**Workshop Lab 2: Building a Medallion Architecture in Databricks**

**Objective**

Demonstrate a complete data engineering pipeline using Databricks features to transform raw CSV data into bronze, silver, and gold tables.

**Step 1: Create the Medallion schema**

CREATE SCHEMA training.bronze;

CREATE SCHEMA training.silver;

CREATE SCHEMA training.gold;

**Step 2: Create Bronze Layer (Raw Ingestion)**

-- Create bronze tables from raw CSV files

CREATE OR REPLACE TABLE training.bronze.customers

AS SELECT \* FROM csv.`/Volumes/training/default/landing/customer.csv` WITH (header="true");

CREATE OR REPLACE TABLE training.bronze.products

AS SELECT \* FROM csv.`/Volumes/training/default/landing/product.csv` WITH (header="true");

CREATE OR REPLACE TABLE training.bronze.orders

AS SELECT \* FROM csv.`/Volumes/training/default/landing/order.csv` WITH (header="true");

**Step 3: Create Silver Layer (Cleaned & Enriched Data)**

-- Clean customers: Standardize names and emails

CREATE OR REPLACE TABLE training.silver.customers

AS SELECT

customer\_id,

initcap(first\_name) AS first\_name,

initcap(last\_name) AS last\_name,

lower(email) AS email,

initcap(city) AS city

FROM training.bronze.customers;

-- Clean products: Handle missing values and data types

CREATE OR REPLACE TABLE training.silver.products

AS SELECT

product\_id,

initcap(product\_name) AS product\_name,

initcap(category) AS category,

CAST(price AS DECIMAL(10,2)) AS price

FROM training.bronze.products

WHERE price IS NOT NULL;

-- Enrich orders: Add calculated columns and validation

CREATE OR REPLACE TABLE training.silver.orders

AS SELECT

o.order\_id,

o.customer\_id,

o.product\_id,

CAST(o.order\_date AS DATE) AS order\_date,

CAST(o.quantity AS INT) AS quantity,

CAST(p.price AS DECIMAL(10, 2)) AS price,

(CAST(o.quantity AS INT) \* CAST(p.price AS DECIMAL(10, 2))) AS order\_total,

CASE WHEN CAST(o.quantity AS INT) > 0 THEN 'Valid' ELSE 'Invalid' END AS status

FROM training.bronze.orders o

JOIN training.bronze.products p

ON o.product\_id = p.product\_id;

**Step 4: Create Gold Layer (Business Aggregates)**

-- Customer Lifetime Value (CLV)

CREATE OR REPLACE TABLE training.gold.customer\_clv

AS SELECT

c.customer\_id,

c.first\_name,

c.last\_name,

c.city,

SUM(o.order\_total) AS total\_spend,

COUNT(o.order\_id) AS order\_count,

AVG(o.order\_total) AS avg\_order\_value

FROM training.silver.customers c

JOIN training.silver.orders o

ON c.customer\_id = o.customer\_id

WHERE o.status = 'Valid'

GROUP BY c.customer\_id, c.first\_name, c.last\_name, c.city;

-- Product Performance

CREATE OR REPLACE TABLE training.gold.product\_performance

AS SELECT

p.product\_id,

p.product\_name,

p.category,

SUM(o.quantity) AS total\_units\_sold,

SUM(o.order\_total) AS total\_revenue,

AVG(p.price) AS avg\_price

FROM training.silver.products p

JOIN training.silver.orders o

ON p.product\_id = o.product\_id

WHERE o.status = 'Valid'

GROUP BY p.product\_id, p.product\_name, p.category;

-- Daily Sales Summary

CREATE OR REPLACE TABLE training.gold.daily\_sales

AS SELECT

order\_date,

SUM(order\_total) AS daily\_revenue,

COUNT(order\_id) AS daily\_orders,

SUM(quantity) AS daily\_units

FROM training.silver.orders

WHERE status = 'Valid'

GROUP BY order\_date;

**Step 5: Verify Pipeline**

-- Check row counts  
SELECT   
 (SELECT COUNT(\*) FROM training.sales.bronze\_customers) AS bronze\_customers,  
 (SELECT COUNT(\*) FROM training.sales.silver\_customers) AS silver\_customers,  
 (SELECT COUNT(\*) FROM training.sales.gold\_customer\_clv) AS gold\_customers;  
   
-- Sample gold data  
SELECT \* FROM training.sales.gold\_product\_performance   
ORDER BY total\_revenue DESC   
LIMIT 5;

**Pipeline Summary**

|  |  |  |  |
| --- | --- | --- | --- |
| **Layer** | **Table** | **Description** | **Features Demonstrated** |
| **Bronze** | bronze.customers | Raw ingested data | CSV ingestion, volume access |
|  | bronze.products | Raw ingested data | Schema inference |
|  | bronze.orders | Raw ingested data | Volume path handling |
| **Silver** | silver.customers | Cleaned customer data | Data standardization, initcap() |
|  | silver.products | Validated products | Data validation, casting |
|  | silver.orders | Enriched orders | Joins, calculated columns, CASE statements |
| **Gold** | gold.customer\_clv | Business metrics | Aggregations, SUM(), AVG() |
|  | gold.product\_performance | Product analytics | Multi-column aggregation |
|  | gold.daily\_sales | Time-based reporting | Date-based grouping |

**Next Steps**

1. Add data quality checks with EXPECTATIONS
2. Schedule pipeline with Databricks Workflows
3. Create visualizations in Databricks SQL
4. Implement incremental processing with Delta Lake