Desenvolvimento e Avaliação de uma Arquitetura Distribuída
para o Cadastro Ambiental Rural

1. DEFINIR LOCAL E MELHOR FORMATO PARA SALVAR OS DADOS

Dados: temas_ambientais.csv (~2.02 GB)

- **Armazenamento**:
    - **Local**: Armazenar os dados no **Databricks File System (DBFS)** para facilitar o acesso e processamento no ambiente Databricks Community.
    - **Formato**: Converter para **Delta Lake Format**. Este formato oferece suporte a **ACID transactions**, **time travel**, e é otimizado para consultas em ambientes distribuídos.
        - **Justificativa**:
            - CSV tem limitações de desempenho para leitura e escrita distribuídas.
            - Delta Lake melhora a escalabilidade e permite gerenciamento transacional dos dados.
        - **Conversão**:
            0. Importar o CSV para o Databricks.
            1. Converter os dados para o formato Delta Lake.
            2. Salvar os dados em um caminho específico no DBFS.

2. PROPOSIÇÃO DA NOVA ARQUITETURA

Objetivo:

Criar uma arquitetura distribuída eficiente que permita:

- Consulta rápida.
- Gerenciamento transacional.
- Escalabilidade horizontal.

Proposta:

- **Armazenamento**:
    - **Delta Lake** para armazenamento transacional dos dados no DBFS.
- **Processamento**:
    - **Apache Spark** (integrado ao Databricks) para processamento distribuído, incluindo leitura, escrita e transformação.
- **Particionamento**:
    - Estratégia de particionamento baseada em **atributos geoespaciais**, com o estado.
- **Camadas da Arquitetura** (Medallion Architecture):
    - **Bronze**: Dados brutos em formato Delta.
    - **Silver**: Dados pré-processados, com transformações aplicadas.
    - **Gold**: Dados prontos para análise e consulta.

Componentes:

1. **Databricks Community**: Ambiente de desenvolvimento e processamento distribuído.
2. **DBFS**: Armazenamento persistente.
3. **Delta Lake**: Formato otimizado para dados transacionais.

3. IMPLEMENTAÇÃO DA ARQUITETURA

Etapas:

1. **Configuração Inicial**:
    - Configurar o workspace no Databricks Community.
    - Carregar o arquivo CSV para o DBFS.
2. **Conversão para Delta Lake**:
    - Criar uma tabela Delta na camada **Bronze**.
    - Aplicar transformações iniciais para gerar a camada **Silver**.
    - Gerar agregações e otimizações para a camada **Gold**.
3. **Validação da Configuração**:

- Testar leitura e escrita das tabelas para garantir a consistência e desempenho.

4. MODELAGEM DO NOVO BANCO DE DADOS

Esquema Proposto:

- **Tabelas**:
  - **Propriedades**: Detalhes de cada propriedade rural.
  - **Geografia**: Dados geoespaciais associados às propriedades.
  - **Atividades**: Informações sobre usos e restrições ambientais.
- **Relacionamentos**:
  - Identificar relacionamentos entre os atributos do CAR.

##

Avaliar o Tempo de Resposta das Consultas

Para avaliar o tempo de execução de cada consulta no Databricks:

A. Usar `time` no Notebook

```
import time

# Exemplo: Medir tempo de execução para Consulta 1
start_time = time.time()

query1 = spark.sql("""
    SELECT uf, SUM(area_do_imovel) AS total_area_hectares
    FROM gold_temas_ambientais
    WHERE uf IN ('MS', 'MT')
    GROUP BY uf
    ORDER BY total_area_hectares DESC
""")
query1.show()

end_time = time.time()
print(f"Tempo de execução: {end_time - start_time:.2f} segundos")
```

B. Monitorar no Databricks UI

1. **Exibir Detalhes de Jobs:**
    - Após executar uma consulta, clique em **"View"** ao lado do botão de execução no Databricks para acessar os detalhes do job.
    - Verifique o tempo total de execução e os estágios no DAG (Directed Acyclic Graph).