INDIAN DATA CLUB

CODE BASICS

databricks

14 DAYS

AI CHALLENGE

DAY 06

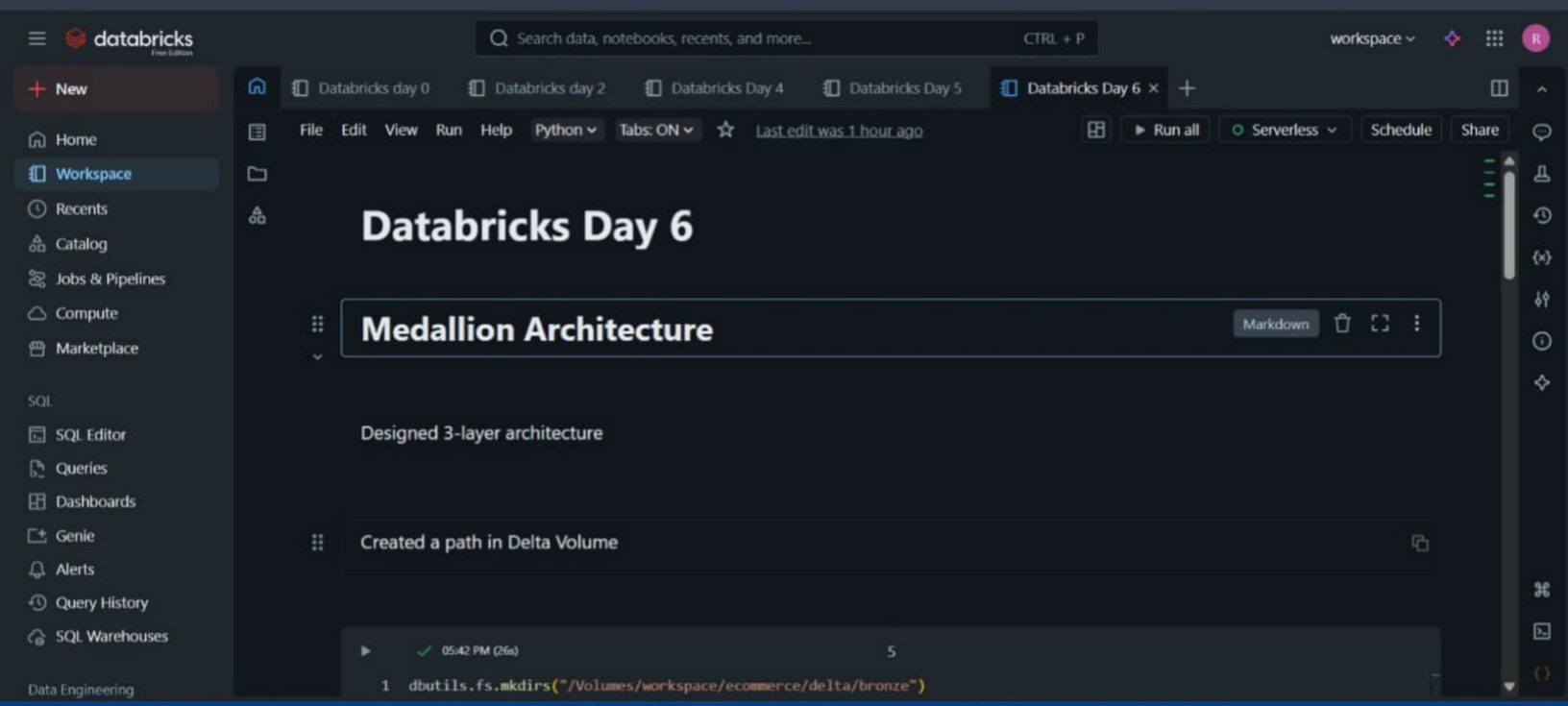
Topic:

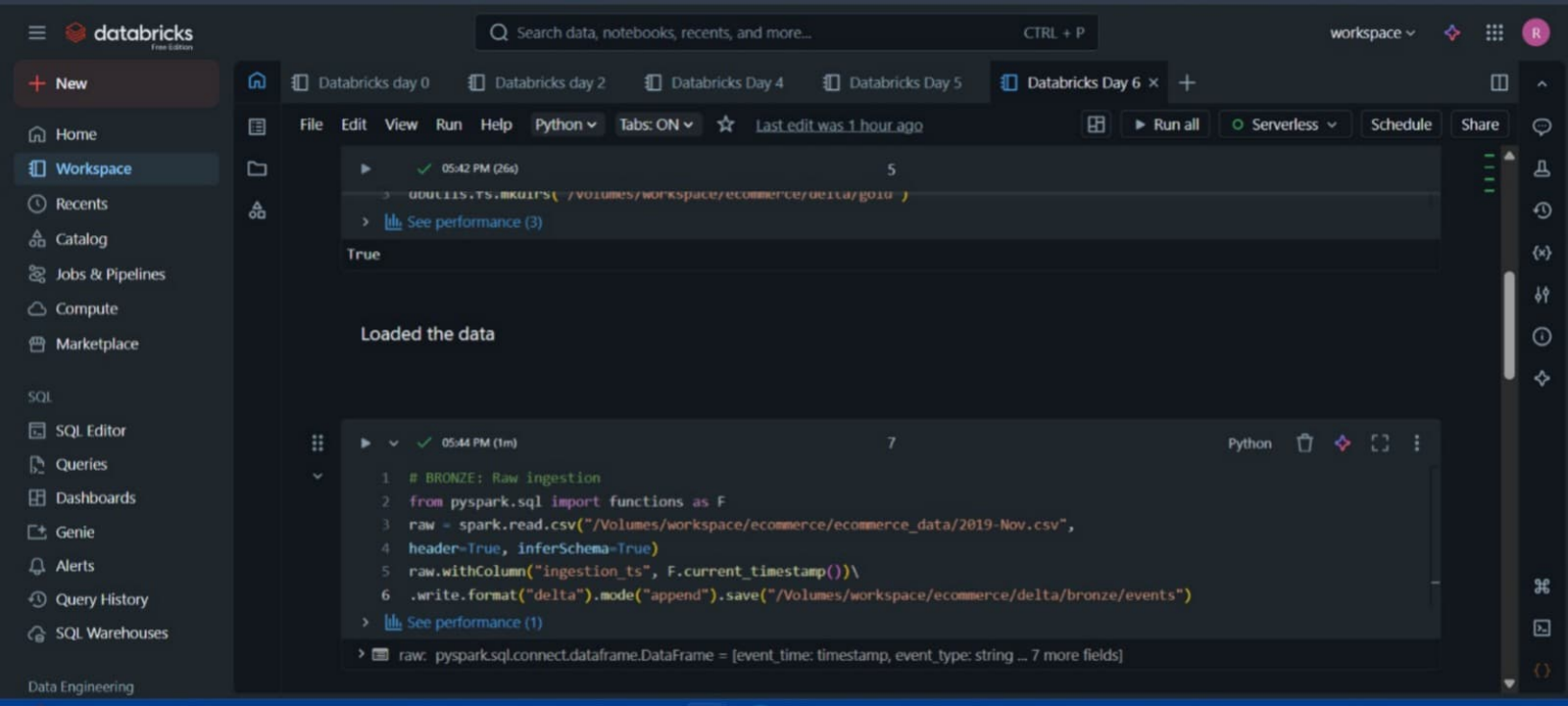
Medallion Architecture

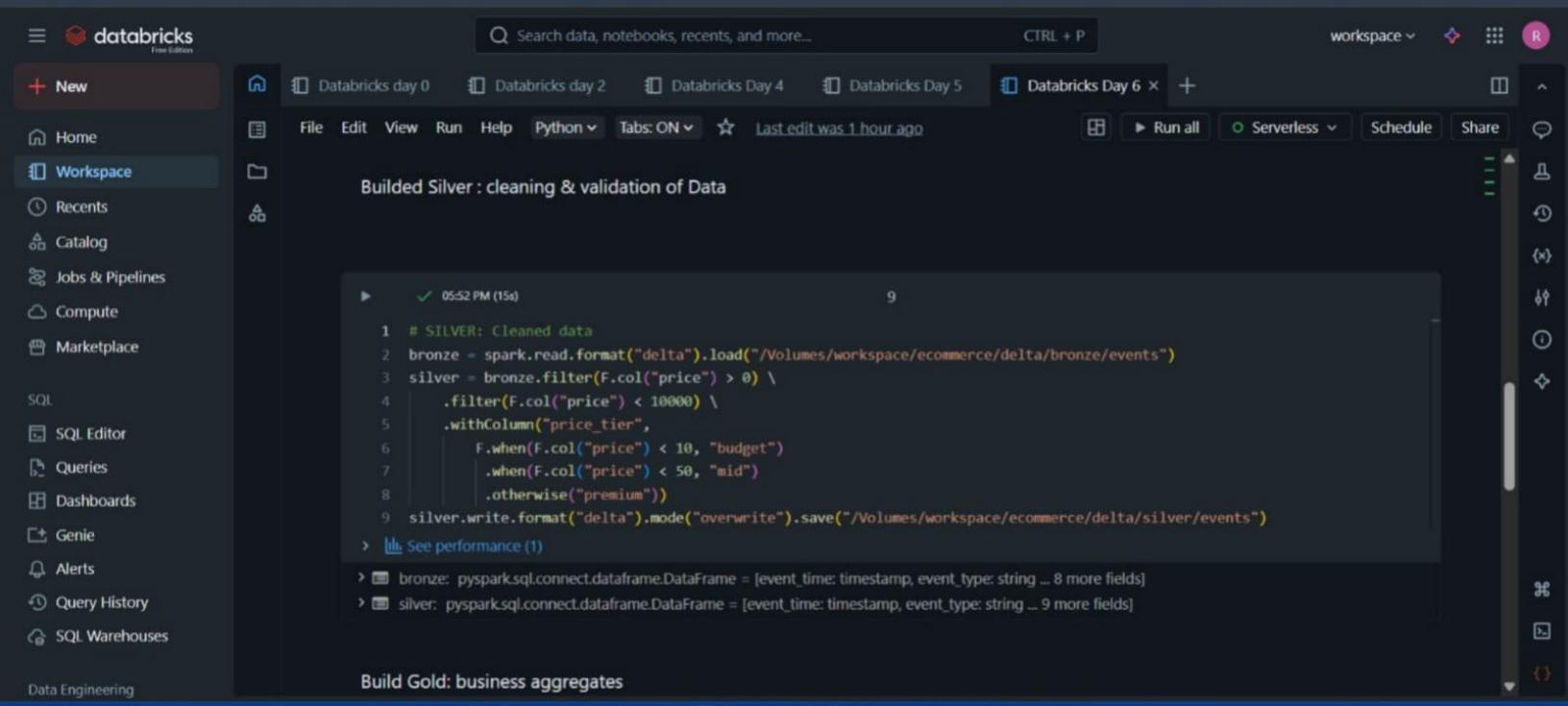
Challenge:

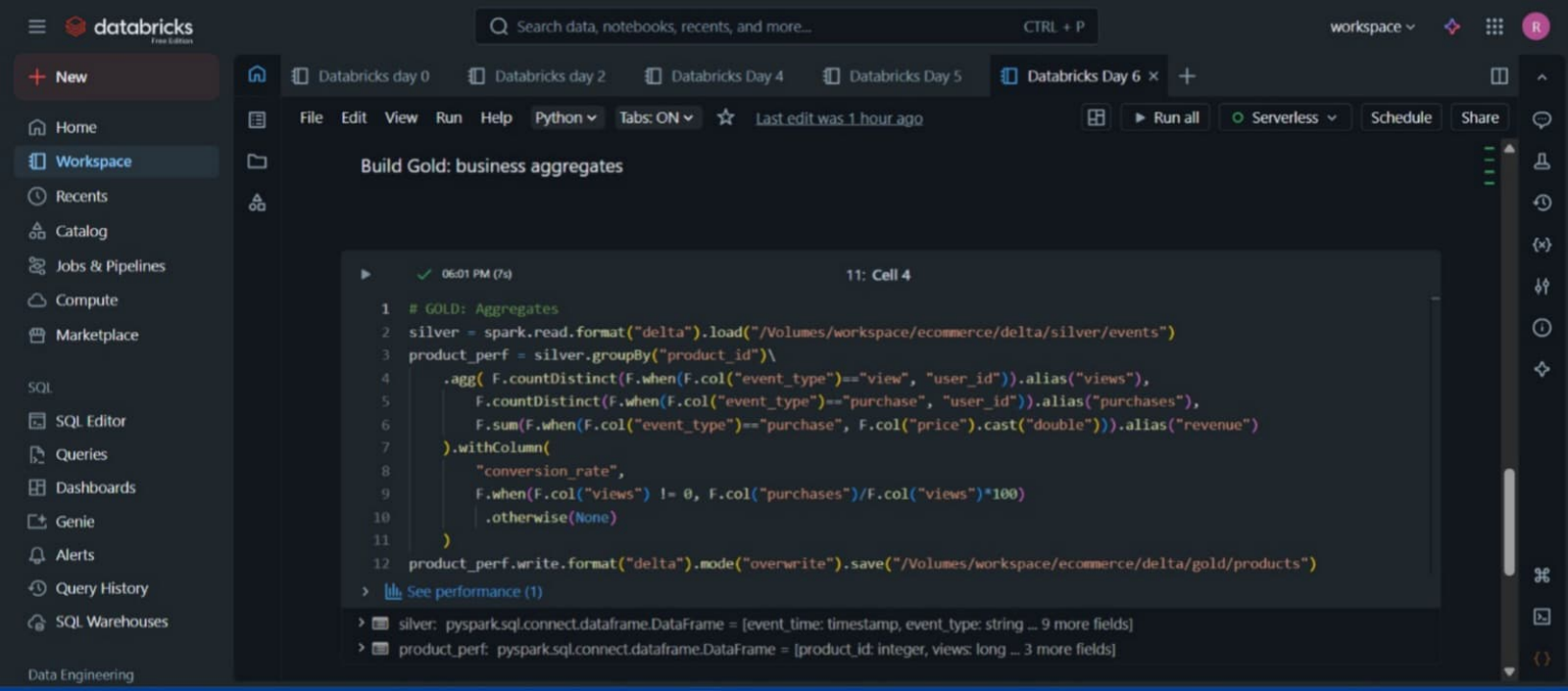
1. Design 3-layer architecture
2. Build Bronze: raw ingestion
3. Build Silver: cleaning & validation
4. Build Gold: business aggregates

#DatabricksWithIDC


databricks
Search data, notebooks, recents, and more... CTRL + P
workspace
New
Home
Workspace
Recents
Catalog
Jobs & Pipelines
Compute
Marketplace
SQL
SQL Editor
Queries
Dashboards
Genie
Alerts
Query History
SQL Warehouses
Data Engineering
Databricks day 0
Databricks day 2
Databricks Day 4
Databricks Day 5
Databricks Day 6
File Edit View Run Help Python Tabs: ON Last edit was 1 hour ago
Run all Serverless Schedule Share
Databricks Day 6
Medallion Architecture
Markdown
Designed 3-layer architecture
Created a path in Delta Volume
05:42 PM (26s)
5
1 dbutils.fs.mkdirs("/Volumes/workspace/ecommerce/delta/bronze")


databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
New
Home
Workspace
Recents
Catalog
Jobs & Pipelines
Compute
Marketplace
SQL
SQL Editor
Queries
Dashboards
Genie
Alerts
Query History
SQL Warehouses
Data Engineering
Databricks day 0
Databricks day 2
Databricks Day 4
Databricks Day 5
Databricks Day 6
File Edit View Run Help Python Tabs: ON
Last edit was 1 hour ago
Run all
Serverless
Schedule
Share
05:42 PM (26s)
5
dbutils.fs.mkdirs("/Volumes/workspace/ecommerce/delta/gold")
See performance (3)
True
Loaded the data
05:44 PM (1m)
7
Python
# BRONZE: Raw ingestion
from pyspark.sql import functions as F
raw = spark.read.csv("/Volumes/workspace/ecommerce/ecommerce_data/2019-Nov.csv",
header=True, inferSchema=True)
raw.withColumn("ingestion_ts", F.current_timestamp())\
.write.format("delta").mode("append").save("/Volumes/workspace/ecommerce/delta/bronze/events")
See performance (1)
raw: pyspark.sql.connect.dataframe.DataFrame = [event_time: timestamp, event_type: string ... 7 more fields]


databricks
Free Edition
Search data, notebooks, recents, and more...
CTRL + P
workspace
New
Home
Workspace
Recents
Catalog
Jobs & Pipelines
Compute
Marketplace
SQL
SQL Editor
Queries
Dashboards
Genie
Alerts
Query History
SQL Warehouses
Data Engineering
Databricks day 0
Databricks day 2
Databricks Day 4
Databricks Day 5
Databricks Day 6
File Edit View Run Help Python Tabs: ON
Last edit was 1 hour ago
Run all
Serverless
Schedule
Share
Builded Silver : cleaning & validation of Data
05:52 PM (15s)
9
# SILVER: Cleaned data
bronze = spark.read.format("delta").load("/Volumes/workspace/ecommerce/delta/bronze/events")
silver = bronze.filter(F.col("price") > 0) \
.filter(F.col("price") < 10000) \
.withColumn("price_tier",
F.when(F.col("price") < 10, "budget")
.when(F.col("price") < 50, "mid")
.otherwise("premium"))
silver.write.format("delta").mode("overwrite").save("/Volumes/workspace/ecommerce/delta/silver/events")
See performance (1)
bronze: pyspark.sql.connect.dataframe.DataFrame = [event_time: timestamp, event_type: string ... 8 more fields]
silver: pyspark.sql.connect.dataframe.DataFrame = [event_time: timestamp, event_type: string ... 9 more fields]
Build Gold: business aggregates


databricks
Search data, notebooks, recents, and more... CTRL + P
workspace
New
Home
Workspace
Recents
Catalog
Jobs & Pipelines
Compute
Marketplace
SQL
SQL Editor
Queries
Dashboards
Genie
Alerts
Query History
SQL Warehouses
Data Engineering
Databricks day 0
Databricks day 2
Databricks Day 4
Databricks Day 5
Databricks Day 6
File Edit View Run Help Python Tabs: ON Last edit was 1 hour ago
Run all Serverless Schedule Share
Build Gold: business aggregates
06:01 PM (7s)
11: Cell 4
# GOLD: Aggregates
silver = spark.read.format("delta").load("/Volumes/workspace/ecommerce/delta/silver/events")
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.col("price").cast("double"))).alias("revenue")
).withColumn(
"conversion_rate",
F.when(F.col("views") != 0, F.col("purchases")/F.col("views")*100)
.otherwise(None)
)
product_perf.write.format("delta").mode("overwrite").save("/Volumes/workspace/ecommerce/delta/gold/products")
See performance (1)
silver: pyspark.sql.connect.dataframe.DataFrame = [event_time: timestamp, event_type: string ... 9 more fields]
product_perf: pyspark.sql.connect.dataframe.DataFrame = [product_id: integer, views: long ... 3 more fields]