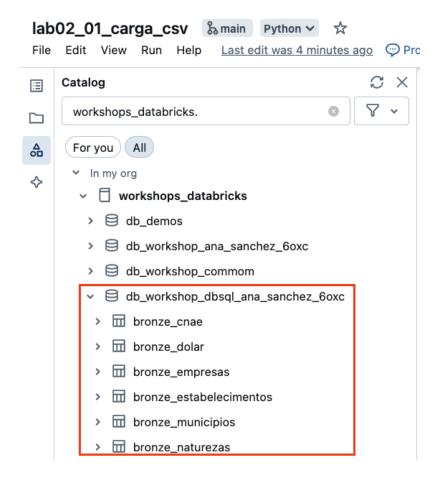
# Semana 02 - DBSQL

Hands-On Lab 00 - Carregar algumas tabelas previas com Python	3
Hands-On Lab 01 - Criação de tabelas com DBSQL	4
Objetivo do exercício	4
Sessão 01: Estrutura TABELAS, DATABASE e CATÁLOGOS	4
Exercício 01.01 - Criação da tabela	5
Exercício 01.02 - Inserindo dados na Tabela através de SQL INSERT	6
Exercício 01.03 - Verificando o conteúdo da TABELA	7
Exercício 01.04 - Alterando o conteúdo da TABELA	7
Exercício 01.05 - Visualizando o Histórico de Atualizações da tabela	9
Exercício 01.06 - Visalizando o conteúdo da tabela na versão anterior (Time Travel)	10
Exercício 01.07 - Restaurando o conteúdo da tabela na versão anterior	10
Exercício 01.08 - Visualizando as propriedades da Tabela	10
Exercício 01.09 - Visualizando as informações detalhadas da Tabela	10
Hands- On 02 - Constraints	11
Exercício 02.01 - Campos NOT NULL	11
Exercício 02.02 - Check Constraint - Garantir a Qualidade dos Dados	12
Exercício 02.03 - PK Constraint - Chave Primaria	14
Exercício 02.01 - FK Constraint - Chave estrangeira	16
Hands-On Lab 03 - Criação da camada Silver	21
Objetivo do exercício	21
Exercício 03.01 - Criação da tabela Silver	21
Exercício 03.02 - Adicionando comentários	22
Exercício 03.03 - Carregando Dados JSON	23
Exercício 03.04 - Criando Streaming Tables	23
Exercício 03.06 - Criando Materialized Views	24
Hands-On Lab 04 - Filtrando e Mascarando linhas e colunas	25
Exercício 04.01 - Filtrando os registros	25
Exercício 04.02 - Mascarando os registros	28
Hands-On Lab 05 - Criação de Alertas	30
Objetivo do exercício	30
Exercício 05.01 - Criação da Query	30
Exercício 05.02 - Criando o Alerta	31
Hands-On Lab 06 - DBSQL API	39
Objetivo do exercício	39
Exercício 06.01 - Testes com o DBSQL API	39
Hands-On Lab 07 - Criando Dashboards	40
Objetivo do exercício	40
Exercício 07.01 - Criando o dashboard	40

40
42
45
47
50
50
50

# Hands-On Lab 00 - Carregar algumas tabelas prévias com Python

- 1. Dentro das pastinhas do repositório que você importou, execute o arquivo lab02\_01\_carga\_csv dentro do diretório 00\_Load Data.
- 2. Verifique se as tabelas foram criadas com sucesso.

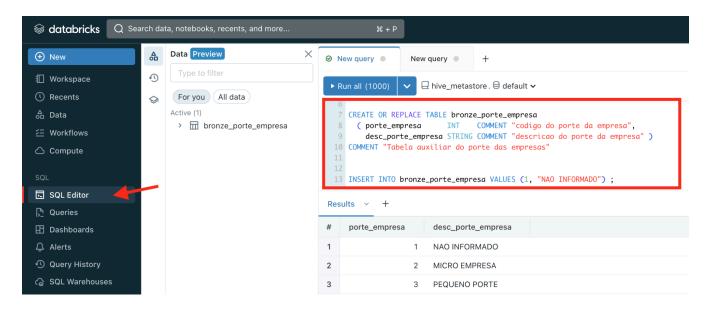


# Hands-On Lab 01 - Criação de tabelas com DBSQL

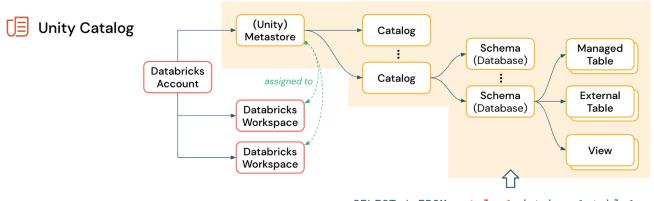
### Objetivo do exercício

O objetivo desse laboratório é conhecer as funcionalidades de consulta (Query) da plataforma Databricks, utilizando a linguagem SQL (e as interfaces visuais), explorando os potenciais Analíticos.

Os exercícios deverão ser executados na opção do Menu lateral "SQL Editor".



# Sessão 01: Estrutura TABELAS, DATABASE e CATÁLOGOS



SELECT \* FROM catalog1.database1.table1;

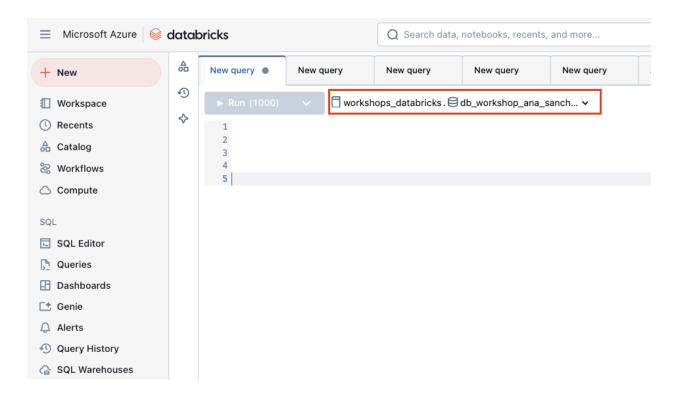
Tópico	Comando	
Catálogo	CREATE CATALOG <nome_catalogo></nome_catalogo>	
Schema	CREATE DATABASE IF NOT EXISTS <nome_catalogo>.<nome_database></nome_database></nome_catalogo>	
Tabela	CREATE OR REPLACE TABLE <nome_catalogo>.<nome_database>.<nome_tabela></nome_tabela></nome_database></nome_catalogo>	
View	CREATE OR REPLACE VIEW <nome_catalogo>.<nome_database>.<nome_view></nome_view></nome_database></nome_catalogo>	

#### Referência:

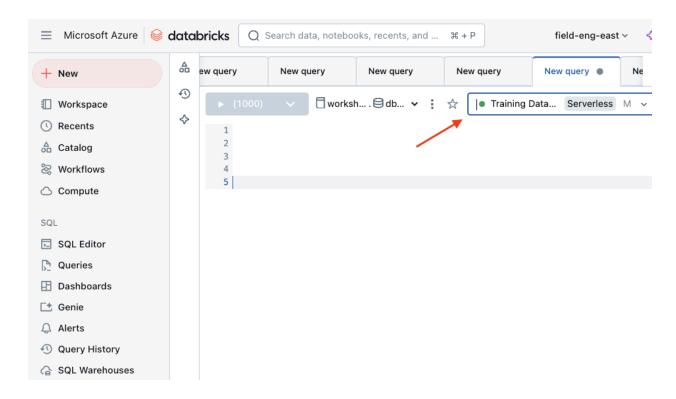
<u>Databricks Help - DDL Syntax</u>

#### Exercício 01.01 - Criação da tabela

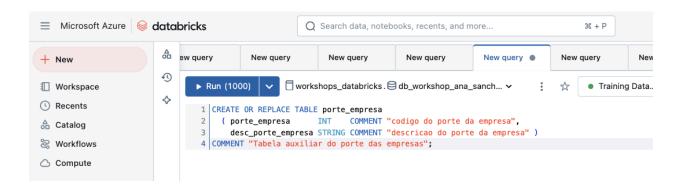
Selecione seu catálogo (workshops\_databricks) e seu schema (db\_workshop\_SEUNOME\_OU\_SUACHAVE) no topo do Query Editor:



Selecione o warehouse Databricks Training:



Digite a query abaixo e clique em Run:



Exercício 01.02 - Inserindo dados na Tabela através de SQL INSERT

```
INSERT INTO porte_empresa VALUES (1, "NAO INFORMADO");
```

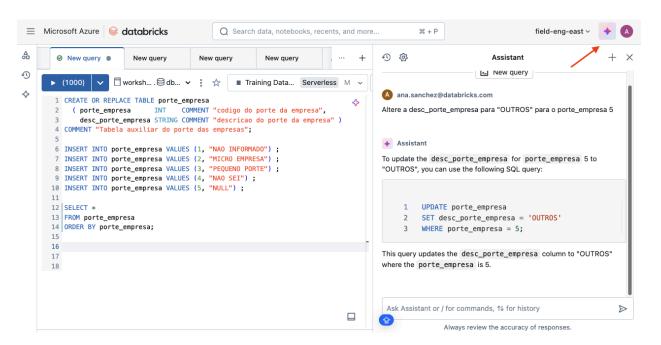
```
INSERT INTO porte_empresa VALUES (2, "MICRO EMPRESA");
INSERT INTO porte_empresa VALUES (3, "PEQUENO PORTE");
INSERT INTO porte_empresa VALUES (4, "NAO SEI");
INSERT INTO porte_empresa VALUES (5, "NULL");
```

#### Exercício 01.03 - Verificando o conteúdo da TABELA

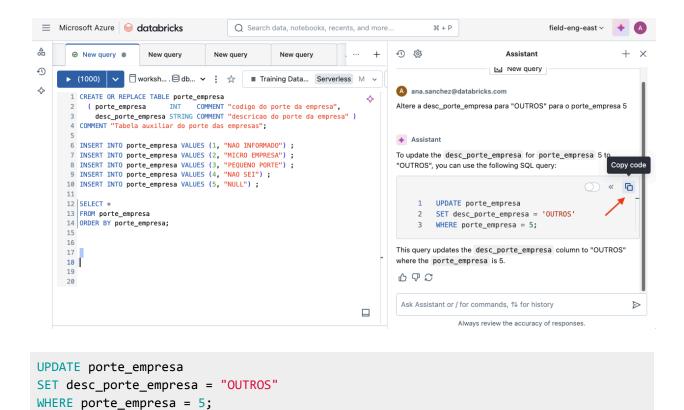
```
SELECT *
FROM porte_empresa
ORDER BY porte_empresa;
```

#### Exercício 01.04 - Alterando o conteúdo da TABELA

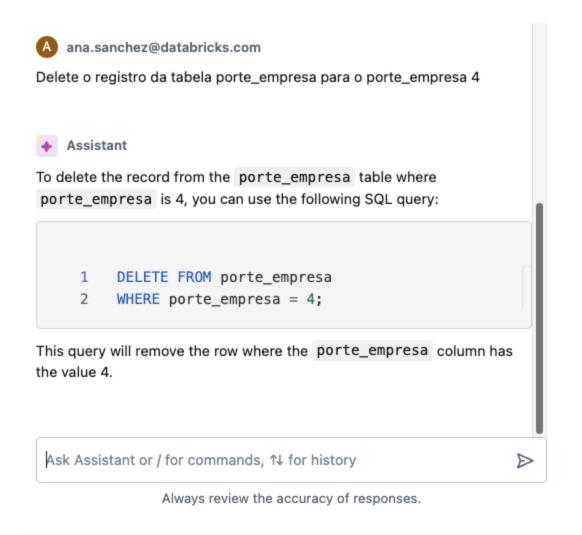
Clique no assistante e peça para ele: Altere a desc\_porte\_empresa para "OUTROS" para o porte empresa 5



Arraste o mouse até o quadro cinza onde contém a resposta e clique no botão apresentado abaixo para copiar o código gerado:



Agora repita o mesmo processo para fazer o seguinte pedido: Delete o registro da tabela porte\_empresa para o porte\_empresa 4



```
DELETE

FROM porte_empresa

WHERE porte_empresa = 4;
```

Exercício 01.05 - Visualizando o Histórico de Atualizações da tabela

```
DESCRIBE HISTORY porte_empresa;
```

Exercício 01.06 - Visalizando o conteúdo da tabela na versão anterior (Time Travel)

```
SELECT * FROM porte_empresa VERSION AS OF 5;
```

Exercício 01.07 - Restaurando o conteúdo da tabela na versão anterior

```
RESTORE TABLE porte_empresa TO VERSION AS OF 5;
```

Exercício 01.08 - Visualizando as propriedades da Tabela

```
DESCRIBE DETAIL porte_empresa;
```

Exercício 01.09 - Visualizando as informações detalhadas da Tabela

```
DESCRIBE TABLE EXTENDED porte_empresa;
```

# Hands- On 02 - Constraints

Documentação: Constraints on Databricks | Databricks on AWS

Exercício 02.01 - Campos NOT NULL

Para adicionarmos uma coluna como chave primária precisamos garantir que uma coluna seja NOT NULL, entã para isso, vamos adicionar essa informação nas colunas que iremos transformar em chaves primárias.

```
ALTER TABLE bronze_cnae
CHANGE COLUMN cod_cnae cod_cnae BIGINT NOT NULL;

ALTER TABLE bronze_empresas
CHANGE COLUMN cnpj_basico cnpj_basico BIGINT NOT NULL;

ALTER TABLE bronze_municipios
CHANGE COLUMN cod_municipio cod_municipio BIGINT NOT NULL;

ALTER TABLE bronze_naturezas
CHANGE COLUMN codigo codigo BIGINT NOT NULL;

ALTER TABLE porte_empresa
CHANGE COLUMN porte_empresa porte_empresa LONG NOT NULL;
```

Teste uma inserção de um dado nulo em uma coluna not null.

```
INSERT INTO porte_empresa ( desc_porte_empresa)
```

```
VALUES ( "TESTE");
```

Note que o seguinte erro irá aparecer, isso porque adicionando NOT NULL na coluna, impedimos que seja inserido registros que não contenham o id do porte\_empresa (coluna porte\_empresa):

```
| S3 | INSERT INTO porte_empresa ( desc_porte_empresa) |
| VALUES ( "TESTE"); |
| S5 | VALUES ( "TESTE"); |
| VAL
```

Agora faça o seguinte teste, insira um registro somente com o ID, e verifique o que vai acontecer:

```
INSERT INTO porte_empresa ( porte_empresa)
VALUES ( 6);
```

O que ocorreu? E porque?



Agora, apague o registro:

```
DELETE FROM porte_empresa
WHERE porte_empresa = 6;
```

Exercício 02.02 - Check Constraint - Garantir a Qualidade dos Dados

Agora criaremos uma constraint de checagem (**CHECK)**, ou seja, iremos bloquear os registros caso não atendam os requisitos de checagem.

Para isso, primeiro vamos verificar se a tabela atual atende os requisitos que iremos criar, vamos verificar se há um padrão na quantidade de caracteres do campo **cnpj\_basico** na tabela **bronze\_empresas**. (cnpj básico não tem o 0001 e os dois últimos dígitos)

```
select distinct (len(cnpj_basico)) from bronze_empresas;
```

```
select distinct (len(cnpj_basico)) from bronze_empresas;
65
```

Raw results > +

	1 <sup>2</sup> <sub>3</sub> len(cnpj_basico)	
1		8

Agora vamos adicionar a constraint:

```
ALTER TABLE bronze_empresas ADD CONSTRAINT cnpjQntCaracteres CHECK (len(cnpj_basico) = 8);
```

E agora, vamos tentar inserir um valor não válido para essa constraint:

```
INSERT INTO bronze_empresas (cnpj_basico)
VALUES (1243);
```

```
INSERT INTO bronze_empresas ( cnpj_basico)

VALUES (1243);

(DELTA_VIOLATE_CONSTRAINT_WITH_VALUES] CHECK constraint cnpjqntcaracteres (len(cnpj_basico) = 8) violated by row with values:
- cnpj_basico : 1243

Raw results > +
```

Faça um teste sem violar a constraint para verificar se funciona:

```
INSERT INTO bronze_empresas ( cnpj_basico)
VALUES (12345678);
```

```
Raw results > +

1<sup>2</sup><sub>3</sub> num_affected_rows 1<sup>2</sup><sub>3</sub> num_inserted_rows

1 1 1 1
```

Agora apague esse registro:

```
DELETE FROM bronze_empresas
WHERE cnpj_basico = 12345678;
```

Exercício 02.03 - PK Constraint - Chave Primaria

Adicione as seguintes chaves primarias:

```
ALTER TABLE bronze_cnae
```

```
ADD CONSTRAINT code_cnae_pk PRIMARY KEY (cod_cnae);

ALTER TABLE bronze_empresas

ADD CONSTRAINT cnpj_basico_pk PRIMARY KEY (cnpj_basico);

ALTER TABLE bronze_municipios

ADD CONSTRAINT cod_municipio_pk PRIMARY KEY (cod_municipio);

ALTER TABLE bronze_naturezas

ADD CONSTRAINT codigo_pk PRIMARY KEY (codigo);

ALTER TABLE porte_empresa

ADD CONSTRAINT porte_empresa_pk PRIMARY KEY (porte_empresa);
```

Verifique se elas foram adicionadas no Catalog Explorer.

```
db_workshop_dbsql_ana_sanch...

→ III bronze_cnae

      123 cod_cnae PK
      A<sup>B</sup>c descricao
 > m bronze_dolar

→ Im bronze_empresas

      1<sup>2</sup><sub>3</sub> cnpj_basico PK
      ABc razao_social
      123 natureza_juridica
      123 qualificacao_responsavel
      ABc capital_social
      1<sup>2</sup><sub>3</sub> porte_empresa
      ABc ente_federativo_responsavel
 > m bronze_estabelecimentos
 123 cod_municipio PK
      ABc nome_municipio

→ III bronze_naturezas

      123 codigo PK
      ABc descricao

▼ III porte_empresa
```

# Exercício 02.01 - FK Constraint - Chave estrangeira

Adicione as seguintes chaves estrangeiras:

123 porte\_empresa PK

ABc desc\_porte\_empresa

```
ALTER TABLE bronze_estabelecimentos
ADD CONSTRAINT fk_cnpj_basico
```

```
FOREIGN KEY (cnpj_basico)
REFERENCES bronze_empresas(cnpj_basico);

ALTER TABLE bronze_estabelecimentos
ADD CONSTRAINT fk_cnae_principal
FOREIGN KEY (cnae_principal)
REFERENCES bronze_cnae(cod_cnae);

ALTER TABLE bronze_empresas
ADD CONSTRAINT fk_porte_empresa
FOREIGN KEY (porte_empresa)
REFERENCES porte_empresa(porte_empresa);

ALTER TABLE bronze_empresas
ADD CONSTRAINT fk_natureza_juridica
FOREIGN KEY (natureza_juridica)
REFERENCES bronze_naturezas(codigo);
```

Verifique se elas foram adicionadas no Catalog Explorer.

- db\_workshop\_dbsql\_ana\_sanch...
  - > m bronze\_cnae
  - > m bronze\_dolar
  - bronze\_empresas
    - 123 cnpj\_basico PK
    - ABc razao\_social
    - 123 natureza\_juridica FK
    - 123 qualificacao\_responsavel
    - ABc capital\_social
    - 123 porte\_empresa FK
    - ABc ente\_federativo\_responsavel
  - - 1<sup>2</sup><sub>3</sub> cnpj\_basico FK
    - 123 cnpj\_ordem
    - 123 cnpj\_dv
    - 123 matriz\_filial
    - ABc nome\_fantasia
    - 123 codigo\_situacao\_cadastral
    - 123 data\_situacao\_cadastral
    - 123 motivo\_situacao\_cadastral
    - .00 nome\_cidade\_exterior
    - .00 codigo\_pais
    - <sup>12</sup><sub>3</sub> data\_inicio\_atividade
    - 123 cnae\_principal FK

Acesse a tabela através do Catalog Explorer e clique em View relationships.

Catalog Explorer > workshops\_databricks > db\_workshop\_dbsql\_ana\_sanchez\_6oxc > bronze\_estabelecimentos □ ☆ Overview Sample Data **Details Permissions** Insights History Lineage Quality Q Filter columns... € View relationships Al generate Column Type Comment Tags Column masking rule ① ① 0 cnpj... O¬ FK bigint 0 ① ① cnpj\_ordem bigint ① cnpj\_dv bigint  $\oplus$ P  $\oplus$ ① 0 matriz\_filial bigint 0 nome\_fantasia string  $\oplus$  $\oplus$ ① ① 0 codigo\_situa... bigint ① ① 0 data\_situaca... bigint 0 ① ① motivo\_situa... bigint ① ① 0 nome\_cidad... double

E vizualize o diagrame de entidade relacional gerado:



# Hands-On Lab 03 - Criação da camada Silver

#### Objetivo do exercício

O objetivo deste exercício é demonstrar como criar uma tabela na camada 'Silver' a partir das outras tabelas já ingeridas no laboratório anterior.

Exercício 03.01 - Criação da tabela Silver

```
CREATE OR REPLACE TABLE silver empresas
AS SELECT
 est.cnpj_basico AS cnpj_basico,
 matriz filial AS nome matriz,
 nome_fantasia AS nome_fantasia_empresa,
 razao social AS nome razao social,
 codigo_situacao_cadastral AS cod_situacao_cadastral,
 data_situacao_cadastral AS data_situacao_cadastral,
 motivo_situacao_cadastral AS motivo_situacao_cadastral,
 data_inicio_atividade AS data_inicio_atividade,
 cnae_principal AS cnae_principal,
 cnae.descricao AS cnae_descricao,
 tipo logradouro AS endereco_tipo_logradouro,
 logradouro AS endereco_nome_logradouro,
 numero AS endereco numero logradouro,
 bairro AS endereco_bairro_logradouro,
 cep AS endereco numero cep,
 uf AS endereco_unidade_federativa,
 codigo municipio siafi AS codigo municipio siafi,
 natureza juridica AS cod natureza juridica,
 nat.descricao AS desc_natureza_juridica,
 qualificacao responsavel AS qualificacao responsavel,
 capital_social AS val_capital_social,
 emp.porte empresa AS cod porte empresa,
 porte.desc_porte_empresa AS desc_porte_empresa,
 ente_federativo_responsavel AS ente_federativo_responsavel
from bronze_estabelecimentos est
join bronze_empresas emp
on est.cnpj_basico = emp.cnpj_basico
left join bronze_cnae cnae
on est.cnae_principal = cnae.cod_cnae
left join porte_empresa porte
```

```
on emp.porte_empresa = porte.porte_empresa
left join bronze_naturezas nat
on emp.natureza_juridica = nat.codigo;
```

Verifique no Catalog Explorer se a tabela realmente foi criada.

Exercício 03.02 - Adicionando comentários

```
COMMENT ON TABLE silver empresas IS 'Tabela contendo dados das empresas';
ALTER TABLE silver empresas ALTER COLUMN cnpj basico COMMENT 'número do CNPJ raiz
de 8 posições';
ALTER TABLE silver empresas ALTER COLUMN nome matriz COMMENT 'Nome da Matriz';
ALTER TABLE silver empresas ALTER COLUMN nome fantasia empresa COMMENT 'Nome
fantasia da empresa';
ALTER TABLE silver empresas ALTER COLUMN nome razao social COMMENT 'Nome da Razão
Social';
ALTER TABLE silver empresas ALTER COLUMN cod situacao cadastral COMMENT 'Código da
Situação Cadastral';
ALTER TABLE silver empresas ALTER COLUMN data situacao cadastral COMMENT 'Data da
Situação Cadastral';
ALTER TABLE silver_empresas ALTER COLUMN motivo_situacao_cadastral COMMENT 'Motivo
da Situação Cadastral';
ALTER TABLE silver_empresas ALTER COLUMN data_inicio_atividade COMMENT 'Data de
início da atividade';
ALTER TABLE silver_empresas ALTER COLUMN cnae_principal COMMENT 'CNAE Código da
Natureza Economica':
ALTER TABLE silver empresas ALTER COLUMN cnae descricao COMMENT 'Descrição do
CNAE':
ALTER TABLE silver empresas ALTER COLUMN endereco tipo logradouro COMMENT 'Endereço
- Tipo de Logradouro';
ALTER TABLE silver empresas ALTER COLUMN endereco nome logradouro COMMENT 'Endereço
- Nome do Logradouro';
ALTER TABLE silver empresas ALTER COLUMN endereco numero logradouro COMMENT
'Endereço - Número do Logradouro';
ALTER TABLE silver empresas ALTER COLUMN endereco bairro logradouro COMMENT
'Endereço - Bairro do Logradouro';
ALTER TABLE silver empresas ALTER COLUMN endereco numero cep COMMENT 'Endereco -
número do CEP';
ALTER TABLE silver empresas ALTER COLUMN endereco unidade federativa COMMENT
'Endereço - Unidade Federativa';
ALTER TABLE silver_empresas ALTER COLUMN codigo_municipio_siafi COMMENT 'Código do
Município SIAFI';
ALTER TABLE silver_empresas ALTER COLUMN cod_natureza_juridica COMMENT 'Código da
Natureza Jurídica';
```

```
ALTER TABLE silver_empresas ALTER COLUMN desc_natureza_juridica COMMENT 'Descrição da Natureza Jurídica';
ALTER TABLE silver_empresas ALTER COLUMN qualificacao_responsavel COMMENT 'Qualificação do Responsável';
ALTER TABLE silver_empresas ALTER COLUMN val_capital_social COMMENT 'Valor do Capital Social';
ALTER TABLE silver_empresas ALTER COLUMN cod_porte_empresa COMMENT 'Código do Porte da Empresa';
ALTER TABLE silver_empresas ALTER COLUMN desc_porte_empresa COMMENT 'Descrição do Porte da Empresa';
ALTER TABLE silver_empresas ALTER COLUMN ente_federativo_responsavel COMMENT 'Ente Federativo Responsável';
```

Verifique no **Catalog Explorer** se a descrição da tabela e os comentários das colunas realmente foram adicionados.

Exercício 03.03 - Carregando Dados JSON

Dentro das pastinhas do repositório que você importou no início desse doc, execute o arquivo lab00\_carga\_csv dentro do diretório lab03.03 - ingerir socios json.

Exercício 03.04 - Criando Streaming Tables

Crie a tabela de streaming que irá carregar as informações dos sócios da empresa:

```
CREATE OR REFRESH STREAMING TABLE bronze_socios(
    cnpj_basico BIGINT COMMENT 'Número do CNPJ raiz de 8 posições',
    identificador_socio BIGINT COMMENT 'Código de identificação do sócio',
    nome_socio STRING COMMENT 'Nome do sócio',
    cpf_cnpj_socio STRING COMMENT 'CPF ou CNPJ do sócio',
    qualificação_socio BIGINT COMMENT 'Código de qualificação do sócio',
    data_entrada_sociedade STRING COMMENT 'Data de entrada do sócio na sociedade',
    pais_socio BIGINT COMMENT 'Código do país do sócio, se estrangeiro',
    representante_legal STRING COMMENT 'Nome do representante legal',
    faixa_etaria BIGINT COMMENT 'Faixa etária do sócio',
    _rescued_data STRING
)

SCHEDULE CRON '0/1 0/1 * ? * * *'

AS SELECT * FROM STREAM

read_files('/Volumes/workshops_databricks/db_workshop_SEUNOME_OU_CHAVE/landing/soci
os/socios*', format => 'json')
```

Lembrem-se de alterar o caminho do volume para o volume informado pelo instrutor.

Dentro das pastinhas do repositório que você importou no início desse doc, execute o arquivo lab00\_carga\_csv dentro do diretório lab03.04 - ingerir novos socios json.

Verifique se a tabela de streaming recebeu os novos dados.

Exercício 03.06 - Criando Materialized Views

Nesta etapa vamos criar a view materializada que conterá algumas agregações que utilizaremos futuramente nos exercícios posteriores. Copiem o código abaixo e executem-no no SQL Editor.

```
CREATE OR REPLACE MATERIALIZED VIEW gold_analise_empresarial
SCHEDULE CRON '0 0 0 * * ? *'
AS
SELECT
c.descricao AS setor_economico,
e.endereco unidade federativa AS uf,
pe.desc porte empresa AS porte empresa,
COUNT(DISTINCT e.cnpj basico) AS qtd empresas,
AVG(s.num socios) AS media socios por empresa,
SUM(CASE WHEN s.pais socio != 1058 THEN 1 ELSE 0 END) / COUNT(DISTINCT
e.cnpj basico) * 100 AS percentual socios estrangeiros,
AVG(CAST(e.val_capital_social AS DOUBLE)) AS capital_social_medio,
COLLECT_SET(n.descricao) AS principais_naturezas_juridicas
FROM
silver empresas e
JOIN
bronze cnae c ON e.cnae principal = c.cod cnae
JOIN
bronze porte empresa pe ON e.desc porte empresa = pe.desc porte empresa
LEFT JOIN
(SELECT cnpj_basico, COUNT(*) AS num_socios, MAX(pais_socio) AS pais_socio
 FROM bronze socios
 GROUP BY cnpj_basico) s ON e.cnpj_basico = s.cnpj_basico
JOIN
bronze_naturezas n ON e.cod_natureza_juridica = n.codigo
GROUP BY ALL
```

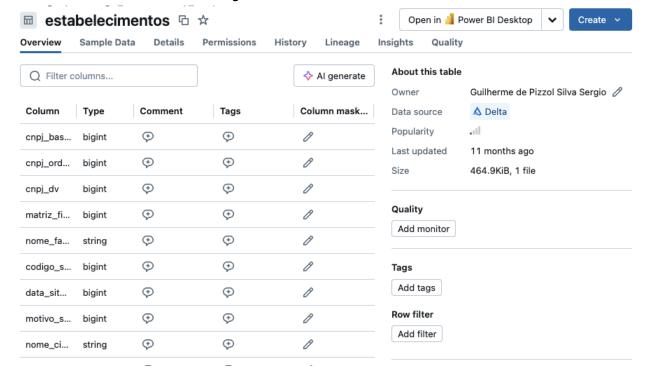
# Hands-On Lab 04 - Filtrando e Mascarando linhas e colunas

Exercício 04.01 - Filtrando os registros

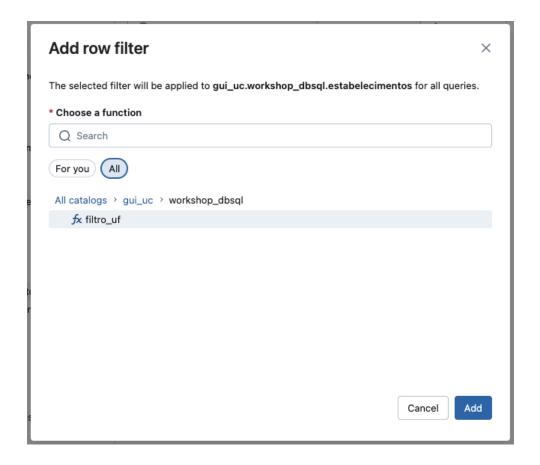
Para criarmos um filtro de linhas dentro do Databricks, é necessário criarmos uma função que irá dizer qual é a lógica por trás da filtragem das linhas, para tal, copie o código abaixo e execute-o em uma nova aba do SQL Editor:

```
CREATE OR REPLACE FUNCTION filtro_uf(uf STRING)
RETURN
is_account_group_member('admin') or
    exists (
        SELECT 1
        FROM silver_empresas
        WHERE is_account_group_member('grupo_1') AND endereco_unidade_federativa = uf
    )
```

Após a criação da função, existem duas formas de adicionar a função à uma tabela. Através da interface, como mostrado nas imagens abaixo:



Navegue até a tabela "bronze\_estabelecimentos" e clique no botão 'Add filter' no canto inferior direito.



Selecione a função que você criou no passo anterior.

Add row filter ×

The selected filter will be applied to gui\_uc.workshop\_dbsql.silver\_empresas for all queries.

#### \* Choose a function



#### **Function definition**

```
is_account_group_member('admin') or
    exists (
        SELECT 1
        FROM silver_empresas
        WHERE is_account_group_member('grupo_1') AND endereco_unidade_federativa
= uf
    )
```

uf

Select a column

Cancel

Add

Selecione a coluna que a função se baseia para filtrar os registros, neste caso, vamos utilizar a coluna 'uf'.

Outra maneira de adicionar o filtro é através do SQL, caso prefira, tente adicionar o filtro através do código:

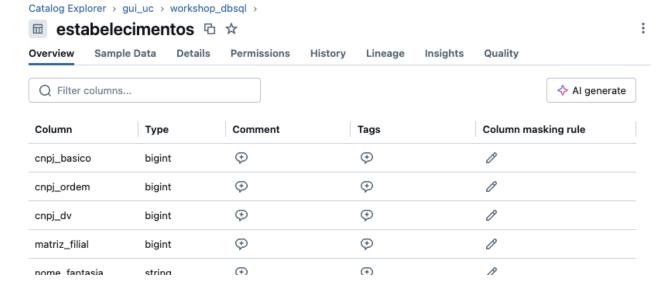
```
ALTER TABLE bronze_estabelecimentos SET ROW FILTER filtro_uf ON (uf);
```

#### Exercício 04.02 - Mascarando os registros

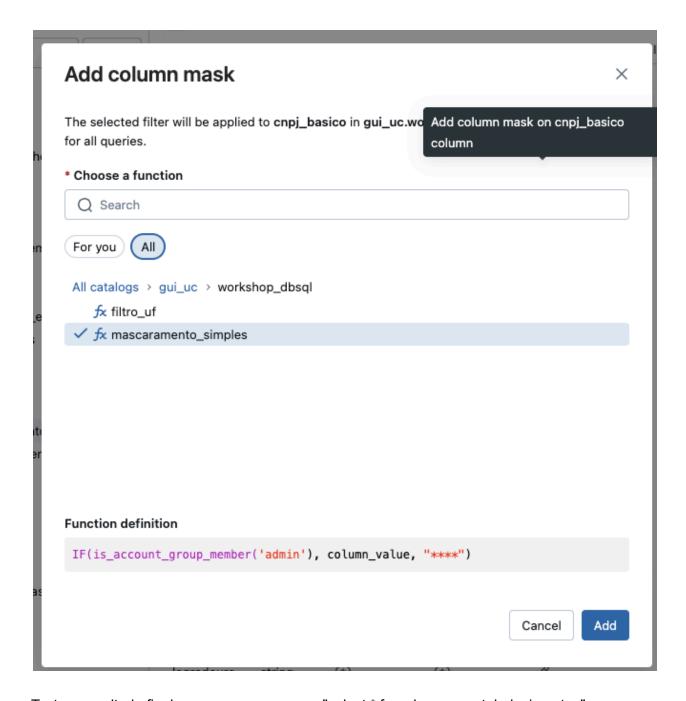
De forma bem semelhante, o mascaramento necessita de uma função por trás para definir quais colunas serão mascaradas. Crie a função de mascaramento com base no código abaixo:

```
CREATE OR REPLACE FUNCTION mascaramento_simples(column_value STRING)
RETURN
IF(is_account_group_member('admin'), column_value, "****");
```

Também é possível adicionar a função de mascaramento, através da interface:



A última coluna na aba de overview de cada tabela contém a informação das funções de mascaramento que são aplicadas à coluna. Vamos adicionar uma nova regra na coluna cnpj basico, para tal, clique no 'lápis' e selecione a função criada no passo anterior:



Teste o resultado final com uma query como "select \* from bronze\_estabelecimentos".

# Hands-On Lab 05 - Criação de Alertas

# Objetivo do exercício

O objetivo desse laboratório é explorar as funcionalidade de criação de um ALERTA

Exercício 05.01 - Criação da Query

Abra uma nova aba de Query:

```
Phands on dbsql  

Tre... Serverless S  

Create new query

Open existing query

Código da Natureza Economica ,

Crição do CNAE';

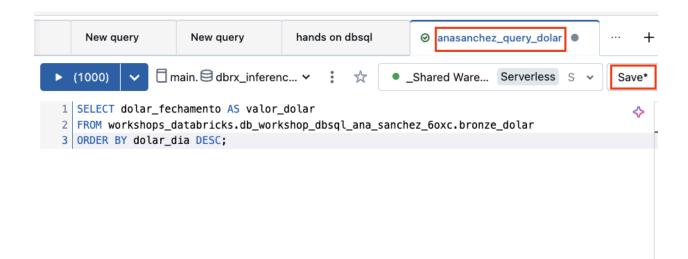
MMENT 'Endereço − Tipo de Logradouro';

MMENT 'Endereço − Nome do Logradouro';

COMMENT 'Endereço − Número do Logradouro';
```

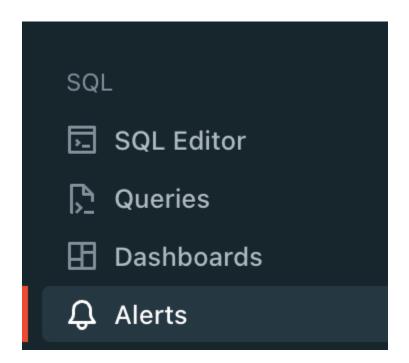
Adicione a seguinte query e salve ela como : SEUNOME query dolar

```
SELECT dolar_fechamento AS valor_dolar
FROM
workshops_databricks.db_workshop_dbsql_SEU_NOME_ou_SUA_CHAVE.bronze_dolar
ORDER BY dolar_dia DESC;
```



#### Exercício 05.02 - Criando o Alerta

Vamos utilizar a opção do menu "ALERT".



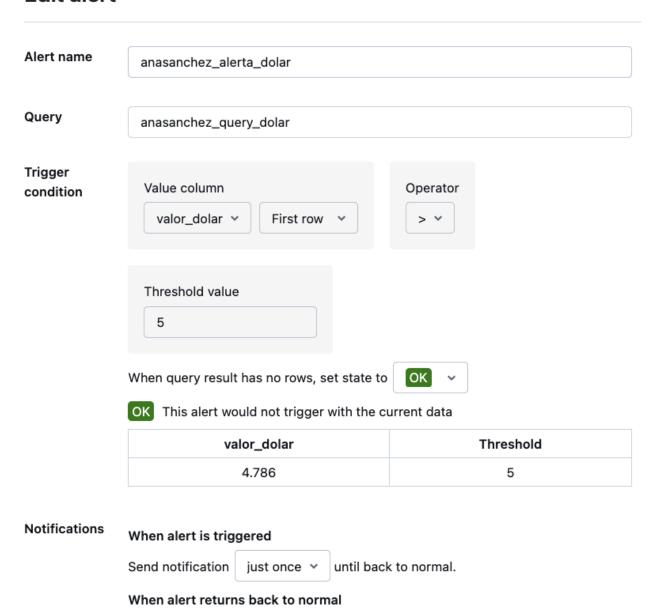
Clique no botão CREATE e configure o ALERTA conforme abaixo, com o seguinte nome: SEUNOME\_alerta\_dolar e após preencher todos os campos clique em Preview alert

Send notification

Use default template >

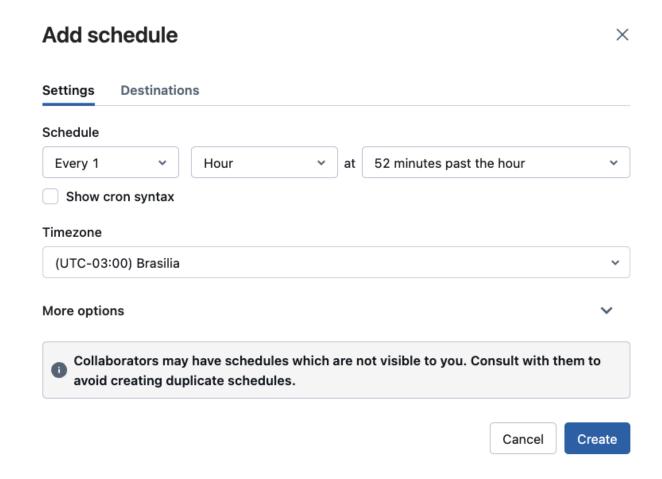
**Template** 

### **Edit alert**

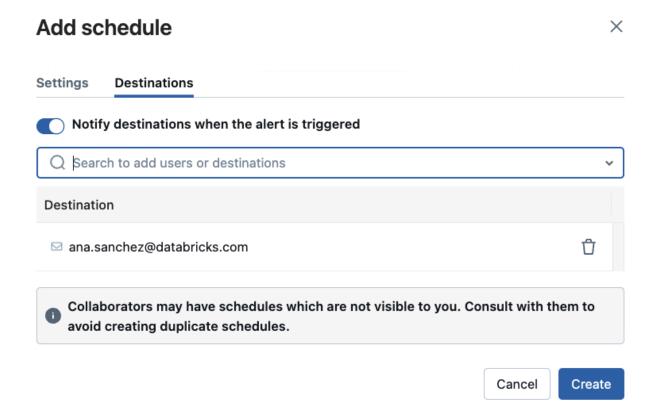


Adicionar um e-mail para quando o alerta dispalar, para isso clique em Add schedule.

Deixe os valores default pois depois da execução manual iremos desligar.

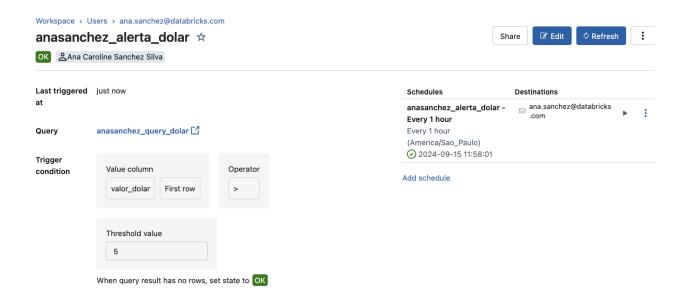


Vá para a aba **Destinations**, adicione o seu email e clique em **Create**:

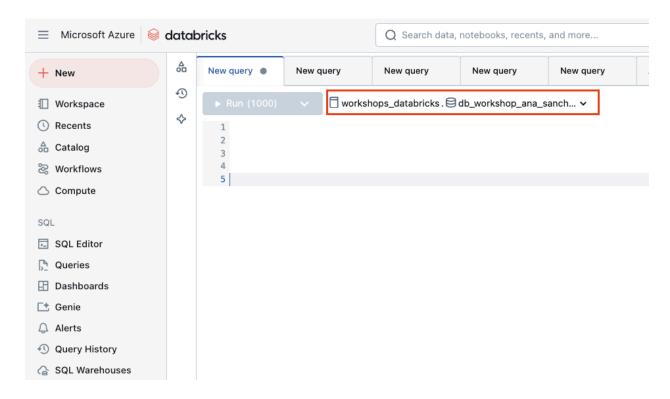


Clique em .

Qualdo terminar você vai ver o status do alerta abaixo do nome dele:



Agora vamos inserir um valor de dolar para um novo dia, com um valor maior que 5 para ver se o alerta vai disparar quando o executarmos, para isso abra uma nova aba o **SQL Editor** e selecione seu catálogo (**workshops\_databricks**) e seu schema (**db\_workshop\_SEUNOME\_OU\_SUACHAVE**) no topo do Query Editor:



Crie uma query para adicionar um dia no maior dia da tabela **bronze\_dolar** no campo **dolar\_dia** 

```
SELECT
   MAX(dolar_dia) ultimo_dia,
   to_date(DATEADD(day, 1, MAX(dolar_dia))) proximo_dia
FROM bronze_dolar;
```

```
6 SELECT
7 MAX(dolar_dia) ultimo_dia,
8 to_date(DATEADD(day, 1, MAX(dolar_dia))) proximo_dia
9 FROM bronze_dolar;
```

#### Raw results > +

	<sup>B</sup> <sub>C</sub> ultimo_dia	🛱 proximo_dia
1	2023-06-30	2023-07-01

Agora insira os seguinte valores na tabela bronze:

- Dolar\_dia: calcule o ultimo dia da tabela e some com 1
- Dolar\_mes: pegue o mês do campo calculado anteriormente
- Dolar\_ano: pegue o ano do campo calculado anteriormente
- Dolar fechamento: 5.12
- Dolar\_abertura: pegue o dolar de fechamento do dia anterior
- Dolar\_max: 5.22
- Dolar\_min: 4.72

```
INSERT INTO bronze_dolar
  (dolar_dia, dolar_mes, dolar_ano, dolar_fechamento, dolar_abertura, dolar_max, dolar_min)
SELECT
  to_date(DATEADD(day, 1, MAX(dolar_dia))),
  month(DATEADD(day, 1, MAX(dolar_dia))),
  year(DATEADD(day, 1, MAX(dolar_dia))),
  5.12,
  (SELECT dolar_fechamento FROM bronze_dolar ORDER BY dolar_dia DESC LIMIT 1),
  5.22,
  4.72
```

#### Verifique o resultado:

FROM bronze\_dolar;

```
SELECT *
FROM workshops_databricks.db_workshop_dbsql_ana_sanchez_6oxc.bronze_dolar
```

#### ORDER BY dolar\_dia DESC limit 1;



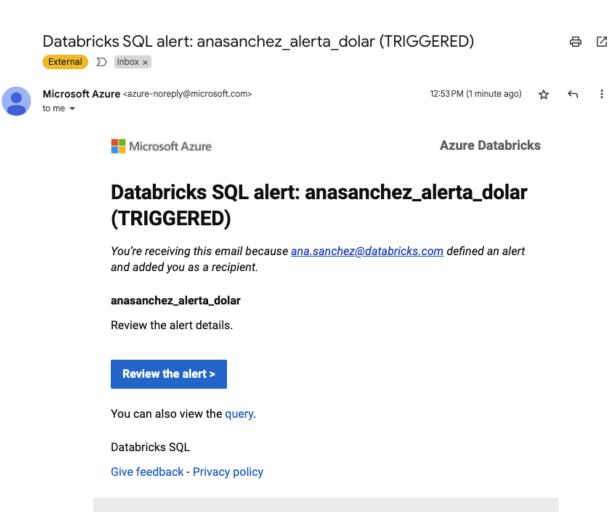
Agora volte para o seu alerta e o execute :

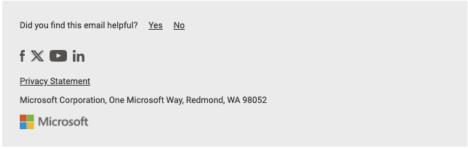
Workspace > Users > ana.sanchez@databricks.com

anasanchez\_alerta\_dolar 🖈

TRIGGERED Ana Caroline Sanchez Silva

Verifique seu email:





← Reply ← Forward

## Hands-On Lab 06 - DBSQL API

## Objetivo do exercício

O objetivo desse laboratório é realizar queries SQL no Databricks SQL através de sua API REST.

#### Exercício 06.01 - Testes com o DBSQL API

- 1. Acesse ferramentas como Postman ou Insomnia.
- 2. Importe a collection de consultas de APIs em sua ferramenta:

  dbsql-api/postman\_files/Databricks SQL Execution API.postman\_collection.json at main

  anasanchezss9/dbsql-api (github.com)
- 3. Importe o arquivo de environment em sua ferramenta: <a href="mailto:dbsql-api/postman\_files/Databricks">dbsql-api/postman\_files/Databricks</a> Environment.postman\_environment.json at main · anasanchezss9/dbsql-api (github.com)
- 4. Siga os passos abaixo do link a seguir para configurar o arquivo de environment e depois realizar os testar com a API: <a href="mailto:dbsql-api/README.md">dbsql-api/README.md</a> at main : <a href="mailto:anasanchezss9/dbsql-api">anasanchezss9/dbsql-api</a> (github.com)

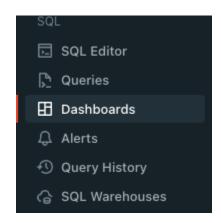
## Hands-On Lab 07 - Criando Dashboards

## Objetivo do exercício

O objetivo desse laboratório é explorar e entender como funciona a nova ferramenta Lakeview Dashboard.

Exercício 07.01 - Criando o dashboard

Acesse o menu "Dashboards" na aba lateral esquerda.



Selecione a aba "Dashboards"

### **Dashboards**

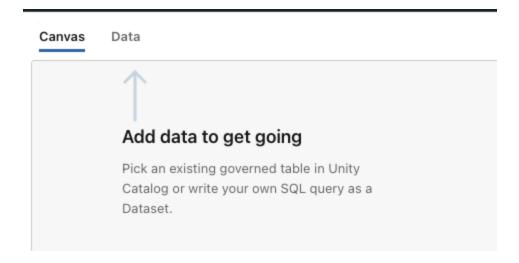


Por último clique no botão "Create Lakeview dashboard" localizado no canto superior direito

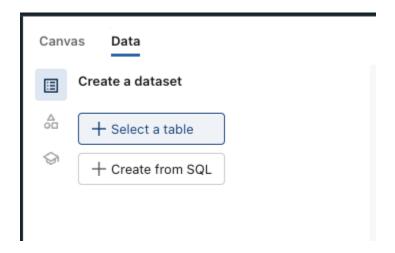


Exercício 07.02 - Criando o dataset

Clique na aba "Data" conforme exibido na figura



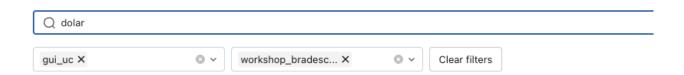
Em seguida clique no botão "+ Select a table"



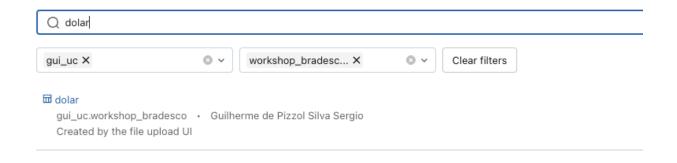
Filtre pela tabela "dolar" adicionando o filtro de catálogo e schema caso necessário (pressione Enter para exibir o resultado da busca)

#### Select table

(i) You may still have content in the Hive Metastore that cannot be searched. Please migrate it to Unity Catalog if you want to search that content.



Selecione a tabela "dolar" que criamos no laboratório anterior, conforme mostrado na imagem abaixo:

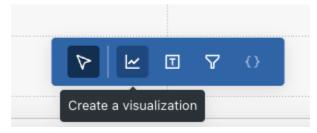


## Exercício 07.03 - Criando visualizações no Canvas

Volte para a aba "Canvas"



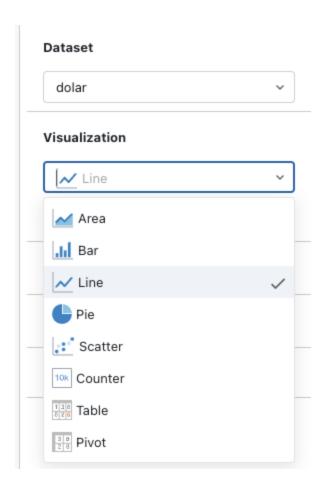
Clique no botão "Create a visualization" conforme mostrado na imagem abaixo:



Em seguida selecione a área que deseja criar a visualização

anvas Data	New Dashboard 2023-10-13 19:02:48 (Saved)				i) (

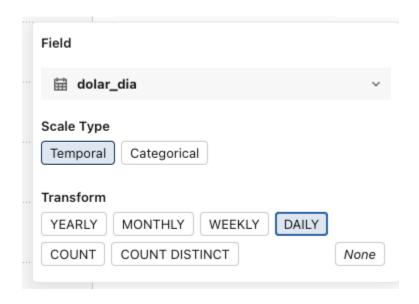
Na aba lateral direita, verifique se o dataset "dolar" já está selecionado, e se assim estiver, selecione a opção "Line" nos tipos de visualizações



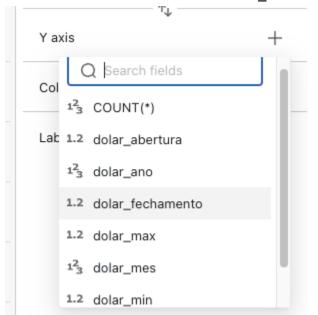
Para o eixo X selecione a coluna "dolar\_dia" conforme a imagem abaixo



Clique na agregação MONTHLY(dolar\_dia) e mude para DAILY

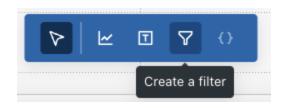


Para o eixo Y selecione a coluna "dolar\_fechamento"

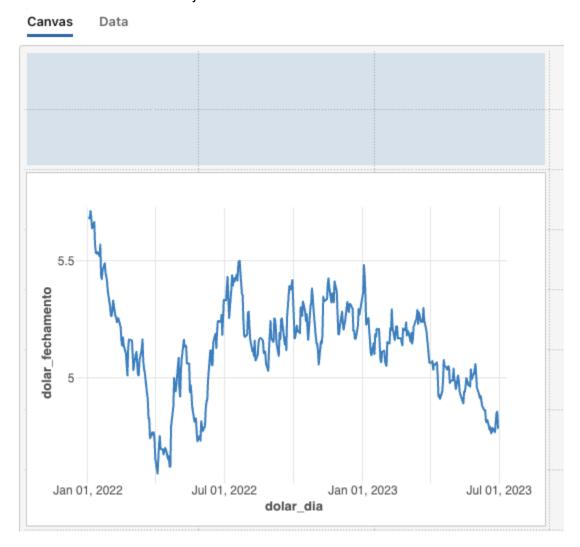


Exercício 07.04 - Criando Filtros

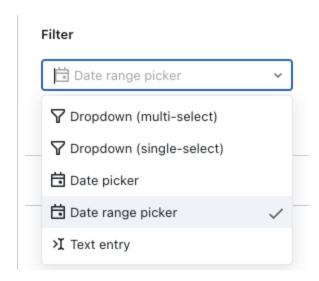
No menu central clique no botão "Create a filter"



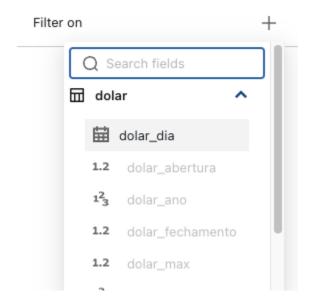
#### Posicione o filtro como desejar



No menu lateral direito selecione como tipo de filtro "Date range picker"

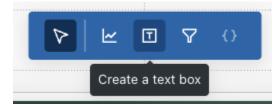


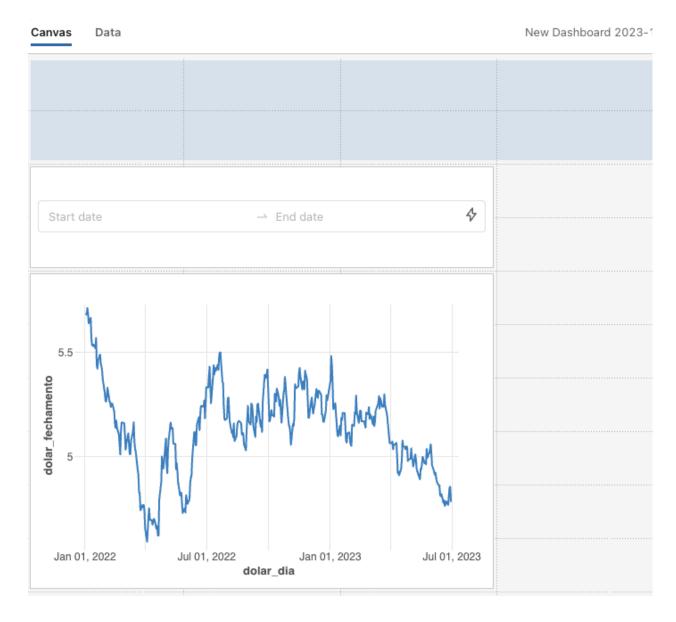
Clique no "+" da opção Filter on e selecione a coluna "dolar\_dia"



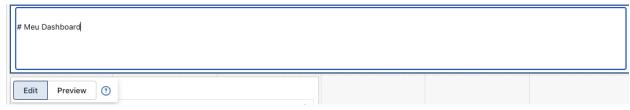
Exercício 07.05 - Adicionando Cabeçalhos

No menu central clique na opção "Create a text box"

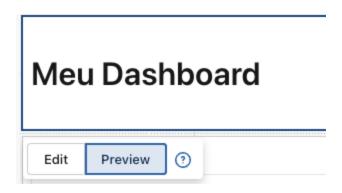




Na caixa de texto, escreva um título para seu dashboard utilizando a notação de markdown como base (lembre-se que é possível de adicionar links e imagens a este campo de texto para deixar seus dashboards ainda mais bacanas!)



Clique em "Preview" para pré-visualizar o resultado



Quando estiver satisfeito, nomeie seu dashboard e compartilhe com seus colegas.

# Hands-On Lab 08 - Conversando com os dados com o Al/Bl Genie

## Objetivo do exercício

O objetivo desse exercício é demonstrar as capacidades do Al/Bl Genie para obter resultados com queries SQL utilizando linguagem natural através da sua interface.

Exercício 08.01 - Criando e configurando seu espaço

O primeiro passo para criar um novo espaço é acessar o menu lateral e clicar no botão 'Genie', como na imagem abaixo:

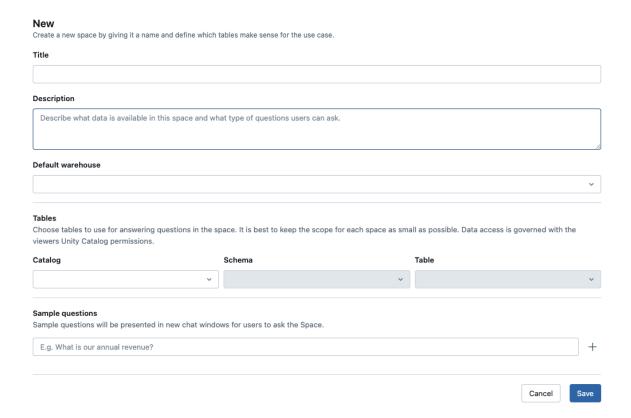


Feito isso, clique no botão 'New' no canto superior direito:

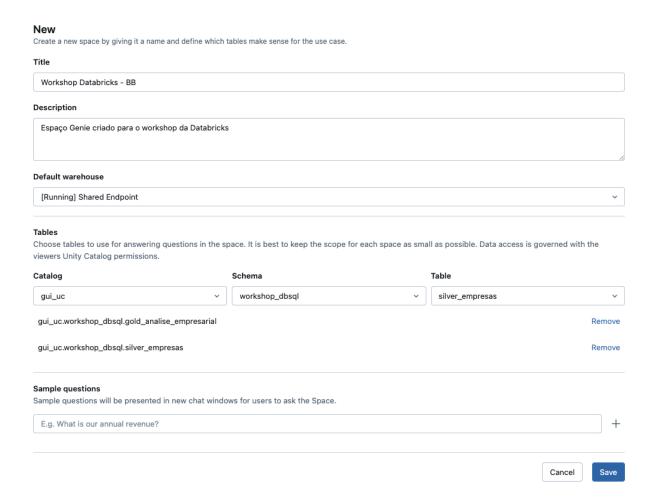


+ New

E comece a configurar seu espaço:



Nessa etapa vamos editar o nome do espaço, sua descrição, selecionar o Warehouse e adicionar as tabelas iniciais que farão parte da análise:

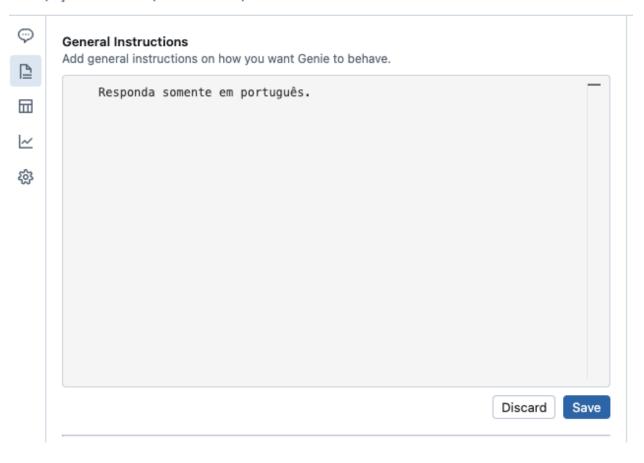


(Iremos utilizar apenas as tabelas gold\_analise\_empresarial e silver\_empresas)

Por último, vamos adicionar uma instrução na aba de 'Instructions' como na imagem abaixo:

# Workshop Databricks - BB // Feedback [2]

Espaço Genie criado para o workshop da Databricks



Fique à vontade para explorar o espaço criado fazendo perguntas na aba 'Chat':

