INDIAN DATA CLUB | ODE BASICS

databricks

# Day 06: Implementing Medallion Architecture

## Databricks 14 Days AI Challenge


Translating data lakehouse theory into executable PySpark pipelines.

#DatabricksWithIDC

INDIAN DATA CLUB

# The Day 06 Challenge Scope

Today's mission parameters based on the challenge prompt:

- Design a 3-layer architecture
- Build Bronze: Establish raw ingestion
- Build Silver: Implement cleaning and validation logic
- Build Gold: Generate business aggregates and KPIs

# The Logic of the Medallion Architecture


**Raw Ingestion**

Historical data captured “as-is” from source.

**Refined & Validated**

Filtered, deduplicated, and enriched. The Enterprise View.

**Business Aggregates**

KPIs and metrics optimized for Dashboards & ML.

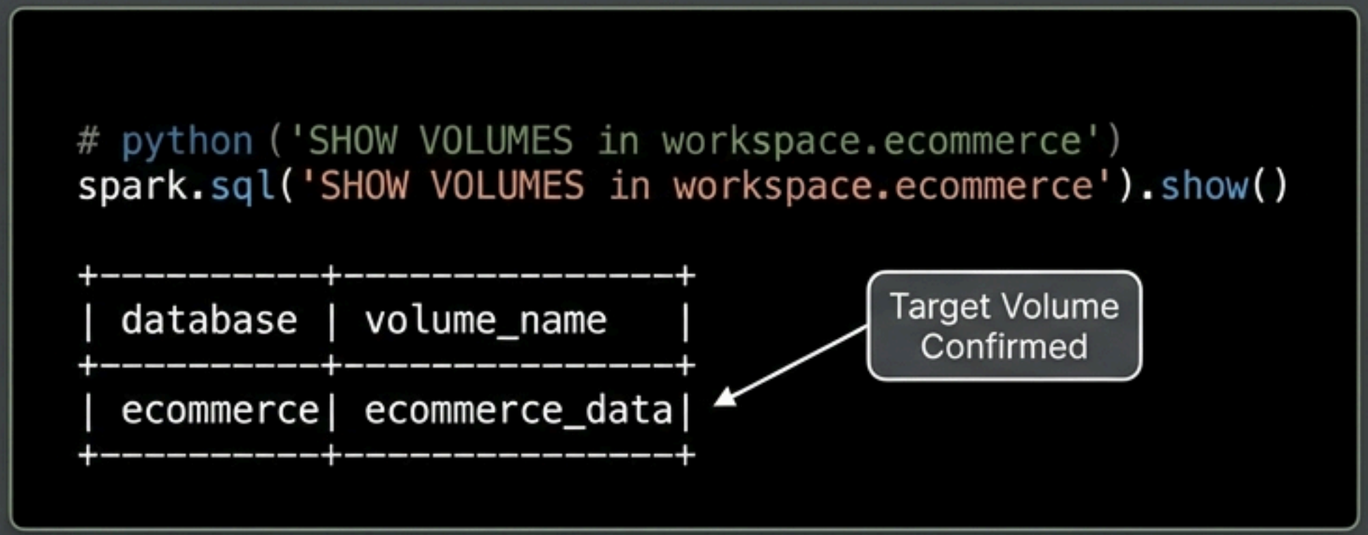
# Verifying the Storage Environment

Before ingestion, we confirm the availability of the storage volumes.

In a Unity Catalog environment, we manage data in logical volumes rather than raw file paths.

```
# python ('SHOW VOLUMES in workspace.ecommerce')
spark.sql('SHOW VOLUMES in workspace.ecommerce').show()

+---------+--------------+
| database | volume_name   |
+----------+---------------+
| ecommerce| ecommerce_data|
+---------+--------------+
```



# Bronze Layer: Ingesting Raw Data

**Goal:** Capture raw CSVs and add audit trails.

```
raw = spark.read.csv('/Volumes/.../2019-Nov.csv', header=True, inferSchema=True)

bronze = raw.withColumn('ingestion_ts', F.current_timestamp())

bronze.write.format('delta').mode('overwrite').save('.../bronze_events')
```

**Auditability:** Captures exact arrival time.

**Reliability:** Converts CSV to Delta Lake format.

# Silver Layer Part 1: Filtering & Deduplication

**Goal:** Remove noise and enforce data quality.

```
silver = bronze.filter((F.col('price') < 10000) & (F.col('price') > 0)) \

    .dropDuplicates(['user_session', 'event_time'])
```

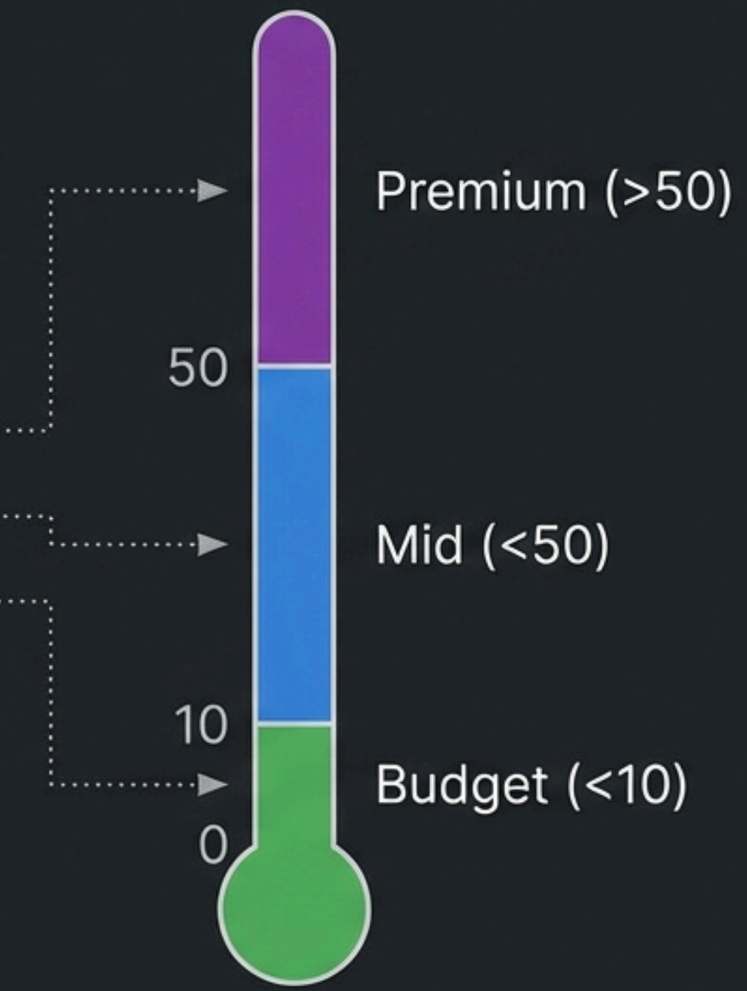
Removes negative prices and outliers.


Ensures distinct session events.

# Silver Layer Part 2: Enrichment

**Goal:** Apply business logic to categorize data.

```
.withColumn('price_tier',
    F.when(F.col('price') < 10, 'budget')
    .when(F.col('price') < 50, 'mid')
    .otherwise('premium'))
```



# Gold Layer: Pivoting to Product Performance

**Goal:** Aggregate events into product-level metrics.

```
product_perf = silver.groupBy('product_id') \
    .agg(
        F.countDistinct(F.when(F.col('event_type')=='view', 'user_id')).alias('views'),
        F.countDistinct(F.when(F.col('event_type')=='purchase', 'user_id'))
            .alias('purchases'),
        F.sum(F.when(F.col('event_type')=='purchase',F.expr('try_cast(price as double)')))
            .alias('revenue')
    )
```

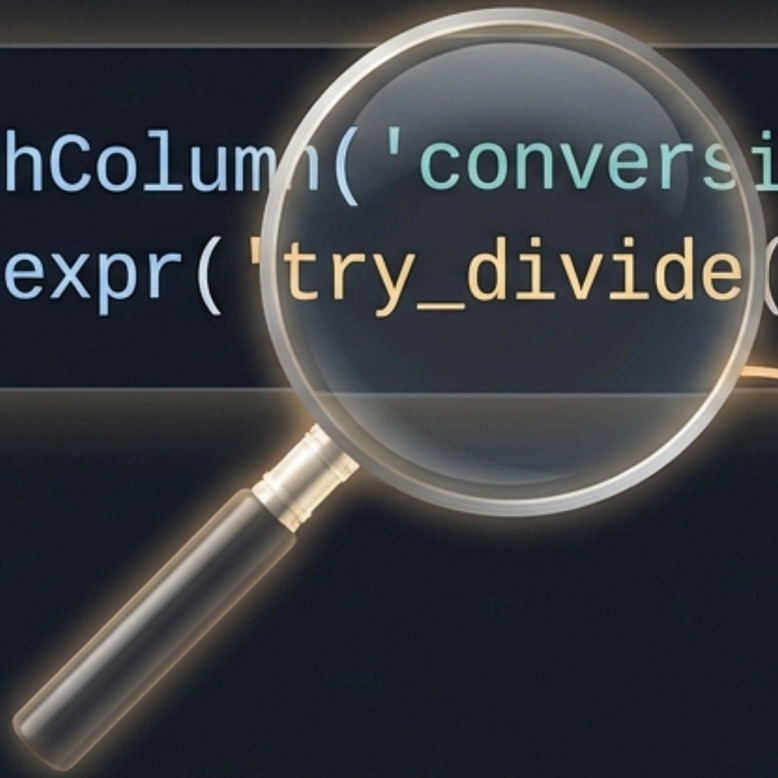
Millions of Events

One Row Per Product

# Robust KPI Calculation: Handling Edge Cases

Preventing pipeline failures with safe arithmetic

```
.withColumn('conversion_rate',
  F.expr('try_divide(purchases, views) * 100'))
```



**CRITICAL SAFETY CHECK:**

If a product has 0 views, standard division throws a 'Division by Zero' error. `try_divide` returns NULL instead, **allowing the pipeline to finish successfully.**

# The Resulting Gold Schema

The final ‘product_perf’ table structure ready for BI consumption:

| Column Name | Data Type |
| --- | --- |
| product_id | Integer (Key) |
| views | Long |
| purchases | Long |
| revenue | Double |
| conversion_rate | Double |

# Full Pipeline Recap


Data Quality & Value Increases →

INDIAN DATA CLUB

CODE BASICS

# Mission Accomplished: Day 06

## Medallion Architecture Implemented Successfully.

We have built a robust, self-correcting pipeline that handles data quality issues and mathematical edge cases.

## Ready for Day 07

INDIAN DATA CLUB