## Delta Powered Snowflake

**Objectiv**e: **Delta-Powered Snowflake** enables seamless integration between real-time data processing in Delta Lake and advanced analytics in Snowflake.

By combining Delta's efficient ingestion and change tracking with Snowflake's elastic compute and BI capabilities, teams can build reliable, low-latency pipelines from raw events to actionable insights. This architecture bridges operational and analytical worlds, powering faster decisions with trusted data.

### Step 1: ****Open Jobs UI and Import JSON****

1. Go to your **Databricks Workspace**
2. In the sidebar, click **Workflows > Jobs**
3. Click **Create Job** → Select **Import Job**
4. **Paste the JSON** from earlier or upload it as a file
5. Click **Confirm** — the job with 4 tasks will be created:
   * bronze\_layer
   * silver\_layer
   * gold\_layer
   * export\_to\_snowflake

### Step 2: ****Create the Notebooks****

Create these four notebooks in the specified paths (adjust to your workspace):

#### **Notebook 1: /Repos/your\_email@example.com/campaign\_etl/bronze\_ingest**

python

df = spark.read.json("/mnt/campaign/raw/")

df.write.format("delta").mode("overwrite").saveAsTable("bronze.campaign\_raw")

#### **Notebook 2: silver\_transform**

python

df = spark.read.table("bronze.campaign\_raw") \

.filter("event\_type IN ('click', 'impression', 'conversion')") \

.filter("campaign\_id IS NOT NULL AND event\_time IS NOT NULL")

df.write.format("delta").mode("overwrite").saveAsTable("silver.campaign\_cleaned")

#### **Notebook 3: gold\_metrics**

python

from pyspark.sql.functions import count, when, col

df = spark.read.table("silver.campaign\_cleaned")

result = df.groupBy("campaign\_id").agg(

count(when(col("event\_type") == "impression", True)).alias("impressions"),

count(when(col("event\_type") == "click", True)).alias("clicks"),

count(when(col("event\_type") == "conversion", True)).alias("conversions")

).withColumn("CTR", col("clicks") / col("impressions")) \

.withColumn("CVR", col("conversions") / col("clicks"))

result.write.format("delta").mode("overwrite").saveAsTable("gold.campaign\_metrics")

#### **Notebook 4: export\_to\_snowflake**

python

SNOWFLAKE\_OPTIONS = {

"sfURL": "your\_account.snowflakecomputing.com",

"sfDatabase": "MARKETING",

"sfSchema": "ANALYTICS",

"sfWarehouse": "COMPUTE\_WH",

"sfRole": "SYSADMIN",

"sfUser": "SAK",

"sfPassword": "Arun@ch@l@"

}

df = spark.read.table("gold.campaign\_metrics")

df.write \

.format("snowflake") \

.options(\*\*SNOWFLAKE\_OPTIONS) \

.option("dbtable", "CAMPAIGN\_METRICS") \

.mode("overwrite") \

.save()

For secure handling of credentials, you can use **Databricks secrets** instead of hardcoding passwords.

**CAMPAIGN\_METRICS Table – Top 3 Rows**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **campaign\_id** | **event\_time** | **impressions** | **clicks** | **conversions** | **ctr** | **cvr** |
| CAMP001 | 2025-07-09 10:00:00 | 1200 | 100 | 25 | 0.083 | 0.25 |
| CAMP002 | 2025-07-09 10:00:00 | 950 | 40 | 5 | 0.042 | 0.125 |
| CAMP003 | 2025-07-09 10:00:00 | 600 | 30 | 10 | 0.050 | 0.333 |

### Step 3: ****Schedule or Trigger the Pipeline****

1. Go to **Workflows > Jobs**
2. Click on the **Delta to Snowflake Campaign ETL** job
3. Click **Edit** > **Schedule** tab
4. Set:
   * **Start time**
   * **Recurring (e.g., every hour, daily, etc.)**
5. Or click **Run Now** to test it

### Workflow in precise for understanding

* Bronze → Silver → Gold → Snowflake runs in order
* Each task reuses the same shared cluster
* You can monitor failures, retries, and durations in the Jobs UI

**Step 4 : Snowflake Query for KPI**

SELECT

m.campaign\_id,

cm.campaign\_name,

cm.owner,

DATE\_TRUNC('HOUR', m.event\_time) AS hour,

SUM(m.impressions) AS total\_impressions,

SUM(m.clicks) AS total\_clicks,

SUM(m.conversions) AS total\_conversions,

ROUND(SUM(m.clicks) \* 1.0 / NULLIF(SUM(m.impressions), 0), 4) AS ctr,

ROUND(SUM(m.conversions) \* 1.0 / NULLIF(SUM(m.clicks), 0), 4) AS cvr,

**-- Status flag based on thresholds**

CASE

WHEN SUM(m.clicks) \* 1.0 / NULLIF(SUM(m.impressions), 0) < 0.03 THEN 'Low CTR'

WHEN SUM(m.conversions) \* 1.0 / NULLIF(SUM(m.clicks), 0) < 0.20 THEN 'Low CVR'

ELSE 'Healthy'

END AS campaign\_status

FROM CAMPAIGN\_METRICS m

JOIN CAMPAIGN\_METADATA cm

ON m.campaign\_id = cm.campaign\_id

GROUP BY m.campaign\_id, cm.campaign\_name, cm.owner, DATE\_TRUNC('HOUR', m.event\_time)

ORDER BY hour DESC, ctr ASC;

**Output Example**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **campaign\_id** | **name** | **owner** | **hour** | **ctr** | **cvr** | **status** |
| CAMP001 | SummerPush | A. Patel | 2025-07-09 12:00:00 | 0.025 | 0.18 | Low CTR |
| CAMP002 | SpringBuzz | J. Singh | 2025-07-09 12:00:00 | 0.045 | 0.15 | Low CVR |
| CAMP003 | Sri Sivakumar | M. Wong | 2025-07-09 12:00:00 | 0.055 | 0.30 | Healthy |

Final Note:

Moving data from **Delta Lake to Snowflake** combines the best of both worlds—Delta handles real-time ingestion, schema enforcement, and ACID-compliant updates at scale, while Snowflake excels at high-performance analytics, BI reporting, and cross-team collaboration.

This integration enables teams to process raw data efficiently in Databricks, then deliver clean, trusted datasets to Snowflake for downstream insights, dashboards, and decision-making—all in a unified, cost-effective pipeline.

Top brands like **Adobe** use a Delta Lake to Snowflake pipeline to combine real-time data ingestion with powerful cloud analytics. Delta Lake handles large-scale, streaming data transformations, while Snowflake provides scalable, secure analytics and BI reporting. This architecture ensures fast, reliable insights across teams with clean, trusted data.