

Ex No: 4	Building and Automating Pipeline in Databricks for an E-Commerce Dataset
Date: 11-09-2025	

Objective:

To design and implement a data pipeline on Databricks using a healthcare dataset, following the Medallion architecture (Bronze → Silver → Gold). The pipeline should ingest raw data, clean and transform it, perform aggregations, and finally automate execution with dashboards and notifications.

Outcomes:

1. A fully functional end-to-end pipeline in Databricks.
2. Clean and aggregated data stored in Bronze, Silver, and Gold tables.
3. Automated dashboards for revenue insights.
4. Scheduled execution with email notifications for monitoring.

Materials:

Software & Platform

- Databricks Community Edition / Enterprise Edition
- Apache Spark (PySpark API)
- Databricks SQL and Notebooks
- Databricks Jobs (for scheduling & automation)

Dataset:-

- `healthcare_orders.csv`

Lab Procedure:

Step 1: Ingestion & Cleaning (Bronze → Silver)

1. Upload the healthcare_orders.csv dataset into Databricks File System (DBFS).
2. Create a Bronze table (healthcare_orders) to store the raw data.
3. Use PySpark to clean the data:
 - Convert order_date to a standard format (yyyy-MM-dd).
 - Compute total_value = quantity × price.
4. Save the cleaned dataset as a Silver table (silver_healthcare_orders).

Step 2: Aggregation & Enrichment (Silver → Gold)

1. Load the Silver table into a DataFrame.
2. Perform aggregations:
 - Calculate total revenue by service category → store as gold_healthcare_service_sales.
 - Calculate daily revenue trends → store as gold_healthcare_daily_sales.
3. Save both results as Gold tables for reporting and analytics.

Step 3: Dashboard & Visualization

1. Use Databricks Notebooks to create interactive dashboards.
2. Add the following visualizations:
 - Bar Chart → Revenue by service category.
 - Line Chart → Daily revenue trends.
 - Filters → City and payment method filters for drill-down analysis.

Step 4: Automation & Scheduling

1. Convert the pipeline into a Databricks Job.
2. Configure a schedule (daily/weekly) for automatic pipeline execution.
3. Ensure that Bronze → Silver → Gold tables are refreshed automatically.
4. Automate dashboard updates after each run.

Step 5: Email Notifications & Alerts

1. Configure email alerts in the Databricks Job settings.
2. Enable:
 - Success notifications → sent to administrators.
 - Failure alerts → sent to data engineers for debugging.

USN:1RVU23CSE153

Name:Ekshu DP

Task_1_Ingestion_Cleaning_healthcare:-

The screenshot shows a Jupyter Notebook cell with the following code:

```
from pyspark.sql.functions import col, to_date
df_raw = spark.table("healthcare_orders")
df_cleaned = (
    df_raw.withColumn("order_date", to_date(col("order_date"), "yyyy-MM-dd"))
    .withColumn("total_value", col("quantity") * col("price"))
)
# Save as a managed Silver table
df_cleaned.createOrReplaceTempView("tmp_healthcare_orders_cleaned")
spark.sql("""
CREATE OR REPLACE TABLE silver_healthcare_orders AS
SELECT * FROM tmp_healthcare_orders_cleaned
""")
> See performance (2)
```

Below the code, there are two expanded sections:

- df_raw: pyspark.sql.connect.DataFrame
- df_cleaned: pyspark.sql.connect.DataFrame

Both are described as DataFrame[`num_affected_rows: bigint, num_inserted_rows: bigint`]

Task2_Aggregation_Reportng_healthcare:-

The screenshot shows a Jupyter Notebook cell with the following code:

```
from pyspark.sql.functions import sum as spark_sum
df_cleaned = spark.table("silver_healthcare_orders")
# Revenue by product category (Gold layer)
df_category_sales = (
    df_cleaned.groupBy("service_category")
    .agg(spark_sum("total_value").alias("total_revenue"))
)
df_category_sales.createOrReplaceTempView("tmp_healthcare_category_sales")
spark.sql("""
CREATE OR REPLACE TABLE gold_healthcare_category_sales AS
SELECT * FROM tmp_healthcare_category_sales
""")
df_daily_sales = (
    df_cleaned.groupBy("order_date")
    .agg(spark_sum("total_value").alias("daily_revenue"))
)
df_daily_sales.createOrReplaceTempView("tmp_healthcare_daily_sales")

spark.sql("""
CREATE OR REPLACE TABLE gold_healthcare_daily_sales AS
SELECT * FROM tmp_healthcare_daily_sales
""")
> See performance (4)
```

Below the code, there are three expanded sections:

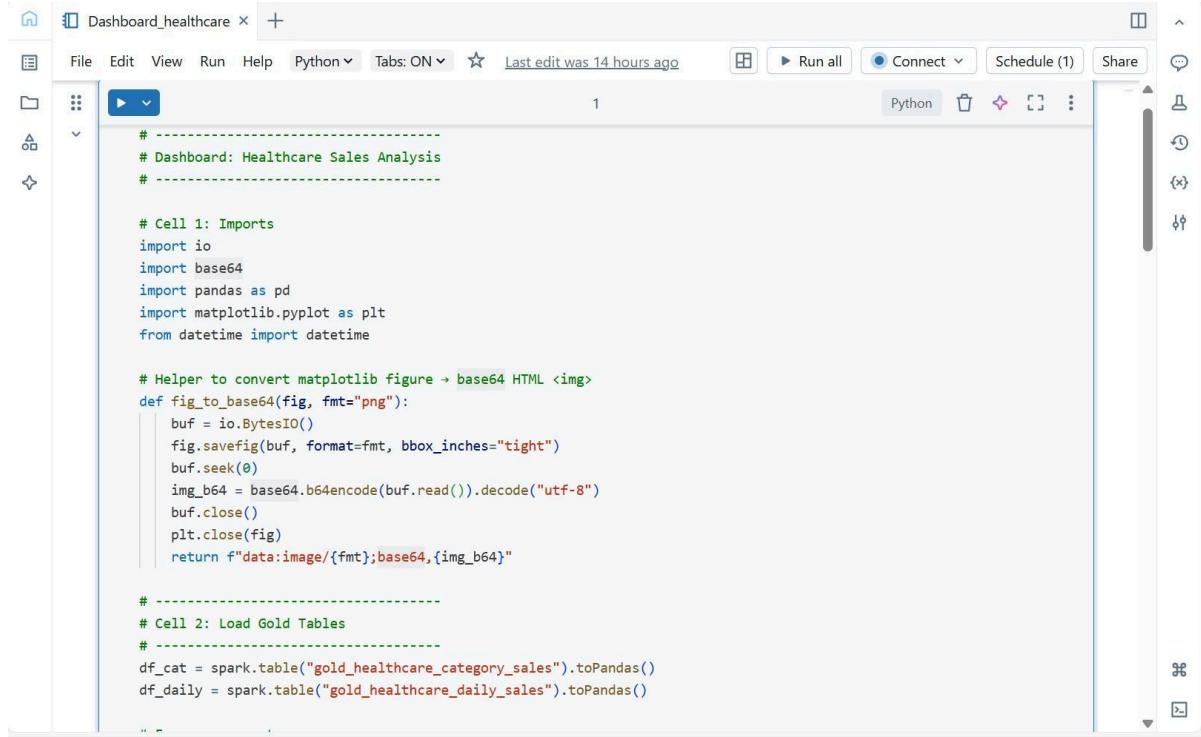
- df_cleaned: pyspark.sql.connect.DataFrame
- df_category_sales: pyspark.sql.connect.DataFrame
- df_daily_sales: pyspark.sql.connect.DataFrame

Both are described as DataFrame[`num_affected_rows: bigint, num_inserted_rows: bigint`]

USN:1RVU23CSE153

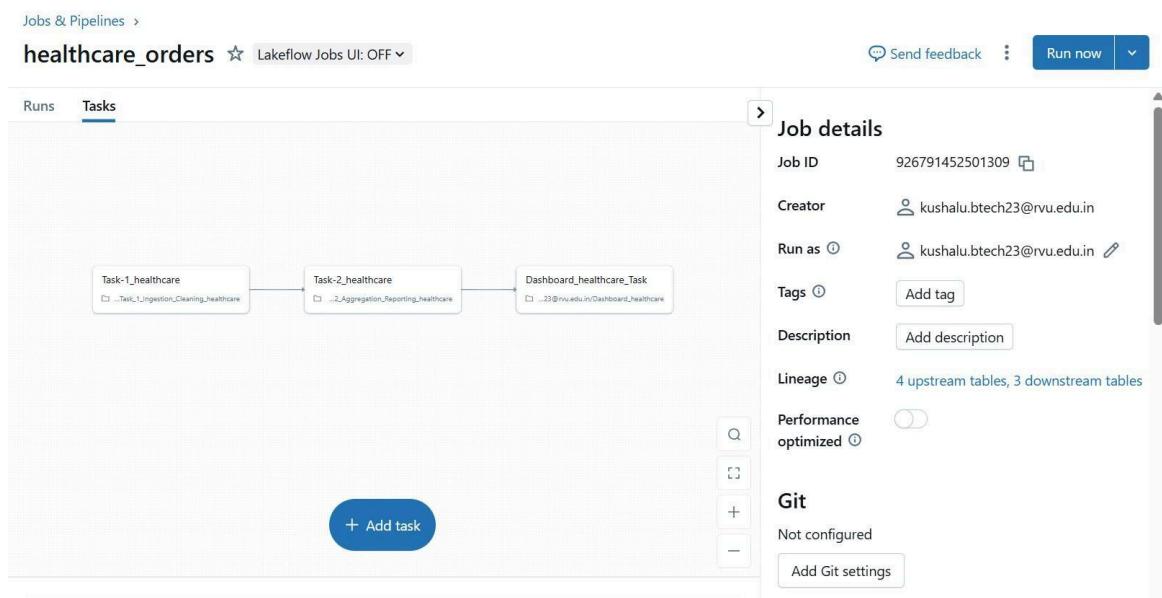
Name:Ekshu DP

Dashboard_healthcare:-



```
# -----  
# Dashboard: Healthcare Sales Analysis  
# -----  
  
# Cell 1: Imports  
import io  
import base64  
import pandas as pd  
import matplotlib.pyplot as plt  
from datetime import datetime  
  
# Helper to convert matplotlib figure → base64 HTML <img>  
def fig_to_base64(fig, fmt="png"):  
    buf = io.BytesIO()  
    fig.savefig(buf, format=fmt, bbox_inches="tight")  
    buf.seek(0)  
    img_b64 = base64.b64encode(buf.read()).decode("utf-8")  
    buf.close()  
    plt.close(fig)  
    return f"data:image/{fmt};base64,{img_b64}"  
  
# -----  
# Cell 2: Load Gold Tables  
# -----  
df_cat = spark.table("gold_healthcare_category_sales").toPandas()  
df_daily = spark.table("gold_healthcare_daily_sales").toPandas()
```

Job Pipeline:-



USN:1RVU23CSE153

Name:Ekshu DP

Jobs & Pipelines >

healthcare_orders ★ Lakeflow Jobs UI: OFF ▾

Send feedback

Run now

Runs Tasks

Runs

Started before



Previous

Next >

Run total duration
7m 16s
3m 38s

Sep 10

Task-1_healthcare

Task-2_healthcare

Dashboard_healthcare_Task

Go to the latest successful run

Cancel runs ▾

Start time	Run ID	Launched	Duration	Status	Error code	Run para...	⋮
Sep 10, 2025, ...	43121710...	Manually	7m 16s	Success			...
Sep 10, 2025, ...	84367200...	Manually	6m 31s	Failed	RunExecut...		...

Job notifications ⓘ

kushalu.btech23@rvu.edu.in

On success, On failure

Edit notifications

Duration and streaming backlog thresholds ⓘ

No thresholds defined

Add metric thresholds

Permissions ⓘ

kushalu.btech23@rvu.edu.in

Is Owner

admins

Can Manage

Edit permissions

Advanced settings ▾

Jobs & Pipelines > healthcare_orders > Run 431217107113341 >

Dashboard_healthcare_Task run ⚡ Succeeded Lakeflow Jobs UI: OFF ▾

Edit task ↗ Repair task

Output

Hide code

Export as HTML

```
▶ df_cat: pandas.core.frame.DataFrame = [service_category: object, total_revenue: int64]  
▶ df_daily: pandas.core.frame.DataFrame = [order_date: datetime64[ns], daily_revenue: int64]
```

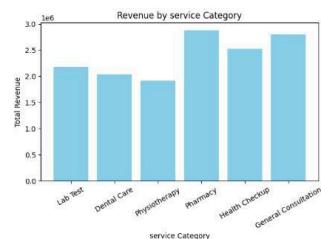
E-Commerce Sales Dashboard

Last Updated: 2025-09-10 15:06:49

Total Revenue
₹ 14,338,429.00

Top Category
Pharmacy

Revenue by Category



Daily Revenue Trend



Task run

Details Metrics ⓘ

Details

Job ID	926791452501309
Job run ID	431217107113341
Task run ID	477716038857901
Run as	kushalu.btech23@rvu.edu.in
Launched	Manually
Started	Sep 10, 2025, 08:36 PM
Ended	Sep 10, 2025, 08:36 PM
Duration	15s
Queue duration	-
Status	Success

USN:1RVU23CSE153
Name:Ekshu DP

Email Notification:-

The screenshot shows an email inbox interface with a message from 'prod-monitoring@databricks.com' about a completed job run.

Inbox Summary: 940 messages

Message Details:

- From:** prod-monitoring@databricks.com (External)
- To:** to me
- Date:** Wed, Sep 10, 8:36 PM (13 hours ago)
- Subject:** 'healthcare_orders'

Databricks Logo: A red icon consisting of three overlapping squares.

Message Content:

A run of this job has completed successfully

Run details

Workspace	workspace [2768814303980812]
Job	healthcare_orders [926791452501309]
Job Run	431217107113341
Status	Success Succeeded
Started at	2025-09-10 14:59:39 UTC
Duration	7m 15s
Launched	Manually