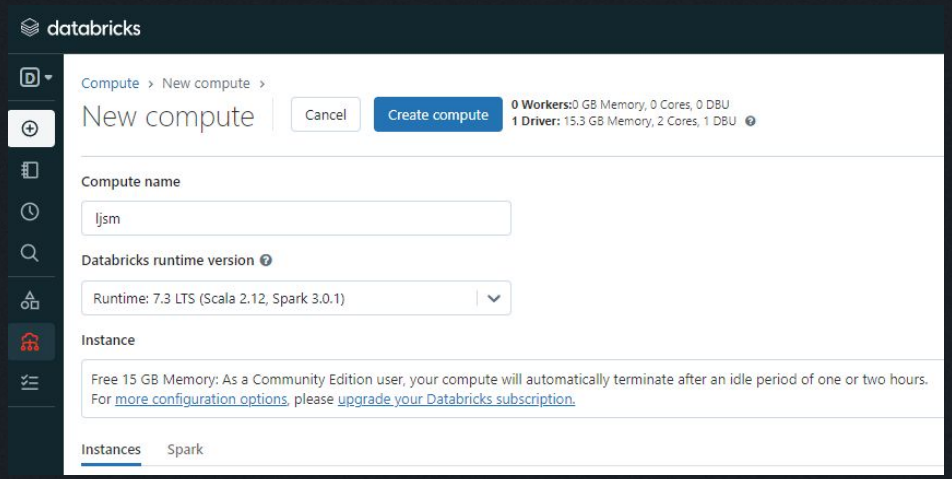
Cluster

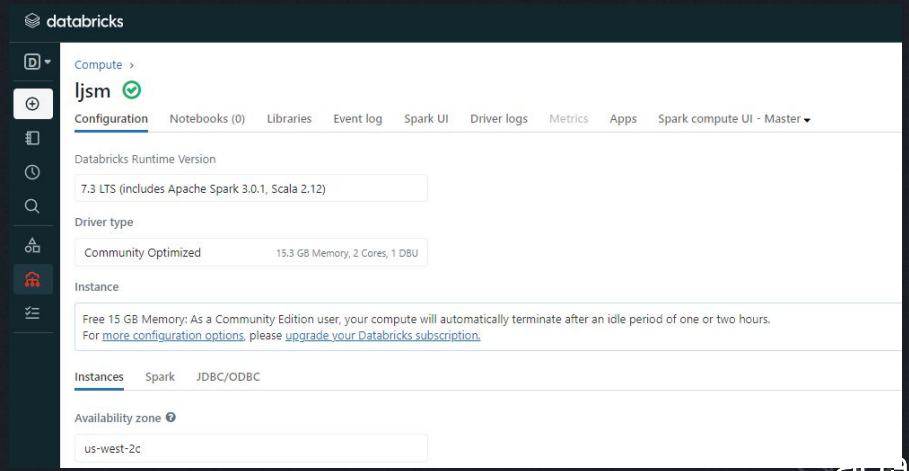
Podemos criar um cluster no botão "Create Cluster", dando um nome qualquer a ele e mantendo as próximas configurações padrões. Caso o projeto que venha a desenvolver requeira uma versão específica do Spark ou Scala, basta alterar durante a criação. Quanto ao poder computacional (quantidade de nós, memória etc.), não podemos alterá-lo na versão community.



Databricks

Cluster



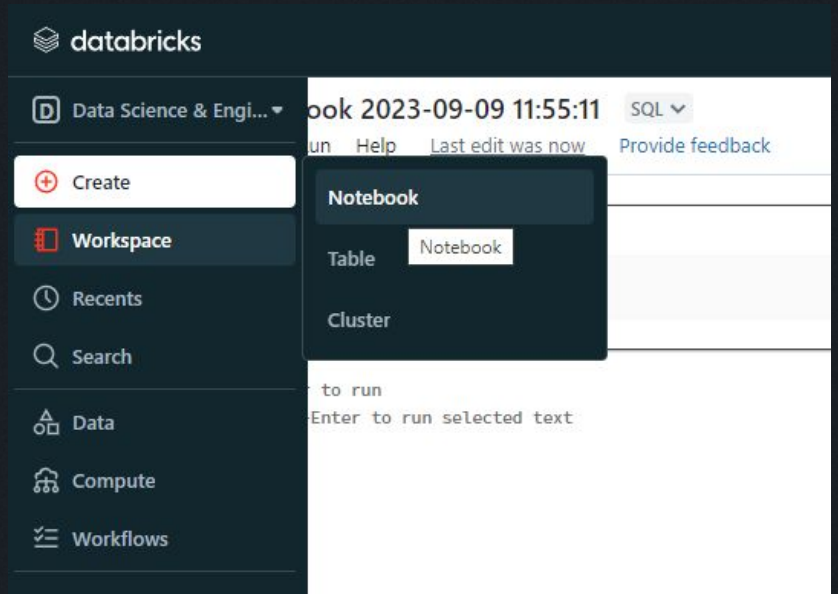


Notebook

> "Um **notebook** é uma interface baseada na Web para um documento que contém código executável, visualizações e texto de narração."

Composto por **células** capazes de executar códigos em diferentes linguagens, é em um notebook que somos capazes de manipular arquivos, criar funções e tabelas etc.

Caso tenha utilizado *Jupyter* anteriormente, saiba que o conceito é o mesmo, assim como a extensão do arquivo .ipynb.



Notebook

Note que, ao lado de seu nome, temos a sua linguagem default (Python), o que quer dizer que por padrão as células esperarão códigos em Python. Abaixo do nome do notebook selecionamos o cluster em que os códigos serão executados (lembrando que a versão community do **Databricks** nos disponibiliza um gratuitamente). Podemos criar quantas células julgarmos necessárias e, para mudar a linguagem de uma individualmente basta selecionar no menu à direita da célula ou em sua primeira linha colocar '#' seguido do nome da linguagem.

Um outro recurso interessante para estudos futuros é acionado no canto superior direito do notebook e diz respeito a versionamento, pois o **Databricks** permite integração com git em versões pagas. (Settings -> User Settings -> Git Integration)

DBFS - Databricks File System

O **DBFS** funciona como uma espécie de HDFS do Hadoop, só que no Databricks. Aqui é possível inserir arquivos e manipulá-los através de comandos similares aos do HDFS. Além disso, diretórios podem ser utilizados para salvar arquivos frutos de processamentos e transformações.

Há uma interface visual para navegar pelo DBFS, para ativá-la navegue em: Seu nome (e-mail) -> Admin Settings -> Workspace settings -> DFBS File Browser: Enabled. Ao atualizar a página, ela estará disponível em uma aba no menu Data à esquerda.

Vale ressaltar que o DBFS é uma ferramenta interessante para a versão community e para quem está degustando/estudando a plataforma. Todavia, para arquiteturas profissionais, a própria Databricks recomenda o uso de soluções como Blob Storage da Azure ou S3 da Amazon para armazenamento de dados.

Ele é transparente para o usuário, visto que, mesmo quando estiver trabalhando com um cluster que tenha nós works, e os arquivos estiver distribuídos entre os nós works, o nó master fará todo trabalho de distribuir ou resgatar dados entre os cluster para você. E o mais legal, ele utiliza semântica de diretórios, ou seja, é como se estivesse trabalhando em diretórios de uma máquina local. Não precisa ficar se preocupando com URLs de arquivos.

dbutils

Databricks Utilities é composto por um conjunto de comandos que podem ser utilizados para trabalhar com armazenamentos de objetos de forma eficiente. Similar ao HDFS DFS do Hadoop (antigo Hadoop fs), está disponível em Notebooks Python, Scala e R.

Comandos

Lista de comandos e orientações:

```
dbutils.fs.help()
```

Criar diretório:

```
dbutils.fs.mkdirs(<caminho até o diretório>)
```

Listar conteúdo de um diretório:

```
dbutils.fs.ls(<caminho até o arquivo/diretório>)
```

Copiar um arquivo ou diretório:

```
dbutils.fs.cp(<caminho até o arquivo/diretório>, <diretório destino>, recursivo (True ou False))
```

Mover um arquivo ou diretório:

```
dbutils.fs.mv(<caminho até o arquivo/diretório>, <diretório destino>, recursivo (True ou False))
```

Deletar um arquivo ou diretório:

```
dbutils.fs.rm(<caminho até o arquivo/diretório>, recursivo (True ou False))
```

Criar arquivo txt:

```
dbutils.fs.put(<caminho até o arquivo>, conteúdo, overwrite (True or False))
```

Retornar os primeiros n bytes de um arquivo:

```
dbutils.fs.ls(<caminho até o arquivo>, quantidade de bytes)
```

Limitações

Algumas limitações que você encontrará ao utilizar a versão **community** do **Databricks** que valem ser citadas:

- O cluster disponibilizado, de forma gratuita, é **desligado** após duas horas de inatividade e não pode ser reinicializado. Com isso, embora arquivos no DBFS e notebooks não sejam perdidos, metadados como tabelas criadas sim.
- Conexões com serviços de armazenamento (mount), como S3 e Blob Storage, utilizando chaves de segurança não são permitidas.

Prática - 1

1) Entrar no site https://databricks.com//
2) Crie um login - Versão gratuita - Community Edition
3) Clique em criar -> Cluster -> Escolher um nome e esperar executar
4) Baixar em CSV o arquivo que esta em:

   https://dadosabertos.bndes.gov.br/dataset/operacoes-financiamento/resource/332d446e-d340-46ef-af64-ee6f36e7bd50

5) Renomear o arquivo que foi feito download
6) Create -> Table -> Upload FIle
7) Jogar o arquivo baixado na pasta Files e esperar o upload
8) Clique no botão Create Table with UI
9) Selecione o cluster em execução e clique em preview table
10) marque a opção First row is header
11) Create Table

Prática - 2

1) Entrar no site https://databricks.com//
2) Crie um login - Versão gratuita - Community Edition
3) Clique em criar -> Cluster -> Escolher um nome e esperar executar
4) Criar um Notebook - Create Notebook - trocar o nome e a linguagem para Python e selecionar o cluster
5) Listar arquivos do diretório FileStore/tables do dbfs

```
display(dbutils.fs.ls("dbfs:/FileStore/tables"))
```

6) Criando um novo diretorio

```
dbutils.fs.mkdirs("dbfs:/FileStore/tables/bndes")

display(dbutils.fs.ls("dbfs:/FileStore/tables"))
```

7) Movendo arquivo ou diretorios

```
dbutils.fs.mv("dbfs:/FileStore/tables/fin.csv", "dbfs:/FileStore/tables/bndes")

display(dbutils.fs.ls("dbfs:/FileStore/tables/bndes/"))
```

8) Copiar arquivos ou diretorios

```
dbutils.fs.cp("dbfs:/FileStore/tables/bndes/fin.csv",
"dbfs:/FileStore/bndes2/financiamento_nao_automaticas.csv")
display(dbutils.fs.ls("dbfs:/FileStore/bndes2/"))
```

Prática - 2

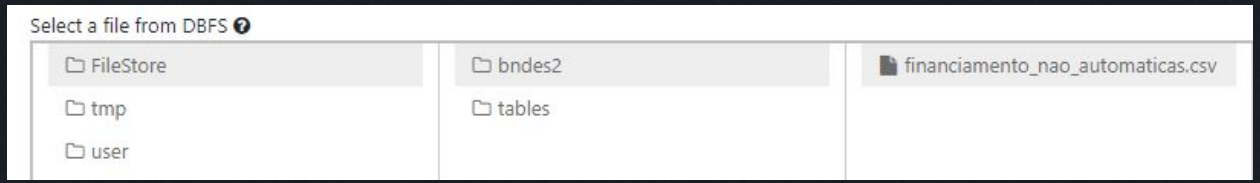
9) Apagando um arquivo

```
dbutils.fs.rm("dbfs:/FileStore/tables/bndes/fin.csv")
```

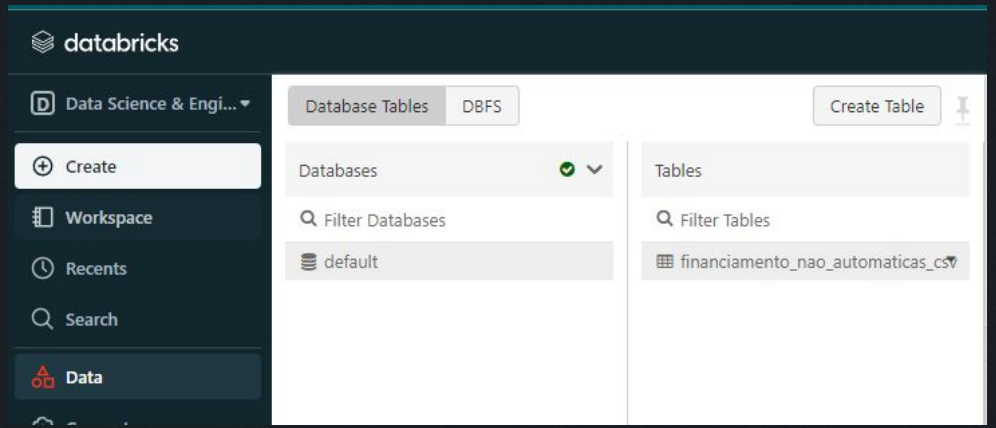
```
dbutils.fs.rm("dbfs:/FileStore/tables/bndes", recurse=True)
```

Prática - 3

1) Entrar no site https://databricks.com//
2) Crie um login - Versão gratuita - Community Edition
3) Clique em criar -> Cluster -> Escolher um nome e esperar executar
4) Criar um Notebook - Create Notebook - trocar o nome e a linguagem para Python e selecionar o cluster
5) Create -> Table -> DBFS
6) Seleciona as pastas e o arquivo que fez upload:



7) Create Table with UI
8) Selecione o cluster
9) Preview Table
10) Column Delimiter ‘,’
11) First row is header
12) Create Table



ada

Prática - 4

1) Entrar no site https://databricks.com//
2) Crie um login - Versão gratuita - Community Edition
3) Clique em criar -> Cluster -> Escolher um nome e esperar executar
4) Baixar em CSV o arquivo que esta em:

   https://dadosabertos.bndes.gov.br/dataset/operacoes-com-entes-da-administracao-publica-direta/resource/1090d538-d01c-4433-909b-67151ed7eb08

5) Renomear o arquivo que foi feito download
6) Create -> Table -> Upload FIle
7) Jogar o arquivo baixado na pasta Files e esperar o upload
8) Clique no botão Create Table in Notebook
9) Trocar os parâmetros para

```
# CSV options
infer_schema = "true"
first_row_is_header = "true"
delimiter = ","
```

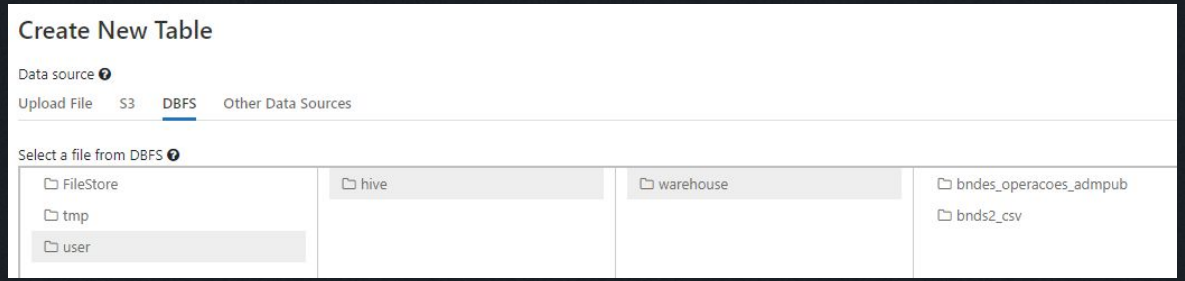
10) Execute as Células
11) Descomente a linha:

```
df.write.format("parquet").saveAsTable(permanent_table_name)
```

12) Execute a Célula

Prática - 4

13)



14) Execute as consultas em uma célula por vez:

```
spark.sql("select ente_publico , data_do_nivel_atual , valor_desembolsado_em_reais from bnds2_csv").show(5)
```

```
SELECT * FROM bnds2_csv WHERE uf == 'BA'
```

```
SELECT * FROM bnds2_csv WHERE nivel_atual == 'EM ANALISE'
```

Indicações e Bibliografias

- Databricks Community
- Documentação oficial
- Databricks getting started
- Databricks na AWS
- Comandos para manipulação do DBFS
- S3 vs HDFS
- https://docs.databricks.com/pt/dev-tools/databricks-utils.html

Obrig.ada