**🚀 VISÃO GERAL**

Você vai:

1. Instalar Python, Spark e DBT
2. Criar um projeto DBT com estrutura personalizada
3. Usar **PySpark como engine**
4. Trabalhar no projeto com **VS Code**
5. Testar e manipular o projeto localmente

**🔧 1. INSTALAÇÕES INICIAIS**

**a) Instale o Python**

* Baixe do site: <https://www.python.org/downloads/windows/>
* Marque: ✅ “Add Python to PATH”
* Verifique:

powershell

CopiarEditar

python --version

pip --version

**b) Instale o Java JDK 8+**

Spark exige Java. Recomendo Temurin JDK 11.

Verifique:

powershell

CopiarEditar

java -version

**c) Instale o Apache Spark**

1. Acesse: <https://spark.apache.org/downloads.html>
2. Escolha:
   * Spark 3.5.1 (ou mais recente)
   * Pré-built for Hadoop 3
3. Extraia para C:\spark

Configure variáveis de ambiente:

powershell

CopiarEditar

$env:SPARK\_HOME="C:\spark"

$env:PATH="$env:SPARK\_HOME\bin;$env:PATH"

**d) Instale o DBT + Spark Adapter**

Crie seu projeto em D:\Projetos\DBT\_Spark:

powershell

CopiarEditar

cd D:\

mkdir Projetos\DBT\_Spark

cd Projetos\DBT\_Spark

python -m venv .venv

.\.venv\Scripts\Activate.ps1

pip install dbt-spark

**🧠 2. CRIAR ESTRUTURA DO PROJETO DBT**

powershell

CopiarEditar

dbt init dbt\_spark\_project

cd dbt\_spark\_project

Delete o conteúdo original de models/ e crie essa estrutura:

pgsql

CopiarEditar

dbt\_spark\_project/

│

├── analysis/

├── data/

│ ├── seed\_clientes.csv

│ └── seed\_vendas.csv

├── logs/

├── macros/

├── models/

│ ├── marts/

│ │ ├── dim\_clientes.sql

│ │ └── fct\_vendas.sql

│ └── staging/

│ ├── stg\_clientes.sql

│ └── stg\_vendas.sql

├── seeds/

├── snapshots/

├── tests/

├── dbt\_project.yml

**🧩 3. CONFIGURAR profiles.yml**

Abra:

powershell

CopiarEditar

notepad $env:USERPROFILE\.dbt\profiles.yml

Adicione:

yaml

CopiarEditar

dbt\_spark\_project:

target: dev

outputs:

dev:

type: spark

method: session

schema: default

database: spark

threads: 1

host: localhost

port: 0

**⚙️ 4. CONFIGURAR dbt\_project.yml**

Atualize o arquivo dbt\_project.yml:

yaml

CopiarEditar

name: 'dbt\_spark\_project'

version: '1.0'

config-version: 2

profile: 'dbt\_spark\_project'

model-paths: ["models"]

analysis-paths: ["analysis"]

test-paths: ["tests"]

seed-paths: ["data"]

macro-paths: ["macros"]

snapshot-paths: ["snapshots"]

target-path: "target"

clean-targets: ["target", "dbt\_modules"]

models:

dbt\_spark\_project:

marts:

+materialized: table

staging:

+materialized: view

**🧪 5. TESTAR CONEXÃO E PRIMEIROS COMANDOS**

Ative o ambiente virtual:

powershell

CopiarEditar

cd D:\Projetos\DBT\_Spark\dbt\_spark\_project

.\.venv\Scripts\Activate.ps1

dbt debug

Se tudo estiver ok, execute:

powershell

CopiarEditar

dbt seed # Carrega os CSVs como tabelas

dbt run # Executa os modelos SQL

dbt test # Roda os testes (se houver)

**📂 6. EXEMPLOS DE MODELOS E SEEDS**

**a) data/seed\_clientes.csv**

csv

CopiarEditar

id,nome,estado

1,Ana,SP

2,Beto,RJ

3,Carlos,MG

**b) data/seed\_vendas.csv**

csv

CopiarEditar

id,cliente\_id,valor

1,1,100

2,2,200

3,1,300

**c) models/staging/stg\_clientes.sql**

sql

CopiarEditar

select

id as cliente\_id,

nome,

estado

from {{ ref('seed\_clientes') }}

**d) models/staging/stg\_vendas.sql**

sql

CopiarEditar

select

id as venda\_id,

cliente\_id,

valor

from {{ ref('seed\_vendas') }}

**e) models/marts/dim\_clientes.sql**

sql

CopiarEditar

select distinct

cliente\_id,

nome,

estado

from {{ ref('stg\_clientes') }}

**f) models/marts/fct\_vendas.sql**

sql

CopiarEditar

select

v.venda\_id,

v.valor,

c.nome,

c.estado

from {{ ref('stg\_vendas') }} v

left join {{ ref('stg\_clientes') }} c

on v.cliente\_id = c.cliente\_id

**🖥️ 7. USAR O VS CODE**

1. Baixe e instale: <https://code.visualstudio.com/>
2. Abra a pasta D:\Projetos\DBT\_Spark\dbt\_spark\_project
3. Instale as extensões:
   * Python
   * DBT Power User
   * SQL
   * YAML
4. No terminal do VS Code, ative o ambiente:

powershell

CopiarEditar

.\.venv\Scripts\Activate.ps1

**✅ 8. BOAS PRÁTICAS**

* Sempre use ref() para dependências entre modelos
* Separe por camadas: staging, marts, etc
* Crie **testes** para not null, unique, relationships
* Documente com dbt docs generate e visualize com dbt docs serve
* Use Git para versionar

**🎯 9. PRÓXIMOS PASSOS**

* Adicionar testes em tests/
* Escrever macros reutilizáveis
* Gerar e abrir a documentação:

powershell

CopiarEditar

dbt docs generate

dbt docs serve

* Aprender sobre:
  + **Snapshots**
  + **Incremental models**
  + **Agendamentos com Airflow ou cron**

Se quiser, posso te mandar todos esses arquivos prontos como template .zip ou montar o conteúdo completo das SQLs e arquivos de teste/macros. Deseja isso?