

Delta Live Tables Hands-on Workshop

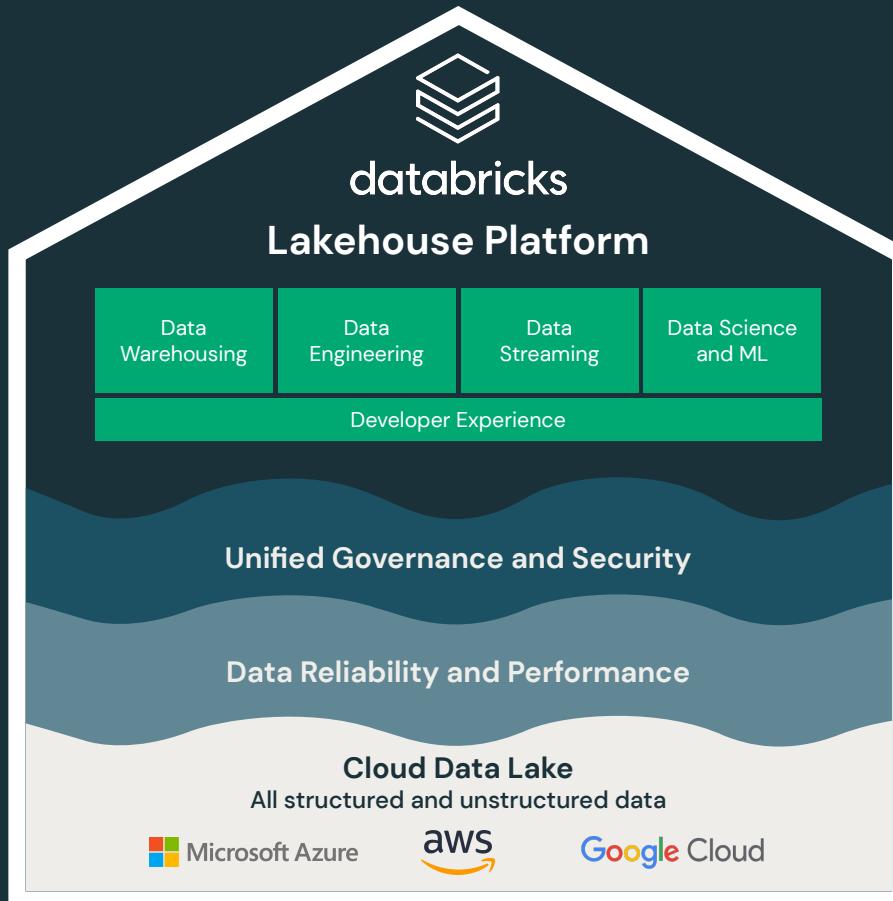
Glenn Wiebe, Sr. Solution Architect

Alex Desroche, Sr. Customer Success Engineer

2022-06-16

Agenda

- Delta Live Tables within Databricks Lakehouse
- DLT Workshop
 - DB SQL & Data Profiling, Simplified Ingestion
 - Pipeline 1 – Sales “Channel” Reference Data Load
 - Simple → Medium → Enhanced DLT definition & iteration
 - Data lineage using built-in functions & capabilities
 - Transition from reference (as is) to master (governed) data loading
 - Pipeline 2 – “Customer” Master Data Load
 - Joins, External Data, Views & Advanced CDC
 - Troubleshooting
 - Event Log – Queries & Dashboards
 - Workflow – MTJ: SalesChannel → Customer Load
- Q&A



Databricks Lakehouse Platform

Simple

Unify your data warehousing and AI use cases on a single platform

Open

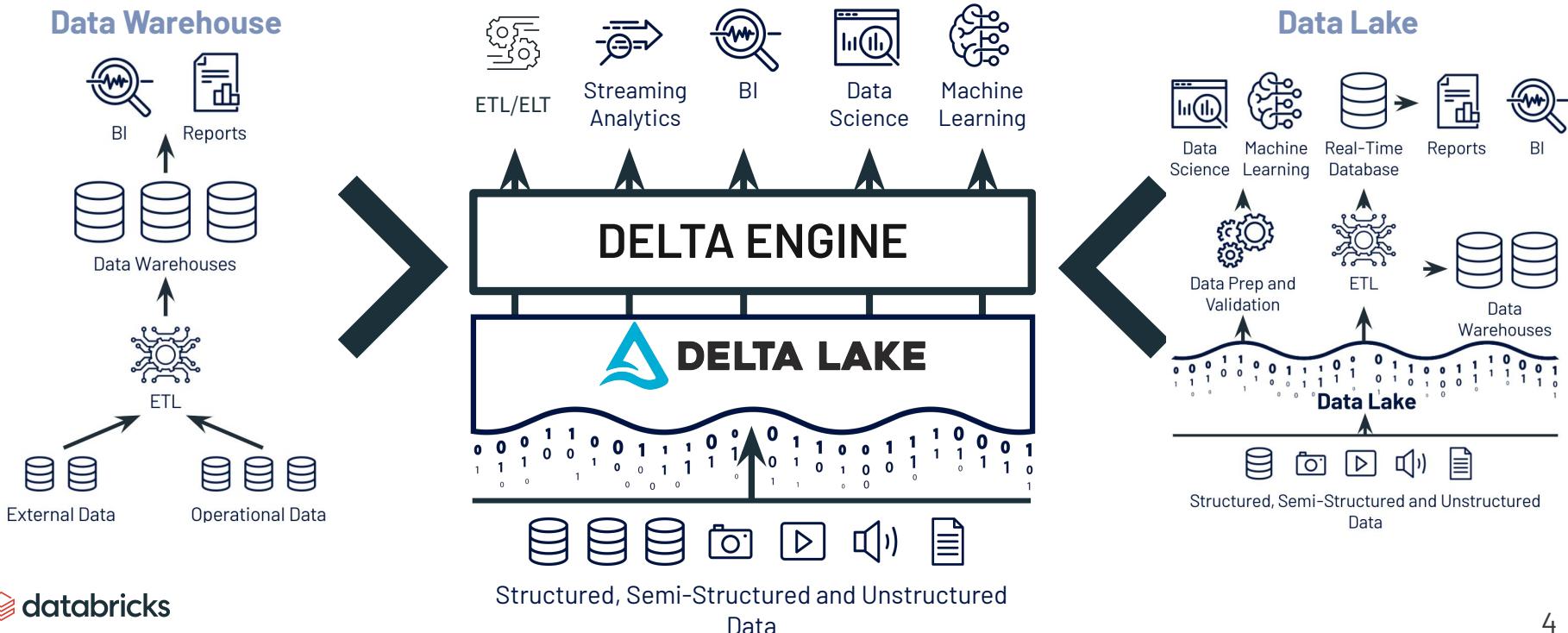
Built on open source and open standards

Multicloud

One consistent data platform across clouds

Delivered by the Lakehouse Architecture

The best of Data Lakes and Data Warehouses

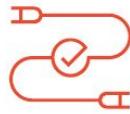


Delta Live Tables

Delta Live Tables

Push-to-start Enterprise Pipelines

Fully Incremental
Data & Aggregates



Auto-scaling
Infrastructure



Enterprise Data
Lineage



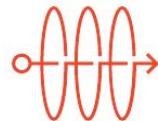
Fine Grained
Access Controls



Data Quality
Enforcements



Configurable
Pipelines



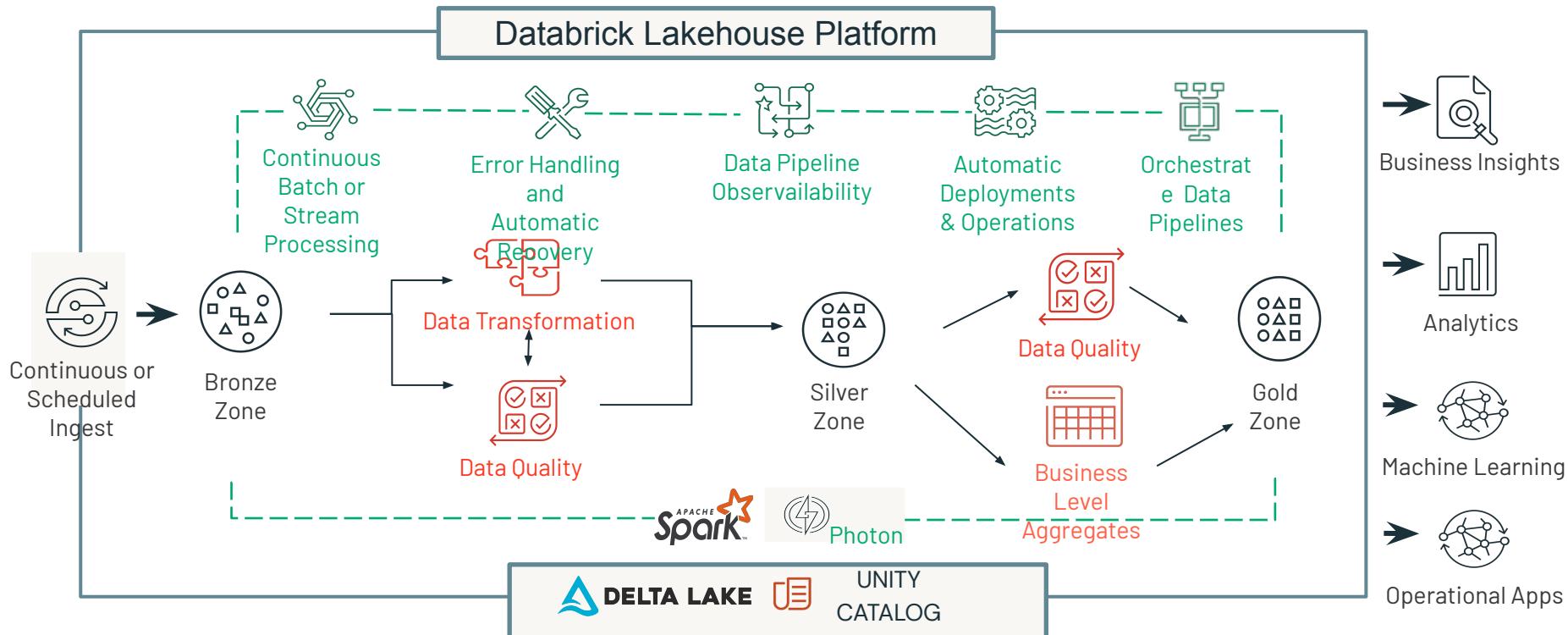
Easy Logging +
Monitoring



Enterprise Data
Catalog



Build Production ETL Pipelines with DLT



Continuous or scheduled data ingestion

Simple SQL Syntax for Streaming Data Ingestion

```
Cmd 4
1 CREATE INCREMENTAL LIVE TABLE sales_orders_raw
2 COMMENT "The raw sales orders, ingested from /databricks-datasets."
3 TBLPROPERTIES ("quality" = "bronze")
4 AS
5 SELECT * FROM cloud_files
6 ("databricks-datasets/retail-org/sales_orders/",
7 "json", map("cloudFiles.inferColumnTypes", "true"));
```

counts (id: ef91bf99-97fd-433a-bfc5-5a5bdccbef4) Last updated: 5 seconds ago

Dashboard Raw Data

Input vs. Processing Rate: 570.6 rec/s Input rate, 673.4 rec/s Processing rate

Batch Duration in seconds: 4.2 s Average, 3 s Latest

Aggregation State: 66 Distinct keys

- **Incrementally and efficiently** process new data files as they arrive in cloud storage using Auto Loader
- Automatically **infer schema** of incoming files or superimpose what you know with **Schema Hints**
- Automatic **schema evolution**
- **Rescue data column** – no data loss again

Schema Evolution



JSON



CSV



AVRO

Coming Soon
PARQUET

Declarative SQL & Python APIs

Source

```
/* Create a temp view on the accounts table */  
CREATE STREAMING LIVE VIEW account_raw AS  
SELECT * FROM cloud_files("/data", "csv");
```

Bronze

```
/* Stage 1: Bronze Table drop invalid rows */  
CREATE STREAMING LIVE TABLE account_bronze AS  
COMMENT "Bronze table with valid account ids"  
SELECT * FROM fire_account_raw ...
```

Silver

```
/* Stage 2:Send rows to Silver, run validation  
rules */  
CREATE STREAMING LIVE TABLE account_silver AS  
COMMENT "Silver Accounts table with validation  
checks"  
SELECT * FROM fire_account_bronze ...
```

Gold

- Use intent-driven declarative development to abstract away the **“how”** and define **“what”** to solve

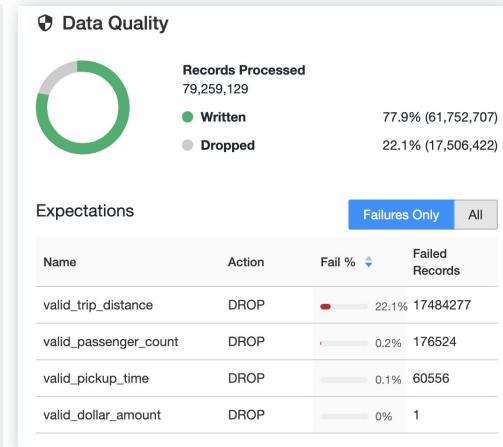
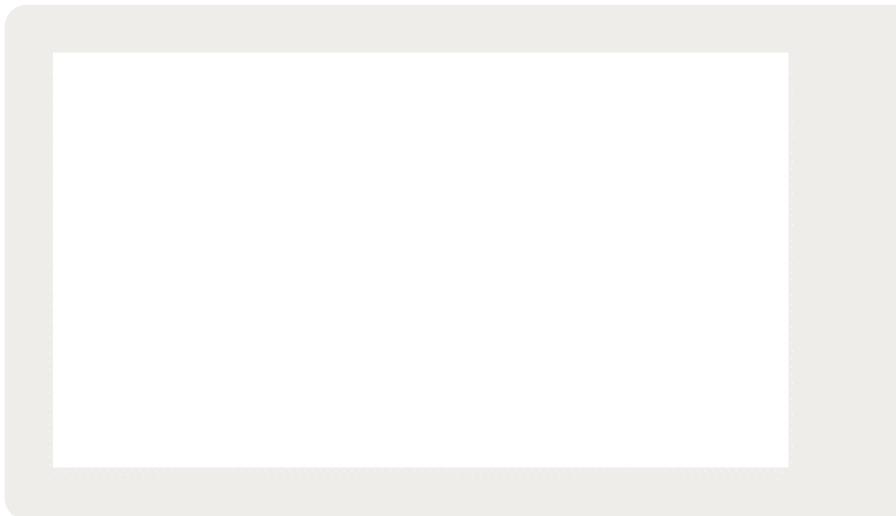
- Automatically generate **lineage** based on table dependencies across the data pipeline

- Automatically checks for errors, missing dependencies and syntax errors

Data quality validation and monitoring

Load and transform at any scale with high quality data pipelines

- **Define data quality & integrity controls** within the pipeline with data expectations
- **Address data quality errors** with flexible policies: fail, drop, alert, quarantine(future)
- All data pipeline runs & quality metrics are captured, tracked and reported



Data pipeline observability

The screenshot displays the Databricks Delta Live Tables SQL Pipeline interface. It includes:

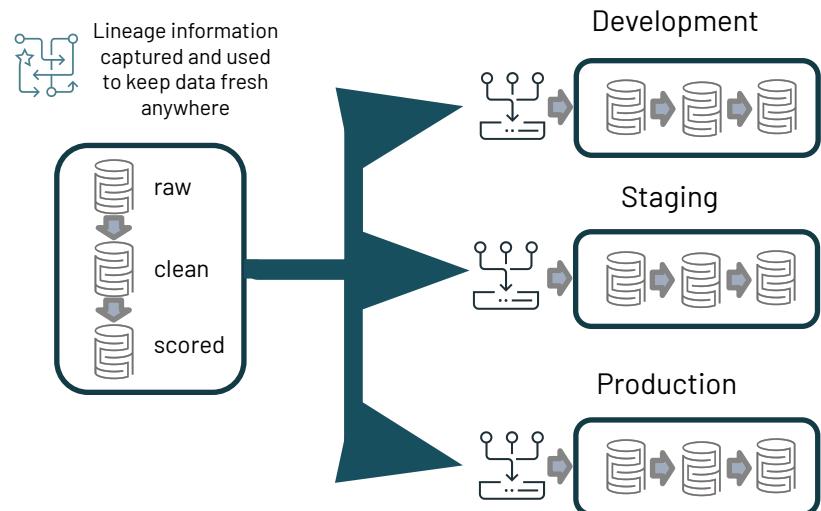
- Lineage Diagram:** Shows the flow of data from various sources (e.g., v_silver_yellow_taxi, v_silver_green_taxi) through transformations (e.g., v_silver_taxi, v_silver_taxi_rate) to final gold tables (e.g., tlr_gold_taxi_for_analytics, tlr_gold_taxi_for_payments).
- Pipeline Event Log Details:** A modal showing a log entry for a specific row, including fields like total_amount, payment_type, pickup_datetime, and pickup_time.
- Data Quality:** A circular progress bar indicating 77.9% (81,752,707) of data is written, with 22.1% (17,506,462) dropped.
- Expectations:** A table showing the status of various expectations across different data sets.
- Data Quality Metrics:** A dashboard with three charts: Data Quality Per Expectation, Data Quality Per Data Set, and % Failed Records.
- Taxi Demo:** A section showing flow progress and update progress for various pipeline stages.
- Taxi Cab Analysis:** Two line charts showing Average Speed per Trip/Hour and Average Distance per Trip/Hour.

- High-quality, high-fidelity lineage diagram that provides visibility into how data flows for impact analysis
- Granular logging for operational, governance, quality and status of the data pipeline at a row level
- Continuously monitor data pipeline jobs to ensure continued operation
- Notifications using Databricks SQL

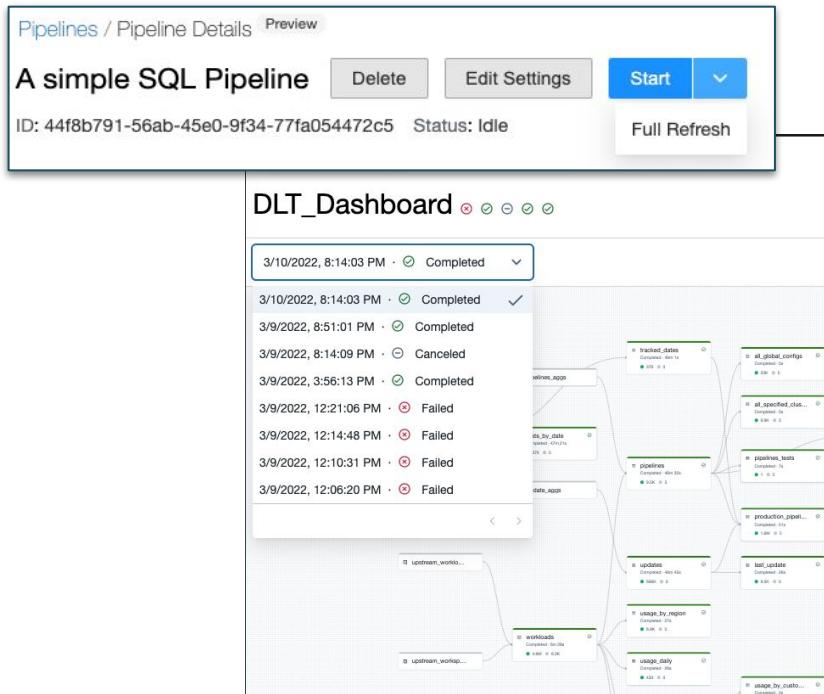


Automated ETL development lifecycle

- Develop in environment(s) separate from production with the ability to easily test it before deploying - entirely in SQL
- Deploy and manage environments using parameterization
- Unit testing and documentation
- Enables metadata-driven ability to programmatically scale to 100s of tables/pipelines dynamically

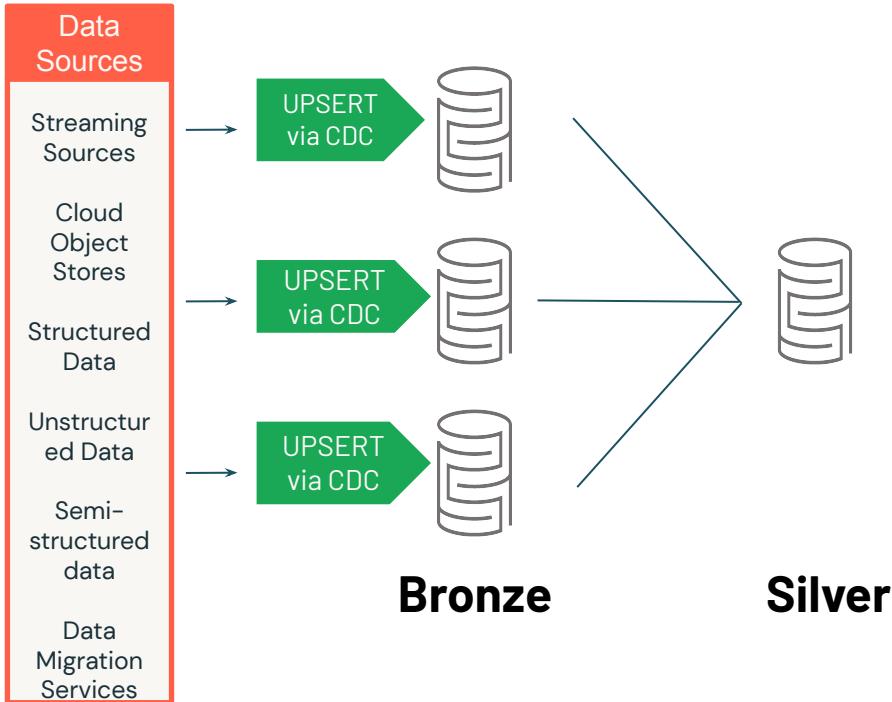


Automated ETL operations



- Reduce down time with automatic error handling and easy replay
- Eliminate maintenance with automatic optimizations of all Delta Live Tables
- **Auto-scaling** adds more resources automatically when needed.

Change data capture (CDC)



- Stream change records (inserts, updates, deletes) from any data source supported by DBR, cloud storage, or DBFS
- Simple, declarative “APPLY CHANGES INTO” API for SQL or Python
- Handles out-of-order events
- Schema evolution

Getting started with Change Data Capture in DLT

Enable **DLT APPLY CHANGES INTO** in **DLT Pipeline**

```
"configuration": {  
  
  "pipelines.applyChangesPreviewEnabled":  
    "true"  
}
```

Create Pipeline X

* Pipeline Name UI JSON

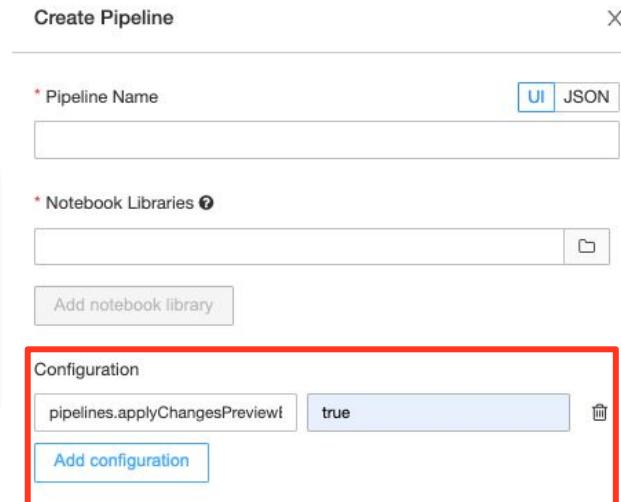
* Notebook Libraries ? []

Add notebook library

Configuration

pipelines.applyChangesPreviewEnabled true []

Add configuration



Getting started with Change Data Capture in DLT

Delete unwanted clients records - Silver Table - DLT SQL

```
1 CREATE INCREMENTAL LIVE TABLE retail_client_cdc_silver;
2
3 APPLY CHANGES INTO live.retail_client_cdc_silver          APPLY CHANGES INTO to propagate inserts, updates and deletes to your target table.
4 FROM stream(live.retail_client_cdc_raw)
5 KEYS (Id)
6 APPLY AS DELETE WHEN operation = "DELETE"               Specifies when a CDC event should be treated as a DELETE rather than an upsert.
7 SEQUENCE BY operation_date --primary key, auto-incrementing ID of any kind that can be used to identify order of events, or timestamp
8 COLUMNS * EXCEPT (operation, operation_date);           To remove columns needed for APPLY CHANGES INTO computation, but not needed in final table state.
```



“Delta Live Tables has helped our teams save time and effort in managing data at this scale. With this capability augmenting the existing lakehouse architecture, Databricks is **disrupting the ETL and data warehouse markets**, which is important for companies like ours. We are excited to continue to work with Databricks as an innovation partner.” – Dan Jeavons, GM Data Science

Use Case + Challenge

- 70+ use cases impacting supply chain, operations, product development, marketing, customer exp
- Large volumes of IoT data from millions of sensors difficult to harness for actionable insights and ML due to operational load created by complex data pipelines

Why Databricks + DLT?

- Lakehouse for unified data warehousing, BI, & ML — enabling new use cases not possible before
- DLT enables Shell to build reliable and scalable data pipelines – automatic job maintenance and deep pipeline visibility saves time and resources

Impact of DLT

- Process **1.3 trillion rows of sensor data** with ease
- Simplifying ETL development and management for **faster insights and ML innovation**

Agenda

- Delta Live Tables within Databricks Lakehouse
- DLT Workshop
 - DB SQL & Data Profiling, Simplified Ingestion
 - Pipeline 1 – Sales “Channel” Reference Data Load
 - Simple → Medium → Enhanced DLT definition & iteration
 - Data lineage using built-in functions & capabilities
 - Transition from reference (as is) to master (governed) data loading
 - Pipeline 2 – “Customer” Master Data Load
 - Joins, External Data, Views & Advanced CDC
 - Troubleshooting
 - Event Log – Queries & Dashboards
 - Workflow – MTJ: SalesChannel → Customer Load
- Q&A

Reference Materials/Links

Delta Live Tables is a framework for building reliable, maintainable, and testable data processing pipelines. You define the transformations to perform on your data, and Delta Live Tables manages task orchestration, cluster management, monitoring, data quality, and error handling.

Instead of defining your data pipelines using a series of separate Apache Spark tasks, Delta Live Tables manages how your data is transformed based on a target schema you define for each processing step. You can also enforce data quality with Delta Live Tables *expectations*. Expectations allow you to define expected data quality and specify how to handle records that fail those expectations.

To get started with Delta Live Tables:

- Develop your first Delta Live Tables pipeline with the [quickstart](#).
- Learn about fundamental Delta Live Tables [concepts](#).
- Learn how to create, run, and manage pipelines with the Delta Live Tables [user interface](#).
- Learn how to develop Delta Live Tables pipelines with [Python](#) or [SQL](#).
- Learn how to manage data quality in your Delta Live Tables pipelines with [expectations](#).

Learn more about Delta Live Tables:

- Use external data sources in your Delta Live Tables pipelines: [Data sources](#)
- Use the data produced by your Delta Live Tables pipelines: [Publish data](#)
- Efficiently process continually arriving data in your Delta Live Tables pipelines: [Streaming data processing](#)
- Use change data capture (CDC) processing in your Delta Live Tables pipelines: [Change data capture with Delta Live Tables](#)
- Use the Delta Live Tables API: [API guide](#)
- Configure your Delta Live Tables pipelines: [Pipeline settings](#)
- Analyze and report on your Delta Live Tables pipelines: [Querying the event log](#)
- Run your Delta Live Tables pipelines with popular workflow orchestration tools: [Workflow tool integration](#)
- Learn how to use access control lists (ACLs) to configure permissions on your Delta Live Tables pipelines: [Access control](#)

Find answers and solutions for Delta Live Tables:

- Implement common tasks in your Delta Live Tables pipelines: [Cookbook](#)
- Review frequently asked questions and issues: [FAQ](#)

Delta Live Tables is a framework for building reliable, maintainable, and testable data processing pipelines. You define the transformations to perform on your data, and Delta Live Tables manages task orchestration, cluster management, monitoring, data quality, and error handling.

Instead of defining your data pipelines using a series of separate Apache Spark tasks, Delta Live Tables manages how your data is transformed based on a target schema you define for each processing step. You can also enforce data quality with Delta Live Tables *expectations*. Expectations allow you to define expected data quality and specify how to handle records that fail those expectations.

To get started with Delta Live Tables:

- Develop your first Delta Live Tables pipeline with the [quickstart](#).
- Learn about fundamental Delta Live Tables [concepts](#).
- Learn how to create, run, and manage pipelines with the Delta Live Tables [user interface](#).
- Learn how to develop Delta Live Tables pipelines with [Python](#) or [SQL](#).
- Learn how to manage data quality in your Delta Live Tables pipelines with [expectations](#).

Learn more about Delta Live Tables:

- Use external data sources in your Delta Live Tables pipelines: [Data sources](#)
- Use the data produced by your Delta Live Tables pipelines: [Publish data](#)
- Efficiently process continually arriving data in your Delta Live Tables pipelines: [Streaming data processing](#)
- Use change data capture (CDC) processing in your Delta Live Tables pipelines: [Change data capture with Delta Live Tables](#)
- Use the Delta Live Tables API: [API guide](#)
- Configure your Delta Live Tables pipelines: [Pipeline settings](#)
- Analyze and report on your Delta Live Tables pipelines: [Querying the event log](#)
- Run your Delta Live Tables pipelines with popular workflow orchestration tools: [Workflow tool integration](#)
- Learn how to use access control lists (ACLs) to configure permissions on your Delta Live Tables pipelines: [Access control](#)

Find answers and solutions for Delta Live Tables:

- Implement common tasks in your Delta Live Tables pipelines: [Cookbook](#)
- Review frequently asked questions and issues: [FAQ](#)

Questions?

Q&A and Post-workshop Discussion

Thank you for joining us. We're here to answer any questions you have about the content, Databricks, or anything else. We'll send out a copy of this presentation after we wrap up.

Workshop Git Repo: https://github.com/ggwiebe/db_dlt_workshop.git

Let us know what you liked and didn't like so that we can continue to improve.

If you're not currently in contact with your Databricks account team and would like to be, let the presenters know and we can assist.

Contact info for presenters

glenn.wiebe@databricks.com , alex.desroches@databricks.com

Workshop Screenshots

DLT Lab Screenshots - Ingest Data - Explorer Wizard

The screenshot illustrates the Databricks Data Explorer interface, specifically the 'Create table in Data' wizard, used for ingesting data from a CSV file.

Left Panel (Data Explorer):

- Shows the 'Data' catalog.
- Selected: **ggw_retail_sandbox** (No data).
- Other entries include: ggw_retail, ggw_retailorg, ggw_sales, ggw_sales_raw, ggw_schema, ggw_telco, ggw_template, ggw_tpcds1tb, ggw_wine.

Middle Panel (Create table in Data):

- Shows the 'Create table via upload' and 'Ingest via partner' options.
- File selected: **channels.csv**.
- Table schema preview:
 - 1 "channelId"
 - 2 1, RETAIL
 - 3 2, WEB, C
 - 4 3, PARTNER, Customer referred from Partners

Right Panel (Catalogs > hive_metastore > ggw_retail_sandbox):

- Shows the **ggw_retail_sandbox.channels** table.
- Comment: Created by the file upload UI.
- Owner: glenn.wiebe@databricks.com.
- Size: 1.3KB, 1 file.
- Schema (selected):

channelId	channelName	description
1	RETAIL	Customer originated from Retail Stores
2	WEB	Customer originated from Online properties
3	PARTNER	Customer referred from Partners
9	OTHER	Unattributed customer origination
- Sample Data, Details, Permissions, History tabs.

Bottom Panel (Create table in Databricks SQL Preview):

- Shows the 'channels.csv 221.00B' file.
- Target table: **ggw_retail_sandbox.channels**.
- Advanced attributes: First row contains the header.
- Previewing 4 rows, 3 columns:

channelId	channelName	description
1	RETAIL	Customer originated from Retail Stores
2	WEB	Customer originated from Online properties
3	PARTNER	Customer referred from Partners
9	OTHER	Unattributed customer origination

DLT Lab Screenshots - Ingest Data - Notebook Wizard

The screenshot shows the Databricks Notebook Wizard interface. On the left, the 'Create New Table' step is displayed, where a CSV file named 'channels.csv' is uploaded to the DBFS target directory. The table attributes are specified: Table Name is 'channels_csv', File Type is 'CSV', and the first row is identified as the header. On the right, the resulting notebook output is shown, titled '2022-06-15 - DBFS Example'. The notebook contains Python code to read the CSV file, register it as a temporary view, and then create a permanent table named 'channels_csv' in the 'ggw_retail_sandbox' database. The resulting data is displayed in a table:

channelId	channelName	description
1	RETAIL	Customer originated from Retail Stores
2	WEB	Customer originated from Online properties
3	PARTNER	Customer referred from Partners
4	OTHER	Unattributed customer origination

Below the table, the notebook continues with more code to query the permanent table and save it as a DataFrame.

DLT Lab Screenshots - Ingest Data - Notebook Wizard #2

Git 2022-06-15 - DBFS Example - Register CSV Table Python

Schedule

Overview

This notebook shows how a CSV file can be the storage for a table in spark/DB SQL.
You simply are registering a table (via CREATE TABLE) against a different storage type (with the USING [type] syntax) and the unmanaged storage path.

Cmd 2

```
# Recall a standard spark read looks like this: # File location and type file_l ...
Show cell
```

Cmd 3

```
1 %sql
2 -- We can register a table against the csv file using the USING clause to set the storage type
3
4 -- mode "FAILFAST" will abort file parsing with a RuntimeException if any malformed lines are encountered
5 CREATE TABLE ggw_retail_sandbox.channels_csv_table
6   USING CSV
7   OPTIONS (path "/FileStore/tables/channels.csv", header "true", mode "FAILFAST")
8 ;
```

OK

Command took 0.79 seconds -- by glenn.wiebe@databricks.com at 6/15/2022, 12:43:56 PM on unknown cluster

Cmd 4

```
1 %sql
2 SELECT *
3   FROM ggw_retail_sandbox.channels_csv_table
4 ;
```

Table Data Profile

	channelId	channelName	description
1	1	RETAIL	Customer originated from Retail Stores
2	2	WEB	Customer originated from Online properties
3	3	PARTNER	Customer referred from Partners
4	9	OTHER	Unattributed customer origination

Showing all 4 rows.

Command took 0.31 seconds -- by glenn.wiebe@databricks.com at 6/15/2022, 12:44:30 PM on unknown cluster

Catalogs > hive_metastore > ggw_retail_sandbox >

hive_metastore.ggw_retail_sandbox.channels_csv_table CSV

Comment: [Comment](#)

Owner: glenn.wiebe@databricks.com [Owner](#)

Size: Unknown

Schema Sample Data Details Permissions History

Created Time	Wed Jun 15 16:43:57 UTC 2022
Last Access	UNKNOWN
Created By	Spark 3.2.1
Type	EXTERNAL
Location	dbfs:/FileStore/tables/channels.csv
Serde Library	org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
Input Format	org.apache.hadoop.mapred.SequenceFileInputFormat
Output Format	org.apache.hadoop.hive.ql.io.HiveSequenceFileOutputFormat
Storage Properties	mode=FAILFAST header=true

DLT Lab Screenshots - Build DLT Pipeline - Query Source

The screenshot shows a DLT pipeline configuration interface with the following details:

- Schema browser:** Shows tables in the `hive_metastore > ggw_retail_sandbox` database, including `channels` and `channels_csv`.
- Shared Endpoint - Photon (L):** A table with 1 row, showing the query used to read the CSV file directly with DB SQL.
- Code Editor:** The query used to read the CSV file directly with DB SQL, including the creation of a temporary table and the use of the `OPTIONS` clause.
- Run Statement:** `SHOW CREATE TABLE ggw_retail_sandbox.channels_csv_table;`
- Table View:** A preview of the `channels` table with 9 rows of data.
- Table Definition:** The `createtab_stmt` for the `channels_csv_table`, which is a `CREATE TABLE` statement using CSV options to ingest data from `dbfs:/FileStore/tables/channels.csv`.

DLT Lab Screenshots - Create DLT Pipeline - Reference

```

1  "channelId","channelName","description"
2  1,RETAIL,Customer originated from Retail
3  2,WEB,Customer originated from Online properties
4  3,PARTNER,Customer referred from Partners
5  9,OTHER,Unattributed customer origination

```

Step 0 - Get Table info from selected table

-- This is the syntax from the CSV Table
-- We will not run this, but it serves to direct our table name

```

CREATE TABLE ggw_retail_sandbox.channels_csv_table (
  channelId INT,
  channelName STRING,
  description STRING)
  USING CSV
OPTIONS ( 'header' = 'true', 'inferSchema' = 'true'
LOCATION 'dbfs:/FileStore/tables/ggw_dlt_wshp/channel'

```

Cmd 4

Step 1-a - simplest query

Cmd 5

```

1  CREATE STREAMING LIVE TABLE channel
2  TBLPROPERTIES ("quality" = "reference")
3  COMMENT "Channel Reference dataset ingested from cloud"
4  AS
5  SELECT *
6  FROM cloud_files('/FileStore/tables/ggw_dlt_wshp/
7  ;

```

The screenshot shows the Databricks interface with several windows open:

- Workflows** (Left): Shows a list of workflows including "GGW Customer Materials", "GGW Department", "GGW DLT Wkshp - Reference", "GGW Ingest2Gold", "GGW Loan Risk", "GGW Loan Risk Part 2", "GGW Loans", "GGW Ordered PII", "GGW Retail Customers", and "GGW Retail Reference".
- Create Pipeline** (Top Center): A modal window titled "Create pipeline" with sections for "Product edition" (Advanced), "Pipeline name" (set to "GGW DLT WShp - Reference"), "Notebook libraries" (a link to a notebook), "Configuration" (Add configuration), and "Target" (set to "ggw_retail_wshp").
- GGW DLT WShp - Reference** (Main View): A pipeline details page for "GGW DLT WShp - Reference". It shows the pipeline was completed on 6/16/2022 at 12:55:50 PM. The pipeline details include:
 - channel** (Table): Path: /Repos/glenn.wiebe@databricks.com/db_dlt_workshop/Not ebooks/ChannelsReference-Live
 - Schema**: channel (int), channelName (string), description (string), _rescued_data (string)
 - Data quality**: 100% (4) Written, 0% (0) Dropped
- Schema browser** (Bottom Left): Shows the schema for the "channel" table:

channel	channelId	channelName	description	_rescued_data
1	RETAIL	Customer originated from Retail Stores	NULL	
2	WEB	Customer originated from Online properties	NULL	
3	PARTNER	Customer referred from Partners	NULL	
9	OTHER	Unattributed customer origination	NULL	
- Table** (Bottom Right): A table view of the "channel" data with the same structure and rows as the schema browser.

DLT Lab Screenshots - DLT Master with SCD Tracking

```
31 SELECT *
32   FROM ggw_retail_wshp.channel_master
33 ;
```

{} Table LIMIT 1000

Table

channelId	channelName	description	input_file_name	dt_ingest_dt	dt_ingest_procedure
1	RETAIL	Customer originated from Retail Stores	/FileStore/tables/ggw_dlt_wshp/channels.csv	2022-06-16 16:29:57.125	RetailReference_Live
2	WEB	Customer originated from Online properties	/FileStore/tables/ggw_dlt_wshp/channels.csv	2022-06-16 16:29:57.125	RetailReference_Live
3	PARTNER	Customer referred from Partners	/FileStore/tables/ggw_dlt_wshp/channels.csv	2022-06-16 16:29:57.125	RetailReference_Live
9	OTHER	Unattributed customer origination	/FileStore/tables/ggw_dlt_wshp/channels.csv	2022-06-16 16:29:57.125	RetailReference_Live

Step 4 - Add SCD Type Tracking to a Silver Master Channel

Cmd 11

```
1 -- SILVER - View against Bronze that will be used to load silver incrementally with APPEND
2 -- CREATE TEMPORARY [STREAMING] LIVE VIEW view_name
3 CREATE TEMPORARY STREAMING LIVE VIEW channel_silver_v (
4   CONSTRAINT valid_file      EXPECT (input_file_name IS NOT NULL) ON VIOLATION DROP
5   CONSTRAINT valid_procedure EXPECT (dt_ingest_procedure IS NOT NULL) ON VIOLATION
6 )
7 COMMENT "View of cleansed Channel (bronze tier) for loading into / mastering in Silver"
8 AS SELECT channelId,
9       channelName,
10      description,
11      input_file_name,
12      dt_ingest_dt,
13      dt_ingest_procedure,
14      dt_ingest_principal
15      FROM STREAM(LIVE.channel) c
```

Cmd 12

```
1 CREATE STREAMING LIVE TABLE channel_master;
2
3 APPLY CHANGES INTO LIVE.channel_master
4   FROM STREAM(LIVE.channel_silver_v)
5   KEYS (channelId)
6   SEQUENCE BY dt_ingest_dt
7   STORED AS SCD TYPE 2
8 ;
```

```
1 SELECT *
2   FROM ggw_retail_wshp.channel_master
3 ;
```

{} Table LIMIT 1000

Table

channelId	channelName	description	input_file_name	dt_ingest_dt	dt_ingest_procedure	dt_ingest_principal	START_AT	END_AT
1	RETAIL	Customer originated from Retail Stores	/FileStore/tables/ggw_dlt_wshp/channels.csv	2022-06-16 16:29:57.125	RetailReference_Live	glenn.wiebe@datricks.com	2022-06-16 16:29:57.125	NULL
2	WEB	Customer originated from Online properties	/FileStore/tables/ggw_dlt_wshp/channels.csv	2022-06-16 16:29:57.125	RetailReference_Live	glenn.wiebe@datricks.com	2022-06-16 16:29:57.125	NULL
3	PARTNER	Customer originated OR referred from Partners	/FileStore/tables/ggw_dlt_wshp/channels.csv	2022-06-16 16:29:57.125	RetailReference_Live	glenn.wiebe@datricks.com	2022-06-16 16:29:57.125	2022-06-16 16:36:17.685
3	PARTNER	Customer originated OR referred from Partners	/FileStore/tables/ggw_dlt_wshp/channels_update1.csv	2022-06-16 16:36:17.685	RetailReference_Live	glenn.wiebe@datricks.com	2022-06-16 16:36:17.685	NULL
4	UNAFFILIATED	Customers that request anonymity	/FileStore/tables/ggw_dlt_wshp/channels_update1.csv	2022-06-16 16:36:17.685	RetailReference_Live	glenn.wiebe@datricks.com	2022-06-16 16:36:17.685	NULL
9	OTHER	Unattributed customer origination	/FileStore/tables/ggw_dlt_wshp/channels.csv	2022-06-16 16:29:57.125	RetailReference_Live	glenn.wiebe@datricks.com	2022-06-16 16:29:57.125	2022-06-16 16:36:17.685
9	OTHER	Unattributed or unknown customer origination	/FileStore/tables/ggw_dlt_wshp/channels_update1.csv	2022-06-16 16:36:17.685	RetailReference_Live	glenn.wiebe@datricks.com	2022-06-16 16:36:17.685	NULL

Workflows / Delta Live Tables / Pipeline details

GGW DLT WShp - Reference

0/16/2022, 1:24:49 PM - Completed

Development Production Delete Permissions Settings Schedule Start

Metastore ggw_retail_wshp.channel

Status Completed

Start time 0/16/2022, 1:24:56 PM

Duration 6s

Comment Channel Reference dataset ingested from cloud object storage landing zone

Schema

```
channelId: integer
channelName: string
description: string
input_file_name: string
dt_ingest_dt: timestamp
dt_ingest_procedure: string
dt_ingest_principal: string
```

Data quality

Written 100% (4)

Dropped 0% (0)

Expectations

Name	Action	Fail %	Failed records
valid_channel_name	DROP	0%	0

All Filter...

52 seconds ago flow_progress Flow 'channel' has COMPLETED.

51 seconds ago flow_progress Flow 'channel' is STARTING.

50 seconds ago flow_progress Flow 'channel' is RUNNING.

47 seconds ago flow_progress Flow 'channel_master' has COMPLETED.

45 seconds ago update_progress Update 56a310 is COMPLETED.

DLT Lab Screenshots - DLT Pipeline - Customer Master

customer-1-insert.csv

```

1 "id","first_name","last_name","email","channel","active"
2 1001,Glenn,Wiebe,ggwiebe@gmail.com,1,1,9999-12-31,2022-01-01
3 1002,Graeme,Wiebe,glenn@wiebes.net,2,1,9999-12-31,2022-01-01

```

Workflows

Jobs Job runs Delta Live Tables

Create Pipeline

Workflows / Delta Live Tables / Pipeline details

GGW DLT WShp - Customer Master

6/16/2022, 2:19:13 PM | Completed

GGW DLT WShp - Customer Master

Development Production Delete Permissions Settings Schedule Stop

Workflows / Delta Live Tables / Pipeline details

GGW DLT WShp - Customer Master

6/16/2022, 2:19:13 PM | Completed

GGW DLT WShp - Customer Master

Development Production Delete Permissions Settings Schedule Start

Step 0 - Get sandbox info

```

CREATE TABLE ggw_retail_sandbox.customer_csv_table (
  id INT,
  first_name STRING,
  last_name STRING,
  email STRING,
  channel INT,
  active INT,
  active_end_date TIMESTAMP,
  update_dt TIMESTAMP,
  update_user STRING)
  USING CSV OPTIONS ('header' = 'true', 'inferSchema' = 'true', 'mode' = 'LOCATION 'dbfs:/FileStore/tables/ggw_dlt_wshp/customer*.csv'

```

Cmd 4

1. BRONZE - Land Raw Data and standardize type

Common Storage Format: Delta

Data Types: Cast & check Nulls

Cmd 5

```

1 -- BRONZE - Cloudfiles AutoLoader reads raw streaming files for "new"
2 CREATE OR REFRESH STREAMING LIVE TABLE customer_bronze
3 TBLPROPERTIES ("quality" = "bronze")
4 COMMENT "New customer data incrementally ingested from cloud object store"
5 AS
6 SELECT *,
7   input_file_name() input_file_name
8 FROM cloud_files('/FileStore/tables/ggw_dlt_wshp/customer*.csv', 'c
9 ;

```

Schema browser Past executions

ggw_dlt_workshop Shared Endpoint - Photon (L) Revert Share Save* Run Selected

hive_metastore > ggw_retail_ws...

Filter tables & columns...

apply_changes_storage_chann...

Refresh schedule Never

channel

Add visualization

channel_master

Table

customer_bronze

id first_name last_name email channel active active_end_date

id INT

1001 Glenn Wiebe ggwiebe@gmail.com 1 1 9999-12-31 00:00:00.000

first_name STRING

1002 Graeme Wiebe glenn@wiebes.net 2 1 9999-12-31 00:00:00.000

last_name STRING

1001 Glenn Wiebe ggwiebe@gmail.com 1 1 9999-12-31 00:00:00.000

email STRING

1002 Graeme Wiebe glenn@wiebes.net 2 1 9999-12-31 00:00:00.000

channel INT

1001 Glenn Wiebe ggwiebe@gmail.com 1 1 9999-12-31 00:00:00.000

active INT

1001 Glenn Wiebe ggwiebe@gmail.com 1 1 9999-12-31 00:00:00.000

active_end_date TIMESTAMP

1001 Glenn Wiebe ggwiebe@gmail.com 1 1 9999-12-31 00:00:00.000

update_dt TIMESTAMP

1001 Glenn Wiebe ggwiebe@gmail.com 1 1 9999-12-31 00:00:00.000

update_user STRING

1001 Glenn Wiebe ggwiebe@gmail.com 1 1 9999-12-31 00:00:00.000

_rescued_data STRING

1001 Glenn Wiebe ggwiebe@gmail.com 1 1 9999-12-31 00:00:00.000

input_file_name STRING

1001 Glenn Wiebe ggwiebe@gmail.com 1 1 9999-12-31 00:00:00.000

databricks

DLT Lab Screenshots - DLT Pipeline - Customer Master -

2

customer-1-insert.csv

```
1 "id","first_name","last_name","email","channel","active","active_end_date","update_dt","update_user"
2 1001,Glenn,Wiebe,ggwiebe@gmail.com,1,1,9999-12-31,2022-01-13 16:12:42.453,glenn@wiebes.net
3 1002,Graeme,Wiebe,glenn@wiebes.net,2,1,9999-12-31,2022-01-13 16:12:42.453,glenn@wiebes.net
```

```
1 -- SILVER - View against Bronze that will be used to load silver incrementally with APPLY CHANGES INTO
2 CREATE TEMPORARY STREAMING LIVE VIEW customer_bronze2silver_v (
3   CONSTRAINT valid_id          EXPECT (id IS NOT NULL) ON VIOLATION DROP ROW,
4   CONSTRAINT valid_active      EXPECT (active BETWEEN 0 AND 1) ON VIOLATION DROP ROW,
5   CONSTRAINT valid_channel     EXPECT (sales_channel IS NOT NULL),
6   CONSTRAINT valid_first_name  EXPECT (first_name IS NOT NULL),
7   CONSTRAINT valid_last_name   EXPECT (last_name IS NOT NULL)
8 )
9 COMMENT "View of cleansed Bronze Customer for loading into Silver."
10 AS SELECT c.id,
11       UPPER(c.first_name) as first_name,
12       UPPER(c.last_name) as last_name,
13       c.email,
14       sc.channelName sales_channel,
15       c.active,
16       c.active_end_date,
17       c.update_dt,
18       c.update_user,
19       current_timestamp() dlt_ingest_dt,
20       "CustomerMaster-Live" dlt_ingest_procedure,
21       current_user() dlt_ingest_principal
22   FROM STREAM(live.customer_bronze) c
23   LEFT JOIN live.sales_channel_v sc
24   ON c.channel = sc.channelId
25 ;
```

```
1 -- SILVER - Incremental Customer table with APPLY CHANGES INTO change handling
2 CREATE INCREMENTAL LIVE TABLE customer_silver
3 TBLPROPERTIES ("quality" = "silver")
4 COMMENT "Clean, merged customers"
5 ;
```

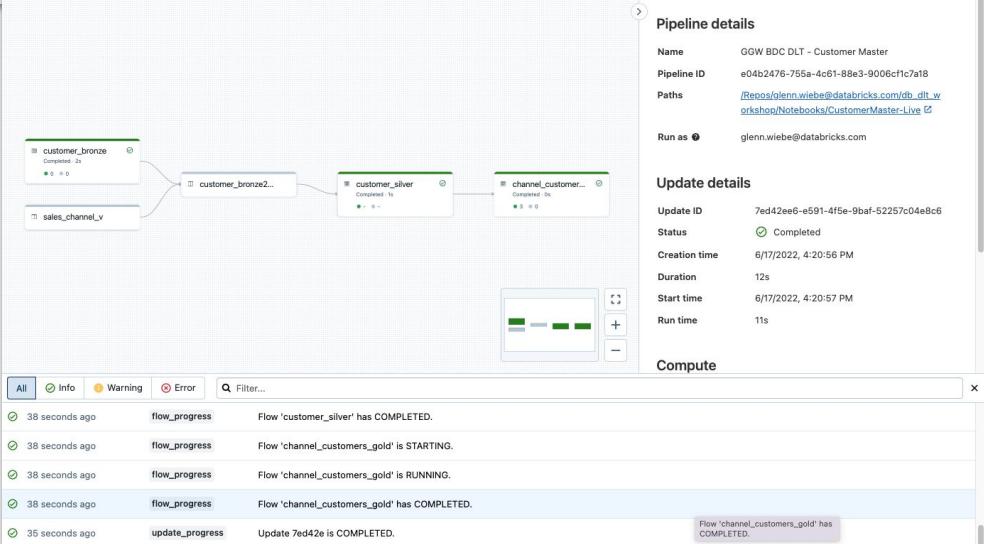
Cmd 12

```
1 APPLY CHANGES INTO live.customer_silver
2 FROM stream(live.customer_bronze2silver_v)
3   KEYS (id)
4   APPLY AS DELETE WHEN active = 0
5   SEQUENCE BY update_dt
6 ;
```

GGW BDC DLT - Customer Master

6/17/2022, 4:20:56 PM · Completed

Development Production ⚙ Delete Permissions Settings Schedule (1) Start



DLT Lab Screenshots - DLT Event Log Config

Create pipeline

UI JSON

* Product edition
Advanced

Help me choose

* Pipeline name
GGW BDC DLT - Customer Master

* Notebook libraries
/Repos/glen.wiebe@databricks.com/db_dlt_workshop/Notebooks/CustomerMaster

Add notebook library

Configuration
Add configuration

Target
ggw_retail_wshp

Storage location
dbfs:/Users/glen.wiebe@databricks.com/dlt_retail_wshp

Pipeline mode
Triggered Continuous

Autopilot options
Enable autoscaling Preview
Learn about Enhanced Autoscaling Preview

Cluster
* Min workers Max workers
1 3

Use Photon Acceleration Preview

Shared Endpoint - Photon (L)

```
86 -- DLT Event Log & Data Quality Scores
87
88 CREATE TABLE ggw_retail_wshp.event_log
89 USING delta
90 LOCATION '/Users/glen.wiebe@databricks.com/dlt_retail_wshp/system/events'
91 ;
92
93 SELECT id,
94 -- sequence,
95 timestamp,
96 level,
97 event_type,
98 message,
99 details
100 FROM ggw_retail_wshp.event_log
101 WHERE level IN ({{ level }})
102 ;
103
```

Level
METRICS X

Table						+ Add visualization
id	timestamp	level	event_type	message	details	
c643e420-ee82-11ec-9a7d-00163e4adf2b	2022-06-17 21:16:38.114	METRICS	cluster_utilization	Reported cluster utilization metrics.	{"cluster_utilization": {"summary_duration_ms":60000,"num_task_slots":8,"avg_num_task_slots":7.99893331555259, "avg_task_slot_utilization":0.0,"num_executors":1,"avg_num_queued_tasks":0.0}}	
1ec5b220-ee80-11ec-9a7d-00163e4adf2b	2022-06-17 20:57:38.114	METRICS	cluster_utilization	Reported cluster utilization metrics.	{"cluster_utilization": {"summary_duration_ms":60000,"num_task_slots":8,"avg_num_task_slots":7.99906666666667, "avg_task_slot_utilization":0.0,"num_executors":1,"avg_num_queued_tasks":0.0}}	
8a0f8420-ee80-11ec-9a7d-00163e4adf2b	2022-06-17 21:00:38.114	METRICS	cluster_utilization	Reported cluster utilization metrics.	{"cluster_utilization": {"summary_duration_ms":60000,"num_task_slots":8,"avg_num_task_slots":7.99906665110851, "avg_task_slot_utilization":0.0,"num_executors":1,"avg_num_queued_tasks":0.0}}	
60a32820-ee81-11ec-9a7d-00163e4adf2b	2022-06-17 21:06:38.114	METRICS	cluster_utilization	Reported cluster utilization metrics.	{"cluster_utilization": {"summary_duration_ms":60000,"num_task_slots":8,"avg_num_task_slots":7.99893333333333, "avg_task_slot_utilization":0.0,"num_executors":1,"avg_num_queued_tasks":0.0}}	
5b11c030-ee7f-11ec-9a7d-00163e4adf2b	2022-06-17 20:52:09.779	METRICS	flow_progress	Completed a streaming update of 'customer_silver'.	{"flow_progress": {"status": "RUNNING", "metrics": {"data_quality": {}}}}	

DLT Lab Screenshots - DLT Event Queries

```

183
184 --- Lineage
185
186 SELECT details:flow_definition.output_dataset,
187     details:flow_definition.input_datasets,
188     details:flow_definition.flows,
189     details:flow_definition.schemas,
190     -- , details:flow_definition.explain_text,
191     -- , details:flow_definition
192 FROM ggw_retail_wshp.event_log
193 WHERE details:flow_definition IS NOT NULL
194 ORDER BY timestamp
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300

```

Refresh schedule Never

+ Add visualization

Table

output_dataset	input_datasets	flow_type	schema
sales_channel_v	NULL	COMPLETE	[{"name": "channelId", "path": ["channelId"], "data_type": "INTEGER"}, {"name": "channelName", "path": ["channelName"], "data_type": "STRING"}, {"name": "description", "path": ["description"], "data_type": "STRING"}]
customer_bronze	NULL	INCREMENTAL	[{"name": "id", "path": ["id"], "data_type": "INTEGER"}, {"name": "first_name", "path": ["first_name"], "data_type": "STRING"}, {"name": "last_name", "path": ["last_name"], "data_type": "STRING"}, {"name": "email", "path": ["email"], "data_type": "STRING"}, {"name": "channel", "path": ["channel"], "data_type": "INTEGER"}, {"name": "active", "path": ["active"], "data_type": "INTEGER"}, {"name": "update_user", "path": ["update_user"], "data_type": "STRING"}]
customer_bronze2silver_v	["sales_channel_v", "customer_bronze"]	INCREMENTAL	[{"name": "id", "path": ["id"]}, {"name": "last_name", "path": ["last_name"]}, {"name": "data_type", "path": ["data_type"]}, {"name": "active", "path": ["active"]}, {"name": "update_dt", "path": ["update_dt"]}, {"name": "dtl_ingest_dt", "path": ["dtl_ingest_dt"]}, {"name": "dtl_ingest_principal", "path": ["dtl_ingest_principal"]}, {"name": "dtl_update_principal", "path": ["dtl_update_principal"]}]
customer_silver	["customer_bronze2silver_v"]	CHANGE	[{"name": "id", "path": ["id"]}, {"name": "last_name", "path": ["last_name"]}, {"name": "data_type", "path": ["data_type"]}, {"name": "active", "path": ["active"]}, {"name": "update_dt", "path": ["update_dt"]}, {"name": "dtl_ingest_dt", "path": ["dtl_ingest_dt"]}, {"name": "dtl_ingest_principal", "path": ["dtl_ingest_principal"]}, {"name": "dtl_update_principal", "path": ["dtl_update_principal"]}]
channel_customers_gold	["customer_silver"]	COMPLETE	[{"name": "sales_channel", "path": ["sales_channel"]}, {"name": "most_recent_cus", "path": ["most_recent_cus"]}]

{} 📈 📉 🔍 🌐 LIMIT 1000

Table

id	flow_progress	status	num_output_rows	backlog_bytes	dropped_records	expectations
5589bc30-ee7f-11ec-9a7d-00163e4adf2b	{"status": "RUNNING", "metrics": {"num_output_rows": 6, "data_quality": {"dropped_records": 0}}}	RUNNING	6	NULL	0	NULL
56269870-ee7f-11ec-9a7d-00163e4adf2b	{"status": "COMPLETED", "metrics": {"num_output_rows": 6, "data_quality": {"dropped_records": 0}}}	COMPLETED	6	NULL	0	NULL
5b11c030-ee7f-11ec-9a7d-00163e4adf2b	{"status": "RUNNING", "metrics": {}, "data_quality": {}}	RUNNING	NULL	NULL	NULL	NULL
5b167b20-ee7f-11ec-9a7d-00163e4adf2b	{"status": "COMPLETED", "metrics": {}, "data_quality": {}}	COMPLETED	NULL	NULL	NULL	NULL
5b57f0f0-ee7f-11ec-9a7d-00163e4adf2b	{"status": "RUNNING", "metrics": {"backlog_bytes": 4432}}	RUNNING	NULL	4432	NULL	NULL
5ba6fb50-ee7f-11ec-9a7d-00163e4adf2b	{"status": "COMPLETED", "metrics": {"num_output_rows": 3, "data_quality": {"dropped_records": 0}}}	COMPLETED	3	NULL	0	NULL

DLT Lab Screenshots - DLT Event Queries & Dashboards

Retail DLT Monitoring

Number of Customers by Channel

Channel type

Channel Type	Count
PARTNER	2
RETAIL	1
WEB	1

4 months ago

Pass/Fail by Constraint Name

Silver Metrics & Quarantine Counts (of most recent load)

DLT Constraint Name

4 months ago

Quarantined Customer Records

Records not loaded due to Violations of Quality Expectations

id	first_name	last_name	email	channel	channelName
1007	Jane	NULL	noreply@databricks.com	NULL	NULL

Clean (Silver) Customers

Customers that have passed all DLT quality expectations

Sales channel

id	first_name	last_name	email	sales_channel	active
1003	DILLON	BOSTWICK	dillon@databricks.com	WEB	1
1004	FRANCO	PATANO	franco.patano@databricks.com	PARTNER	1
1001	GLENN	WIEBE	ggwiebe@gmail.com	RETAIL	1
1005	CHRIS	FISH	chris.fish@databricks.com	PARTNER	1

4 months ago

Customers (bronze) by Last Name

Customer records loaded from raw CloudFiles storage

Last Name

LastName	W%
Wiebe	100%

id	first_name	last_name	email	channel	active	active_end_date
1001	Glenn	Wiebe	ggwiebe@gmail.com	1	1	9999-1
1002	Graeme	Wiebe	glenn@wiebes.net	2	1	9999-1
1002	Glenn	Wiebe	glenn@wiebes.net	1	1	9999-1
1002	Glenn	Wiebe	glenn@wiebes.net	1	0	2022-0
1007	Jane	NULL	noreply@databricks.com	NULL	1	9999-1

4 months ago

Customer Lineage Schema

Schema of Customer DLT Pipeline

max_timestamp	output_dataset	input_datasets	flow_type
4	4	4	4

DLT Lab Screenshots - Pipeline Workflows

Workflows > Jobs > GGW BDC DLT - Retail

GGW BDC DLT - Retail

Runs Tasks

The screenshot shows the Databricks UI for a pipeline named 'GGW BDC DLT - Retail'. On the left, a pipeline diagram is displayed with two stages: 'Load_SalesChannel_Reference' (Delta Live Table Pipeline) and 'Load_Customer_Master' (Delta Live Table Pipeline). A task configuration dialog is open in the foreground, showing the following details:

- Task name ***: Load_Customer_Master
- Type ***: Delta Live Tables pipeline
- Pipeline ***: GGW BDC DLT - Customer Master
- Depends on**: Load_SalesChannel_Reference

On the right, the pipeline details are shown:

- Job details**: Job ID 903895552193943, Creator glenn.wiebe@databricks.com, Run as glenn.wiebe@databricks.com, Tags
- Git**: Add Git setting
- Schedule**: Paused - At 04:00 UTC, Edit schedule
- Runs**: Active runs, Start time Run ID Launched Duration Status Actions
- Email alerts**: glenn.wiebe@databricks.com, On failure, Edit alerts
- Completed runs (past 60 days)**: Refresh, Start time Run ID Launched Duration Status Actions

Run now / Run now with different parameters

Workflows / Delta Live Tables / Pipeline details

GGW BDC DLT - Channel Reference

Development Production ⚡ Delete Permissions Settings Schedule (1) Start

6/17/2022, 4:02:25 PM - Completed

Pipeline details

Name GGW BDC DLT - Channel Reference

Pipeline ID 3cc03976-0c7f-4495-b108-6862db67f3f6

Paths [/Repos/glen.wiebe@databricks.com/db/dlt_workshop/Notebook@ChannelReferenceLive@](#)

Run as glenn.wiebe@databricks.com

Update details

Update ID 1cfdf51e-078a-47d4-ba03-bfffcdeada3e

All Info Warning Error Filter...

- 3 minutes ago flow_progress Flow 'channel' has COMPLETED.
- 3 minutes ago flow_progress Flow 'channel_master' is STARTING.
- 3 minutes ago flow_progress Flow 'channel_master' is RUNNING.
- 3 minutes ago flow_progress Flow 'channel_master' has COMPLETED.
- 3 minutes ago update_progress Update 1cfdf51e is COMPLETED.

Workflows / Delta Live Tables / Pipeline details

GGW BDC DLT - Customer Master

Development Production ⚡ Delete Permissions Settings Schedule (1) Stop

6/17/2022, 4:06:52 PM - Running

Pipeline details

Name GGW BDC DLT - Customer Master

Pipeline ID e04b2476-755a-4c61-88e3-9006fc1cfa18

Paths [/Repos/glen.wiebe@databricks.com/db/dlt_workshop/Notebook@CustomerMasterLive@](#)

Run as glenn.wiebe@databricks.com

Update details

Update ID 7ba39a43-83a4-43d3-a121-2b09433502f

All Info Warning Error Filter...

- 9 minutes ago user_action User glenn.wiebe@databricks.com started an update.
- 9 minutes ago update_progress Update 7ba39a43 is WAITING_FOR_RESOURCES.
- 5 minutes ago update_progress Update 7ba39a43 is INITIALIZING.
- 5 minutes ago update_progress Update 7ba39a43 is SETTING_UP_TABLES.
- 4 minutes ago flow_definition Flow 'customer_bronze' defined as INCREMENTAL.

Update 7ba39a43 is SETTING_UP_TABLES.

