```
# ================================
# DuckDB vs Pandas Benchmark Demo
# ================================

# Install DuckDB in Colab
!pip install duckdb --quiet

import pandas as pd
import duckdb
import time
import os
```

```
Start coding or generate with AI.
```

```
# ============================
# Install required libraries
# ============================
!pip install duckdb pyarrow pandas --quiet
# duckdb      → Embedded OLAP database engine for super-fast analytics
# pyarrow     → Required for Parquet file handling in Pandas and DuckDB
# pandas      → Data analysis library

# ============================
# Import necessary libraries
# ============================
import pandas as pd       # For reading/writing CSV & Parquet, and Pandas transformations
import duckdb             # For querying Parquet/CSV directly with SQL
import time               # For measuring execution time

# ============================
# Define file paths (Google Colab environment)
# ============================
csv_path = "/content/sample_data/huge_sales.csv"             # Path to original CSV file (1M+ rows)

# ============================
# Step 1: Convert CSV → Parquet
# ============================
df_csv = pd.read_csv(csv_path)                   # Load CSV file into Pandas DataFrame (entire dataset in memory)
df_csv.to_parquet(parquet_path, compression='snappy')
# Convert DataFrame to Parquet format using Snappy compression (columnar format → faster read + smaller size)

# ============================
# Step 2: Benchmark Pandas (CSV Input)
# ============================
def benchmark_pandas_csv():
    start = time.time()                          # Record start time
    df = pd.read_csv(csv_path)                   # Load CSV file into DataFrame
    result = df.groupby('product').agg(          # Group data by 'product'
        total_sales=pd.NamedAgg(column='price', aggfunc='sum'),    # Sum of 'price' for each product
        avg_quantity=pd.NamedAgg(column='quantity', aggfunc='mean') # Average 'quantity' for each product
    )
    print(f"CSV (Pandas) time: {round(time.time() - start, 2)} sec")
    # Print total execution time (rounded to 2 decimals)
    return result                                # Return the aggregated DataFrame

# ============================
# Step 3: Benchmark Pandas (Parquet Input)
# ============================
def benchmark_pandas_parquet():
    start = time.time()                          # Record start time
    df = pd.read_parquet(parquet_path)           # Load Parquet file into DataFrame
    result = df.groupby('product').agg(          # Group data by 'product'
        total_sales=pd.NamedAgg(column='price', aggfunc='sum'),    # Sum of 'price'
        avg_quantity=pd.NamedAgg(column='quantity', aggfunc='mean') # Average of 'quantity'
    )
    print(f"Parquet (Pandas) time: {round(time.time() - start, 2)} sec")
    return result                                # Return aggregated DataFrame

# ============================
# Step 4: Benchmark DuckDB (Query Parquet Directly)
# ============================
def benchmark_duckdb_parquet():
    start = time.time()                          # Record start time
    con = duckdb.connect()                       # Create an in-memory DuckDB connection
    # Run SQL query directly on Parquet file
```

```
    # Run SQL query directly on Parquet file
    result = con.execute(f"""
        SELECT product,
               SUM(price) AS total_sales,
               AVG(quantity) AS avg_quantity
        FROM parquet_scan('{parquet_path}')
        GROUP BY product
    """).fetchdf()                                # Fetch results as Pandas DataFrame
    print(f"Parquet (DuckDB) time: {round(time.time() - start, 2)} sec")
    return result                                 # Return aggregated DataFrame

# ============================
# Step 5: Run all benchmarks
# ============================
csv_result   = benchmark_pandas_csv()             # CSV read & transform using Pandas
pq_result    = benchmark_pandas_parquet()         # Parquet read & transform using Pandas
duck_result  = benchmark_duckdb_parquet()         # Direct SQL on Parquet using DuckDB

# ============================
# Step 6: Display results
# ============================
print("\n=== Benchmark Results ===")
print("\n--- CSV Pandas ---")
print(csv_result)                                 # Show Pandas CSV aggregation
print("\n--- Parquet Pandas ---")
print(pq_result)                                  # Show Pandas Parquet aggregation
print("\n--- DuckDB Parquet ---")
print(duck_result)                                # Show DuckDB aggregation results
```

```
CSV (Pandas) time: 0.09 sec
Parquet (Pandas) time: 0.04 sec
Parquet (DuckDB) time: 0.02 sec

=== Benchmark Results ===

--- CSV Pandas ---
             total_sales  avg_quantity
product
Doodad        6956883.45      4.990542
Gadget        6816435.56      5.005479
Thingam             0.00           NaN
Thingamajig   6864554.15      4.979009
Widget        6873191.00      5.049581

--- Parquet Pandas ---
             total_sales  avg_quantity
product
Doodad        6956883.45      4.990542
Gadget        6816435.56      5.005479
Thingam             0.00           NaN
Thingamajig   6864554.15      4.979009
Widget        6873191.00      5.049581

--- DuckDB Parquet ---
       product  total_sales  avg_quantity
0       Gadget   6816435.56      5.005479
1       Doodad   6956883.45      4.990542
2  Thingamajig   6864554.15      4.979009
3       Widget   6873191.00      5.049581
4      Thingam          NaN           NaN
```

```
Start coding or generate with AI.
```