01_Bronze_Ingestion (Python)

Import notebook

CVE LAKEHOUSE - Bronze Layer

Bronze Layer Configuration

```
from pyspark.sql.functions import col, to_timestamp, year
import time

PARQUET_SOURCE_PATH = "/Volumes/workspace/default/cve_lakehouse_data/2024_parquet.parquet"
TEMP_JSON_DIR = "/Volumes/workspace/default/cve_lakehouse_data/staging_json"
BRONZE_OUTPUT_PATH = "/Volumes/workspace/default/cve_lakehouse_data/bronze"
BRONZE_TABLE_NAME = "cve_lakehouse.bronze_records"
TARGET_YEAR = 2024

spark.conf.set("spark.sql.shuffle.partitions", "8")

print(f"Configuration loaded: {TARGET_YEAR} CVEs")
```

```
Configuration loaded: 2024 CVEs
```

Read Parquet and Stage JSON

```
start_time = time.time()

print("Reading from Parquet...")
df_raw = spark.read.format("parquet").load(PARQUET_SOURCE_PATH)

print("Converting to JSON format...")
try:
    dbutils.fs.rm(TEMP_JSON_DIR, recurse=True)
except:
    pass

df_raw.select("json_data").write.mode("overwrite").text(TEMP_JSON_DIR)
print("JSON staged successfully")
```

▸ df_raw: pyspark.sql.connect.dataframe.DataFrame = [json_data: string]

```
Reading from Parquet...
Converting to JSON format...
JSON staged successfully
```

Parse JSON to Structured Format

;

```
print("Reading as JSON...")
parquet_read_start = time.time()

df_parsed = spark.read.json(TEMP_JSON_DIR)
raw_count = df_parsed.count()

parquet_read_time = time.time() - parquet_read_start
print(f"Loaded and parsed {raw_count:,} records in {parquet_read_time:.2f}s")
```

▸ df_parsed: pyspark.sql.connect.dataframe.DataFrame = [containers: struct, cveMetadata: struct ... 2 more fields]

```
Reading as JSON...
Loaded and parsed 38,727 records in 1.54s
```

Filter to 2024 and Quality Checks

```
print("Filtering to 2024...")
filter_start = time.time()

date_col = col("cveMetadata.datePublished").cast("string")
df_parsed = df_parsed.withColumn("_datePublished_ts", to_timestamp(date_col))
df_2024 = df_parsed.filter(year(col("_datePublished_ts")) == TARGET_YEAR)

cnt_2024 = df_2024.count()
null_ids = df_2024.filter(col("cveMetadata.cveId").isNull()).count()
distinct_ids = df_2024.select("cveMetadata.cveId").distinct().count()

filter_time = time.time() - filter_start

print(f"\nData Quality Checks:")
print(f"Total: {raw_count:,}")
print(f"2024 filtered: {cnt_2024:,}")
print(f"Null IDs: {null_ids}")
print(f"Distinct IDs: {distinct_ids:,}")

assert cnt_2024 >= 30000, f"Too few rows: {cnt_2024:,}"
assert null_ids == 0, "Null IDs found"
assert distinct_ids == cnt_2024, "IDs not unique"
print("Quality checks passed")
```

▸ df_2024: pyspark.sql.connect.dataframe.DataFrame = [containers: struct, cveMetadata: struct ... 3 more fields]
▸ df_parsed: pyspark.sql.connect.dataframe.DataFrame = [containers: struct, cveMetadata: struct ... 3 more fields]

```
Filtering to 2024...

Data Quality Checks:
Total: 38,727
2024 filtered: 32,924
Null IDs: 0
Distinct IDs: 32,924
Quality checks passed
```

Write Bronze Delta

```
print("\nWriting to Delta...")
delta_start = time.time()

try:
    dbutils.fs.rm(BRONZE_OUTPUT_PATH, recurse=True)
except:
    pass

(df_2024
 .repartition(8)
 .write
 .format("delta")
 .mode("overwrite")
 .option("delta.columnMapping.mode", "name")
 .save(BRONZE_OUTPUT_PATH))

delta_time = time.time() - delta_start
print(f"Delta write completed in {delta_time:.2f}s")
```

```
Writing to Delta...
Delta write completed in 3.73s
```

Cleanup and Summary

```
print("Cleaning up temp files...")
try:
    dbutils.fs.rm(TEMP_JSON_DIR, recurse=True)
except:
    pass

total_time = time.time() - start_time
print(f"\nTiming: Parse+read {parquet_read_time:.2f}s | Filter {filter_time:.2f}s | Delta {delta_time:.2f}s |
Total {total_time:.2f}s")
print(f"\nDelta files written to: {BRONZE_OUTPUT_PATH}")
print(f"Records: {cnt_2024:,}")
```

```
Cleaning up temp files...

Timing: Parse+read 1.54s | Filter 1.55s | Delta 3.73s | Total 36.98s

Delta files written to: /Volumes/workspace/default/cve_lakehouse_data/bronze
Records: 32,924
```

Register Bronze Table (ALTERNATIVE)

```
# Read Delta directly and register as temp view
df_bronze = spark.read.format("delta").load(BRONZE_OUTPUT_PATH)
df_bronze.createOrReplaceTempView("bronze_records")

print(f"Table available as: bronze_records")
print(f"Records: {df_bronze.count():,}")
```

```
▸ df_bronze:  pyspark.sql.connect.dataframe.DataFrame = [containers: struct, cveMetadata: struct ... 3 more fields]
Table available as: bronze_records
Records: 32,924
```

Verification

```
print("Bronze Layer Verification\n")

# Show sample records
spark.sql("SELECT * FROM bronze_records LIMIT 5").show(truncate=False)

# Show specific columns
spark.sql("""
    SELECT
        cveMetadata.cveId as CVE_ID,
        cveMetadata.datePublished as Published,
        cveMetadata.state as State
    FROM bronze_records
    LIMIT 10
""").show(truncate=False)

# Final count
final_count = spark.sql("SELECT COUNT(*) as total FROM bronze_records").collect()[0]['total']
print(f"\n🎉 Bronze Layer Complete: {final_count:,} records")
```

```
-------------------------------------------------------------------+-----------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------
--------+----------+-----------+----------------------+

+--------------+-----------------------+----------+
|CVE_ID        |Published              |State     |
+--------------+-----------------------+----------+
|CVE-2024-0968 |2024-03-02T21:38:41.309Z|REJECTED |
|CVE-2024-37353|2024-06-21T10:18:10.995Z|REJECTED |
|CVE-2024-11879|2024-12-14T04:23:40.550Z|REJECTED |
|CVE-2024-0035 |2024-02-16T00:08:17.297Z|PUBLISHED|
|CVE-2024-0045 |2024-03-11T16:35:21.876Z|PUBLISHED|
|CVE-2024-0014 |2024-02-16T00:08:14.746Z|PUBLISHED|
|CVE-2024-0151 |2024-04-24T17:12:43.184Z|PUBLISHED|
|CVE-2024-0213 |2024-01-09T13:01:13.209Z|PUBLISHED|
|CVE-2024-0167 |2024-02-12T18:23:44.036Z|PUBLISHED|
|CVE-2024-0165 |2024-02-12T18:30:52.482Z|PUBLISHED|
+--------------+-----------------------+----------+


🎉 Bronze Layer Complete: 32,924 records
```

Register Bronze Table with Column Mapping

```
▸ df_bronze:  pyspark.sql.connect.dataframe.DataFrame = [containers: struct, cveMetadata: struct ... 3 more fields]
Delta table contains 32,924 records
Table registered: cve_bronze.records
+--------------+--------------------+
|         cveId|       datePublished|
+--------------+--------------------+
| CVE-2024-5758|2024-06-08T06:54:...|
|CVE-2024-46503| 2024-09-30T00:00:00|
| CVE-2024-0015|2024-02-16T18:33:...|
| CVE-2024-0018|2024-02-16T19:33:...|
| CVE-2024-0033|2024-02-16T00:08:...|
+--------------+--------------------+

+-------------+
```

```
|total_records|
+-------------+
|        32924|
+-------------+
```