Eastern Analytics, Inc

We Are Data Analytics People

EASTERN ANALYTICS
SYSTEMS, SOLUTIONS & SERVICES

+1 (781) 757-7036

https://analytics-people.com

Webinar Series

AZURE DATABRICKS: INTRODUCTION TO THE DATABRICKS LAKEHOUSE PLATFORM

April 5, 2023

Microsoft

EASTERN ANALYTICS
SYSTEMS, SOLUTIONS & SERVICES

About Us

Eastern Analytics' architects have been building Analytics platforms and helping customers unlock the true value of data for over 25 years.

We specialize in Microsoft Analytics, Azure AI/ML and Power BI.

Scott Pietroski

As Eastern Analytics' founding partner, Scott's focus is Solution Architecture, customer engagement and project delivery.

Scott.Pietroski@eastern-analytics.us
781-757-7036

Kerrilee Pietroski

Kerrilee is Eastern Analytics' Director of Marketing & Communications, leading strategic marketing initiatives and corporate communications.

Kerrilee.Pietroski@eastern-analytics.us
781-783-7610

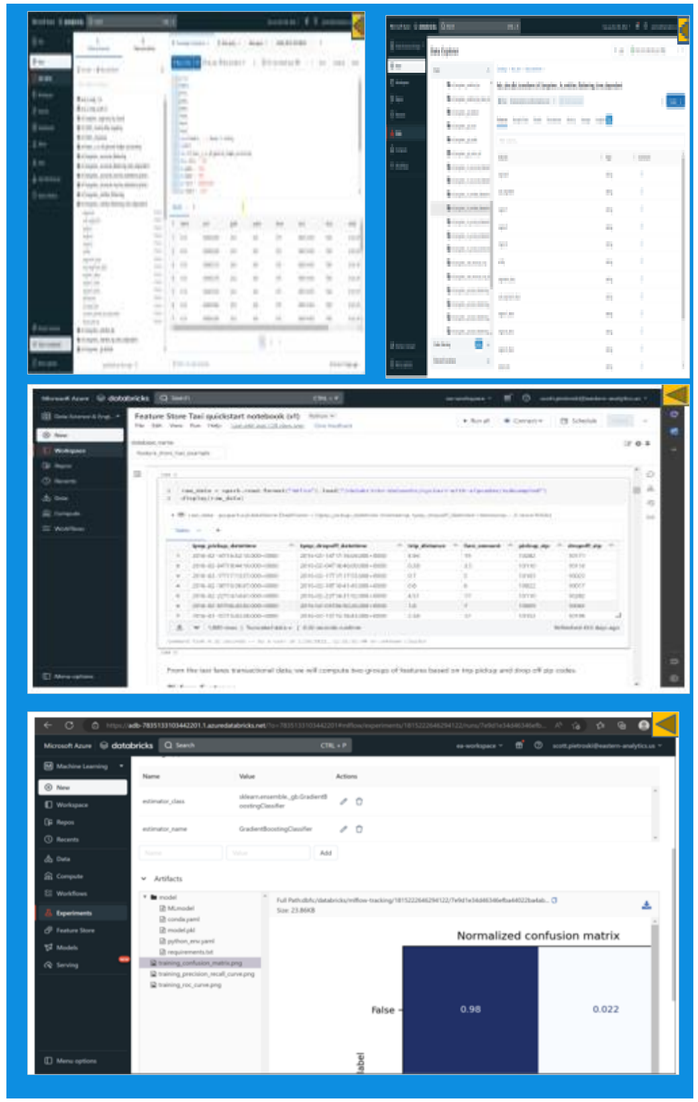
Today's Presentation:

- Databricks Intro
- Databricks on MS Azure
- Databricks Lakehouse & Delta Lake (how they work together)
- Easy to understand use cases
- Q&A

Note: Today's presentation references information from Eastern Analytics own projects and information obtained from the Databricks partner program.

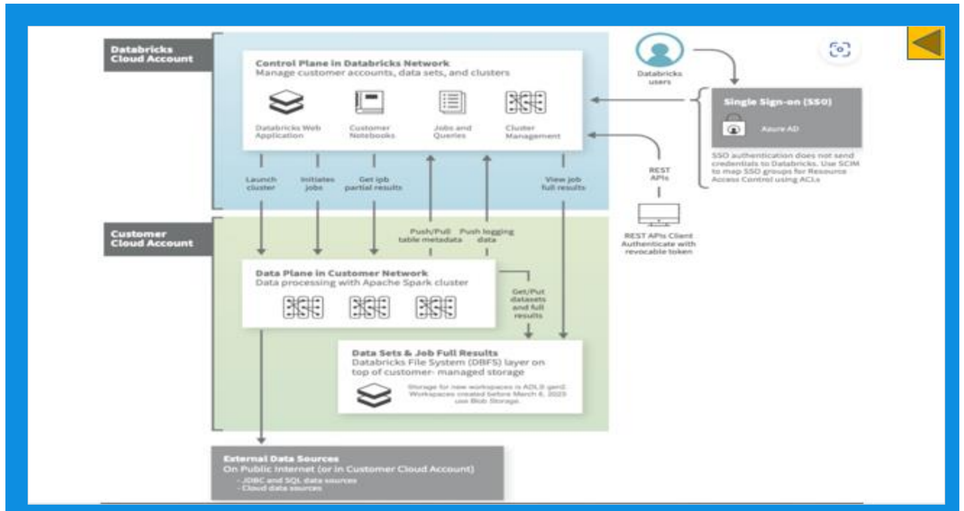
Databricks – Unified data management system

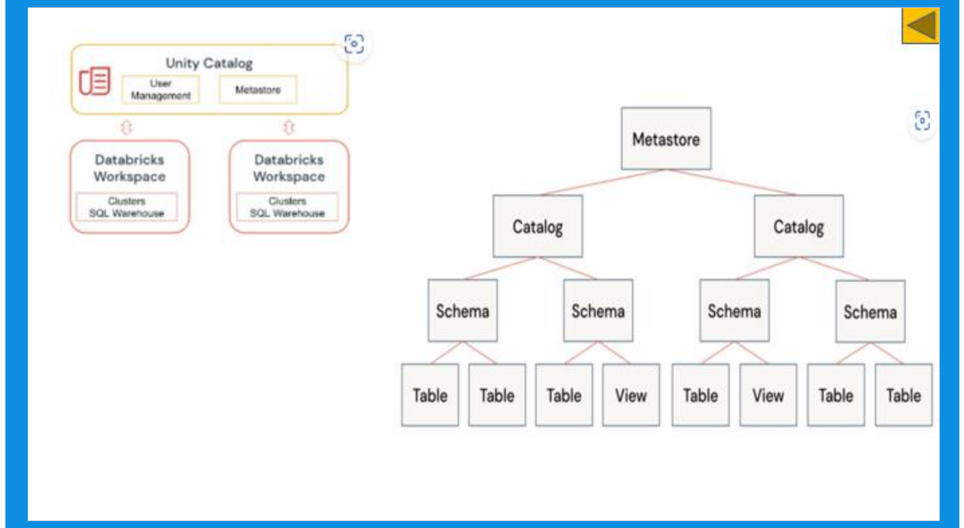
- **Cloud Database Management System/SQL** – Uses cheap blob storage to store large quantities of data. Supports basic SQL, tables and views along with ACID transaction capabilities
- **Cloud Data Engineering Platform/Engineering** – Designed to support streaming & batch ingestion, Change Data Capture (CDC), allows for massive scaling of compute resource. Supports multiple languages including Python, Scala, R and SQL.
- **Data Science Platform/ML** – Includes AutoML, has full integration with MLFlow and comes with pre-defined computes that include standard libraries.



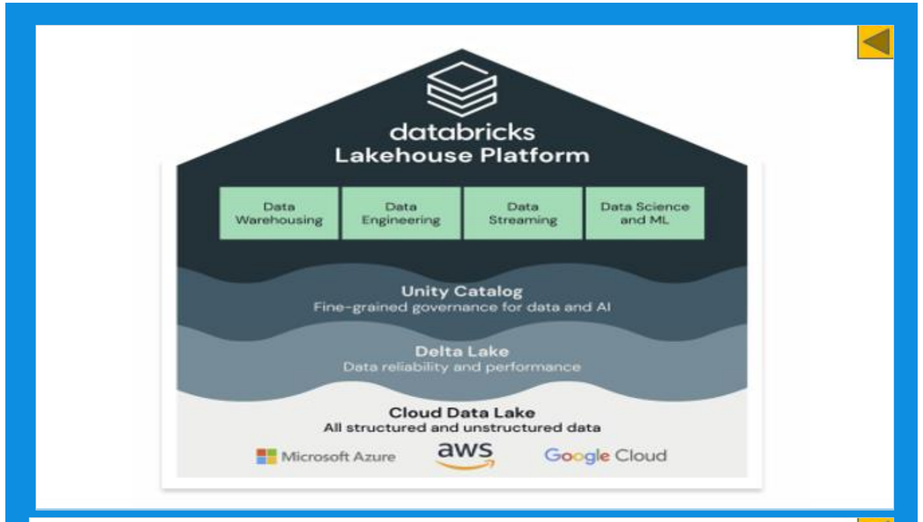
Databricks and MS Azure

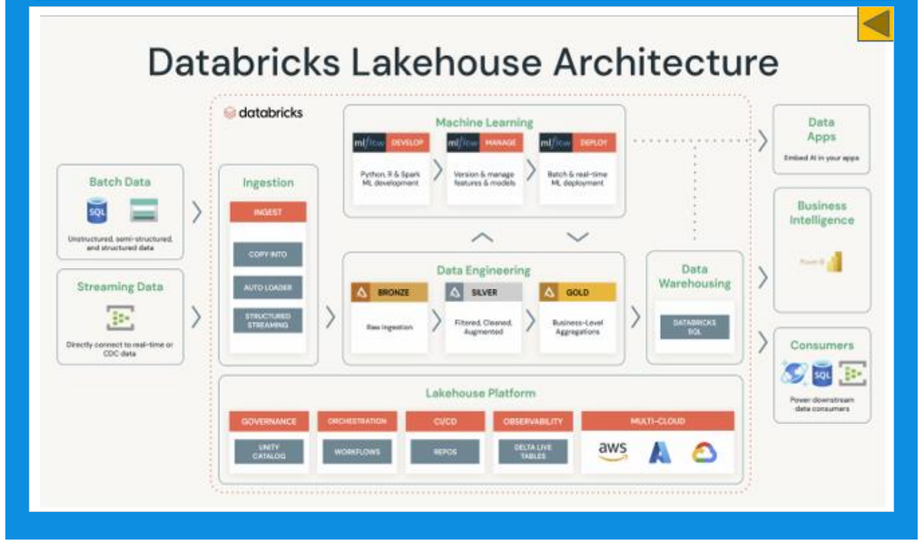
- **Databricks on Azure**
  - **Data Security** – Databricks is built for the cloud. With Databricks, your data stays within your own storage account.
  - **Unity Catalog** – Provides account level control for securing workspaces and data. Provides Data Governance including logging and lineage across the platform. Must be implemented.
  - **Enforcement** – Access controls are enforced everywhere. They are enforced in programing and when accessing data via a SQL Endpoint





Databricks Lakehouse – Why?





Databricks Lakehouse

- **Data Lake** – A storage area designed to accept massive amounts of raw data. Data can be structured, semi-structured or unstructured – and is organized by different sources, formats and data types
- **Data Warehouse –** A structured data store (usually RDBMS) where data is cleaned, transformed and aggregated for use in BI & Reporting
- **Databricks "Lakehouse"**– A combined Data Lake and Data Warehouse in one. Major advantage – it includes functionality to ingest, process and store massive amounts of data for further downstream use

Delta Lake – A unified data management system

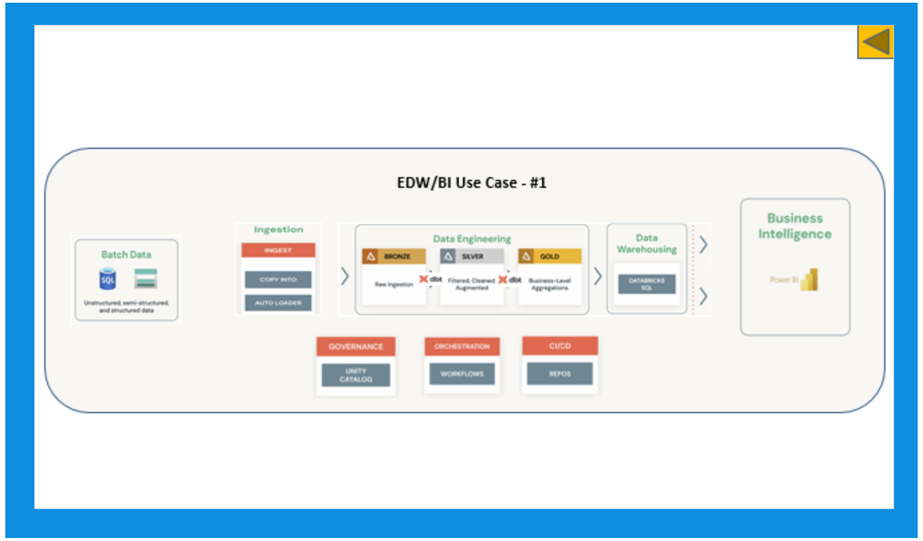
Delta Lake

- Sits on top of your data lake
- Open source storage framework
- Made up of Delta Tables - Delta Tables STORE data
- Includes a transaction log to support CDC
- Support "Time Travel" or rollbacks
- Based on "Medallion Approach"
- Allows SQL access via SQL warehouse compute and endpoints

Delta Live Tables

- They are how you MOVE data thru tables
- They are pipelines that tie together notebooks
- Notebooks/scripts are used for actual code
- Support streaming or batch jobs
- Shows lineage of data flows
- Orchestration tool included with Databricks

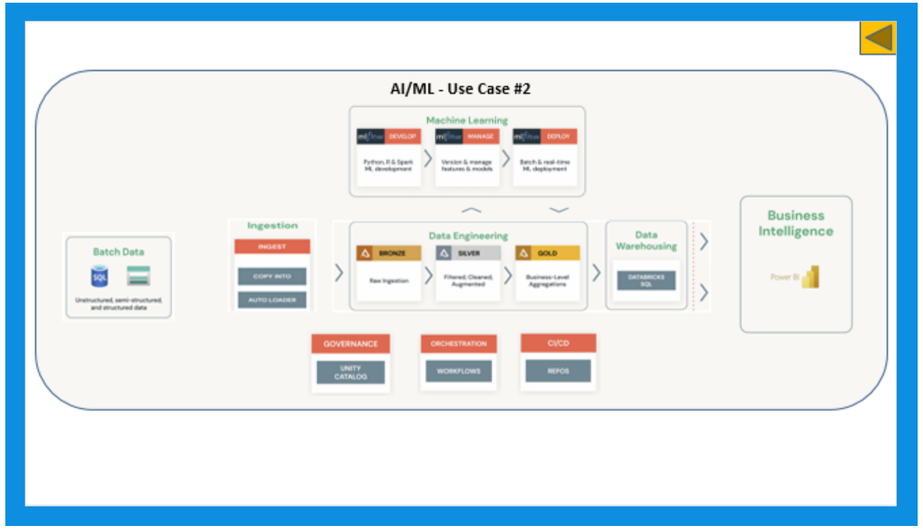
Use Case #1 – EDW/BI



Using Delta Tables + SQL Warehouse for Power BI

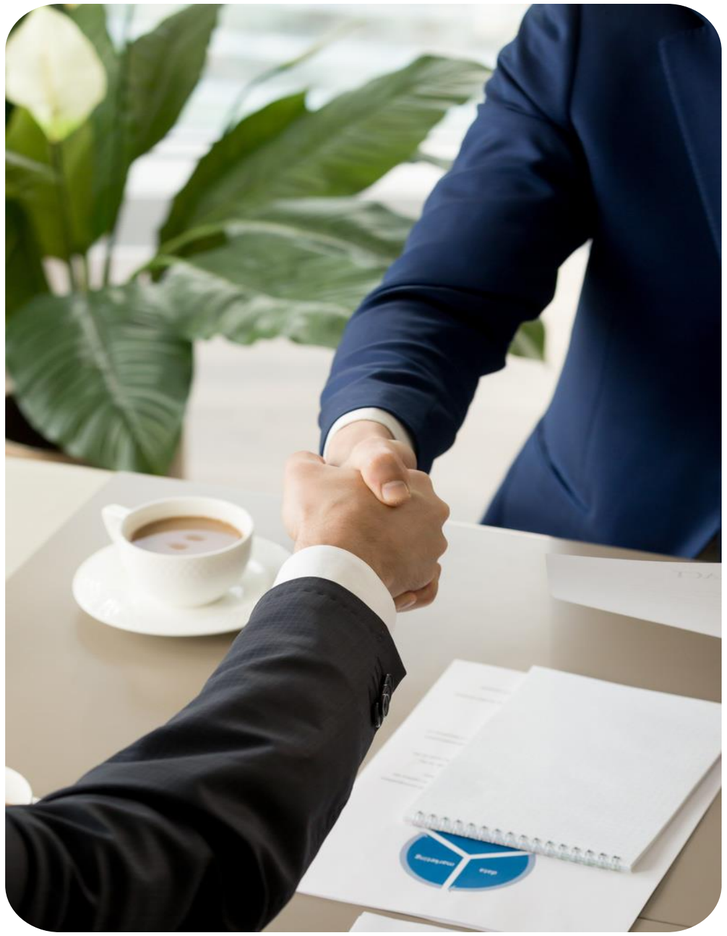
- **Unity Catalog –** Used for Data Governance. Controls and monitors access cross workspaces and catalogs
- **Auto Loader** – Used for data ingestion into the Delta Lake. Data is dropped in a landing zone and auto ingested via Data Engineering notebooks.
- **Data Transformations** – Performed via Notebooks/DBT, run in batch for medallion table hops. Orchestration via scheduled jobs. Workflows can be used for batch processing within the medallion flow
- **Power BI** – Consumes data from the "Gold" layer via a SQL Warehouse/serverless endpoint.

Use Case #2 – Data Science & ML



Databricks – Built for Data Science

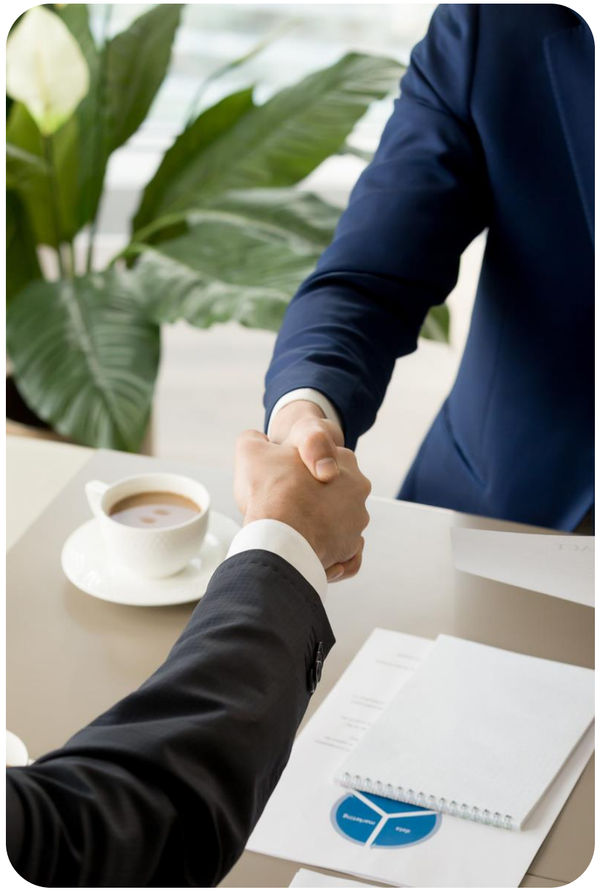
- **Experiments** – Coded in notebooks using standard ML libraries and compute. Can be coded in Python, Scala & R.
- **Scalable Compute** – Pre-configured clusters come with standard ML libraries such as SciKit Learn and TensorFlow.
- **MLFlow** –AutoML uses MLFlow for experiment tracking. MLFlow is included in ML computes to easily track experiments, statistics and outputs (models).
- **Models** – Models are stored in a model repository and can be tagged/versioned. Models can be published & for real-time consumption via Serving capabilities.
- **Feature Store** - Used for feature engineering. Provides consistency in feature reuse across models.



Q & A

+1 (781) 757-7036

https://analytics-people.com



Thank You

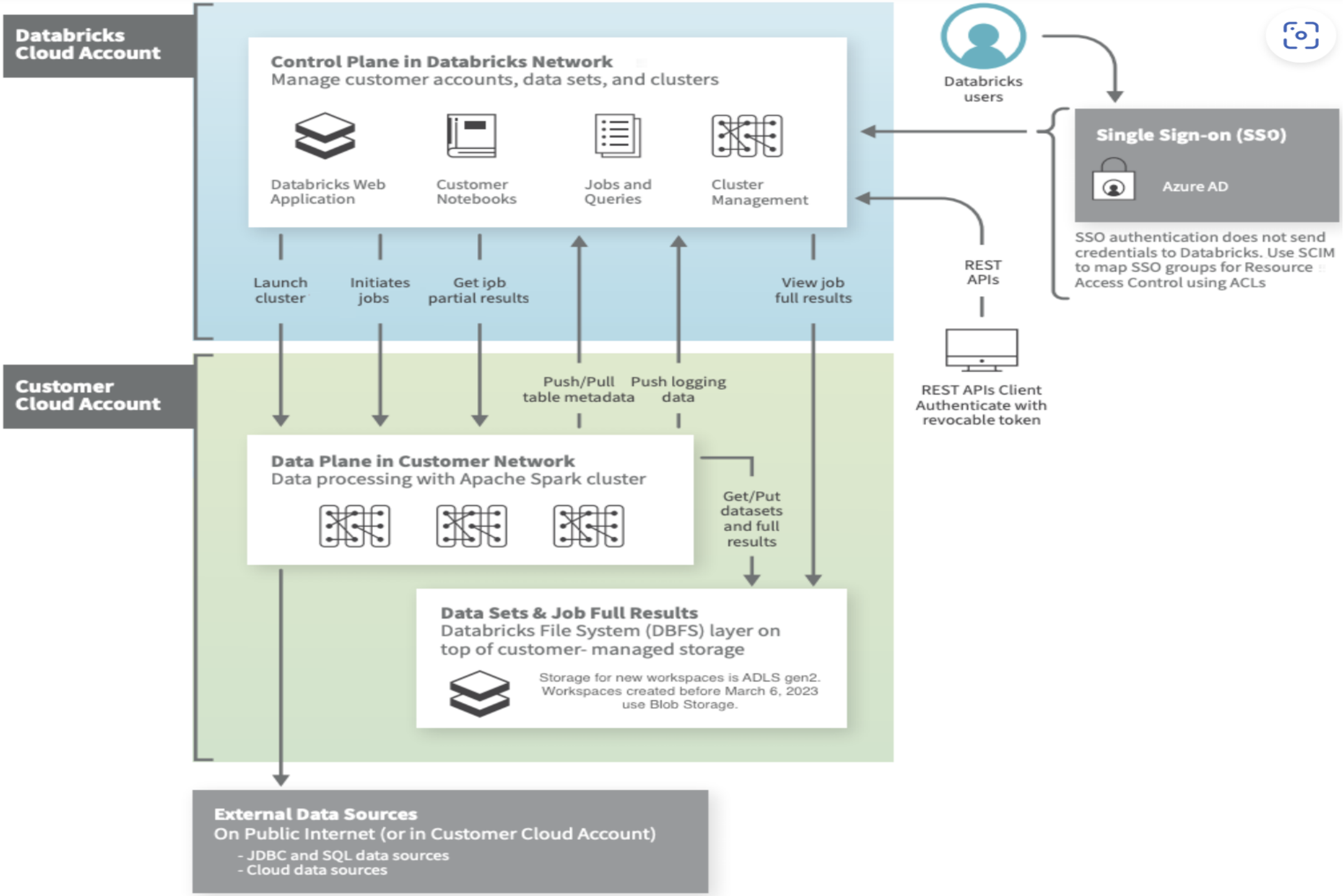
We Are Here to Help

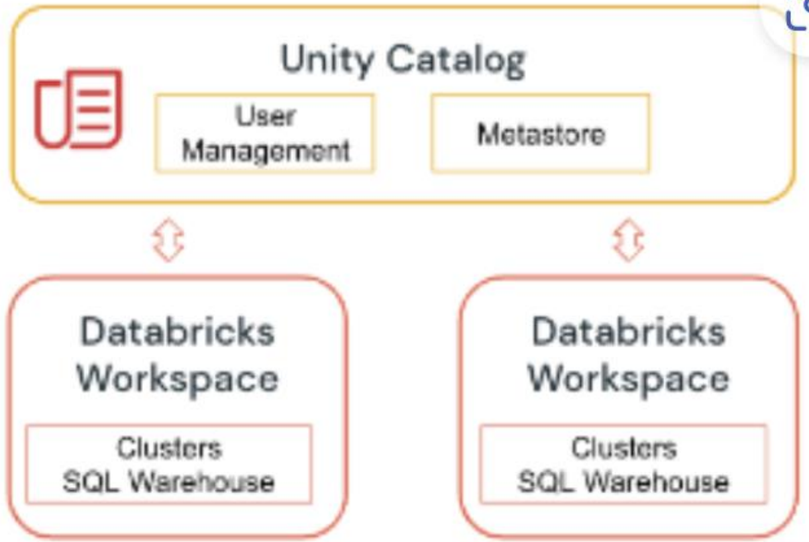
Let us know how we can help take your company to the next level to gain the competitive advantage.

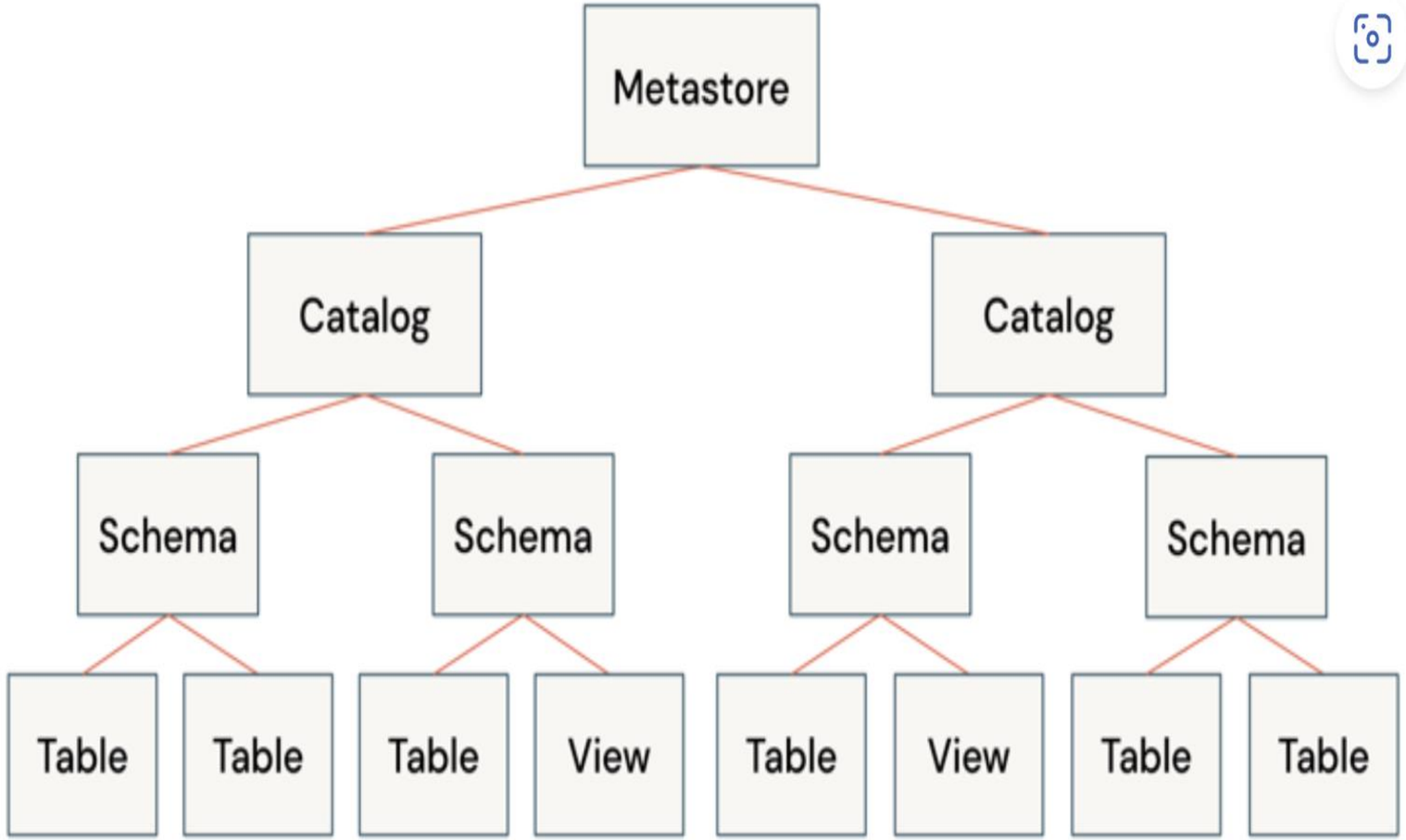
+1 (781)757-7036

https://analytics-people.com

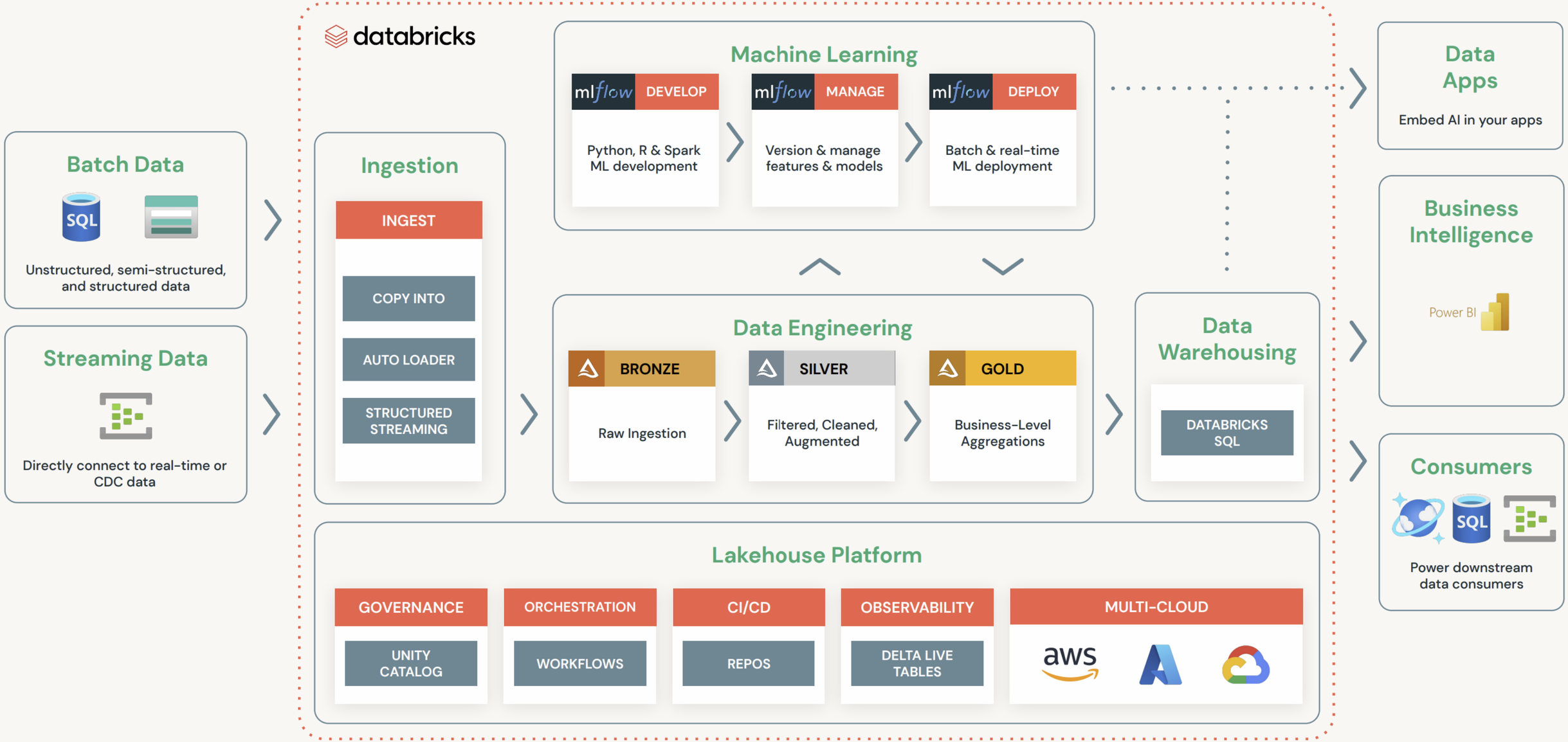
EASTERN ANALYTICS
SYSTEMS, SOLUTIONS & SERVICES

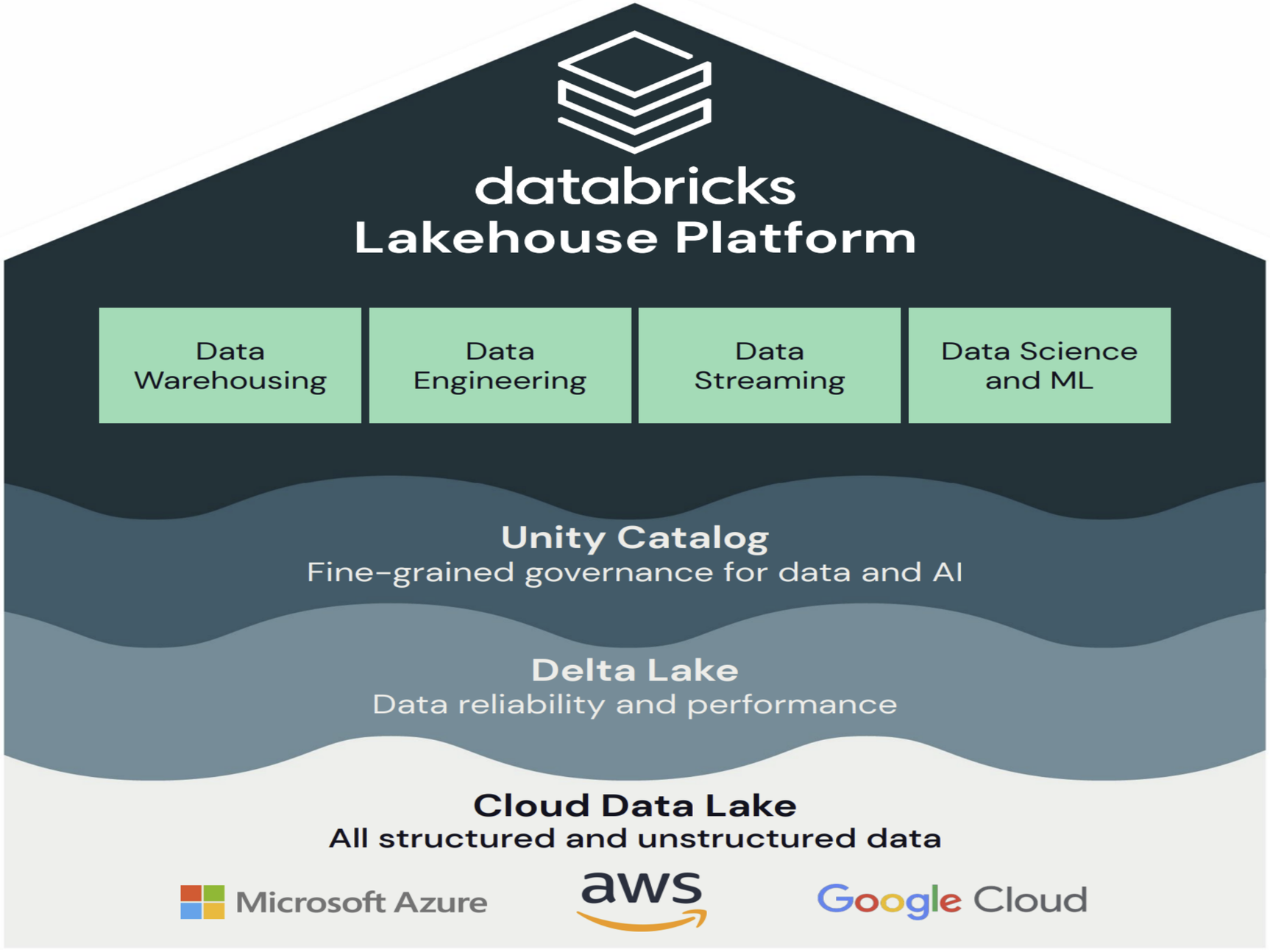

Databricks Cloud Account
Control Plane in Databricks Network
Manage customer accounts, data sets, and clusters
Databricks Web Application
Customer Notebooks
Jobs and Queries
Cluster Management
Launch cluster
Initiates jobs
Get job partial results
View job full results
Push/Pull table metadata
Push logging data
Customer Cloud Account
Data Plane in Customer Network
Data processing with Apache Spark cluster
Get/Put datasets and full results
Data Sets & Job Full Results
Databricks File System (DBFS) layer on top of customer- managed storage
Storage for new workspaces is ADLS gen2. Workspaces created before March 6, 2023 use Blob Storage.
External Data Sources
On Public Internet (or in Customer Cloud Account)
- JDBC and SQL data sources
- Cloud data sources
Databricks users
Single Sign-on (SSO)
Azure AD
SSO authentication does not send credentials to Databricks. Use SCIM to map SSO groups for Resource Access Control using ACLs
REST APIs
REST APIs Client Authenticate with revocable token

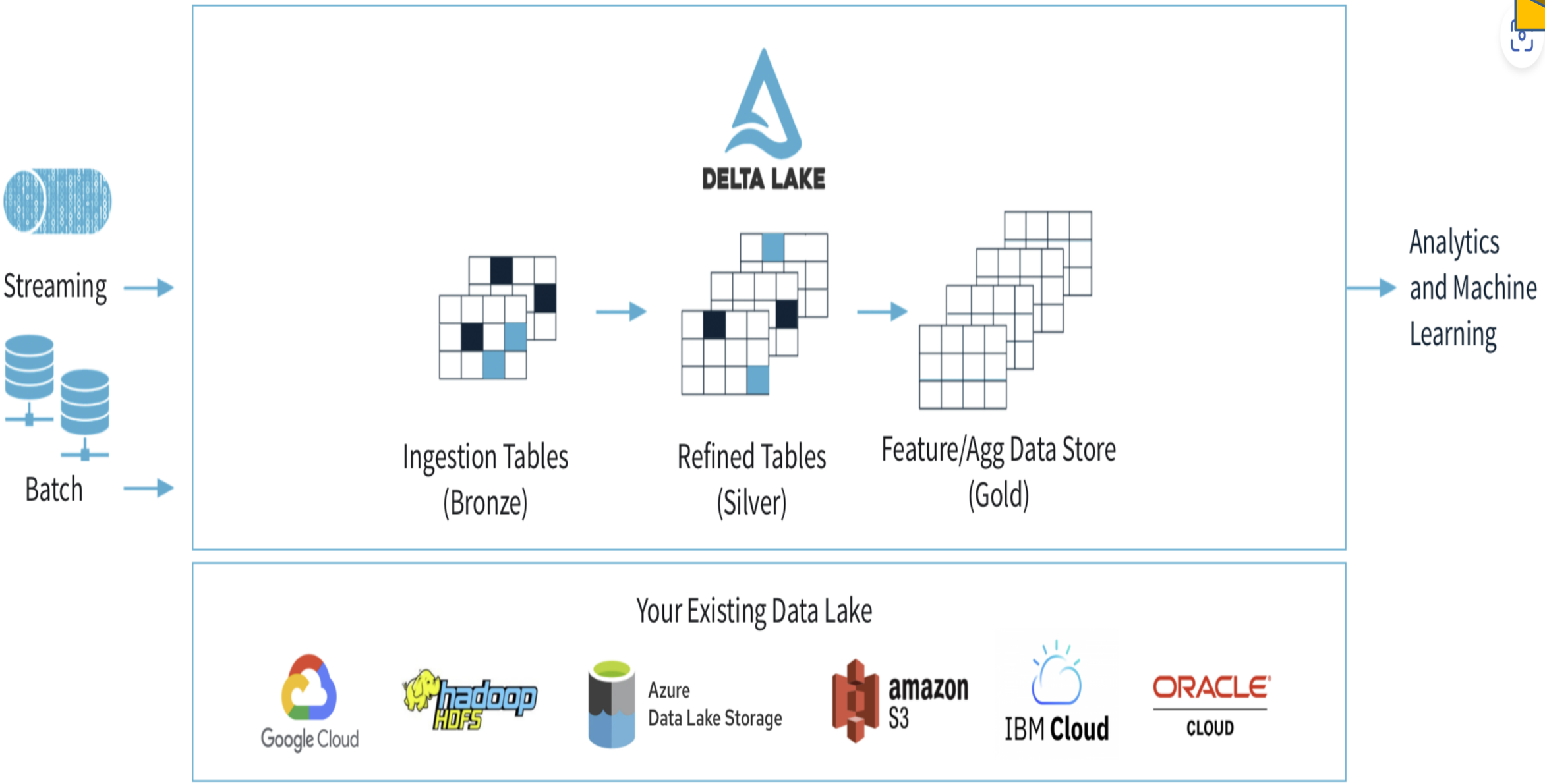

Unity Catalog
User Management
Metastore
Databricks Workspace
Clusters
SQL Warehouse
Databricks Workspace
Clusters
SQL Warehouse


Metastore
Catalog
Catalog
Schema
Schema
Schema
Schema
Table
Table
Table
View
Table
View
Table
Table

Databricks Lakehouse Architecture




databricks
Lakehouse Platform
Data Warehousing
Data Engineering
Data Streaming
Data Science and ML
Unity Catalog
Fine-grained governance for data and AI
Delta Lake
Data reliability and performance
Cloud Data Lake
All structured and unstructured data
Microsoft Azure
aws
Google Cloud


Streaming
Batch
DELTA LAKE
Ingestion Tables
(Bronze)
Refined Tables
(Silver)
Feature/Agg Data Store
(Gold)
Analytics
and Machine
Learning
Your Existing Data Lake
Google Cloud
hadoop
HDFS
Azure
Data Lake Storage
amazon
S3
IBM Cloud
ORACLE
CLOUD

Microsoft Azure | databricks | Search CTRL + P | ea-workspace | scott.pietroski@eastern-analytics.us

Data Science & Engi...

- New
- Workspace
- Repos
- Recents
- Data
- Compute
- Workflows
- Menu options

Feature Store Taxi quickstart notebook (v1)

Python

File Edit View Run Help Last edit was 128 days ago Give feedback

Run all | Connect | Schedule | Share

database_name

feature_store_taxi_example

Cmd 5

```
raw_data = spark.read.format("delta").load("/databricks-datasets/nyctaxi-with-zipcodes/subsampled")
display(raw_data)
```

raw_data: pyspark.sql.dataframe.DataFrame = [tpep_pickup_datetime: timestamp, tpep_dropoff_datetime: timestamp ... 4 more fields]

Table +

	tpep_pickup_datetime	tpep_dropoff_datetime	trip_distance	fare_amount	pickup_zip	dropoff_zip
1	2016-02-14T16:52:13.000+0000	2016-02-14T17:16:04.000+0000	4.94	19	10282	10171
2	2016-02-04T18:44:19.000+0000	2016-02-04T18:46:00.000+0000	0.28	3.5	10110	10110
3	2016-02-17T17:13:57.000+0000	2016-02-17T17:17:55.000+0000	0.7	5	10103	10023
4	2016-02-18T10:36:07.000+0000	2016-02-18T10:41:45.000+0000	0.8	6	10022	10017
5	2016-02-22T14:14:41.000+0000	2016-02-22T14:31:52.000+0000	4.51	17	10110	10282
6	2016-02-05T06:45:02.000+0000	2016-02-05T06:50:26.000+0000	1.8	7	10009	10065
7	2016-02-15T15:03:28.000+0000	2016-02-15T15:18:45.000+0000	2.58	12	10153	10199

1,000 rows | Truncated data | 0.32 seconds runtime Refreshed 433 days ago

Command took 0.32 seconds -- by a user at 1/26/2022, 11:51:51 AM on unknown cluster

Cmd 6

From the taxi fares transactional data, we will compute two groups of features based on trip pickup and drop off zip codes.

Pickup features

Microsoft Azure | databricks | Search CTRL + P | has-aa-dev-hdc-dbw | pietros@[redacted]

SQL

- New
- SQL Editor
- Workspace
- Queries
- Dashboards
- Alerts
- Data
- SQL Warehouses
- Query History
- Partner Connect
- 1/3 Tasks Completed
- Menu options

Schema browser | Past executions

hdc_dev > dbt_transform

Filter tables & columns...

- ana_2_step_1_fs
- ana_2_step_multi_fs
- trf_anaplan__expenses_by_brand
- trf_d365__brand_title_mapping
- trf_d365__expenses
- trf_hana__z_cv_df_general_ledger_accounting
- trf_longview__accounts_flattening
- trf_longview__accounts_flattening_time_dependent
- trf_longview__accounts_income_statement_planni...
- trf_longview__accounts_income_statement_planni...
- trf_longview__entities_flattening
- trf_longview__entities_flattening_time_dependent
  - segment STRING
  - sub_segment STRING
  - region1 STRING
  - region2 STRING
  - region3 STRING
  - entity STRING
  - segment_desc STRING
  - sub_segment_desc STRING
  - region1_desc STRING
  - region2_desc STRING
  - region3_desc STRING
  - datanode STRING
  - timeperiods STRING
  - current_period_at_load_time STRING
  - fiscal_period STRING
- trf_longview__entities_hp
- trf_longview__entities_hp_time_dependent
- trf_longview__gl_default

Last fetch: an hour ago

Percentage_Calculation | New query | New query | HANA_WITH_DECIMALS

Run (1000) | hdc_dev . dbt_transform | HDC Dev Warehouse Pro S | Save* | Schedule | Share

```
select
rbukrs,
prctr,
gjahr,
poper,
rbusa,
racct,
rhcur,
concat(rbukrs, '_', rbusa) as entity,
sum(hsl)
from trf_hana__z_cv_df_general_ledger_accounting
where rhcur = 'TWD'
and gjahr = '2022'
and poper = '008'
and racct = '0000141005'
and rbukrs = '6120'
```

Results

#	rbukrs	prctr	gjahr	poper	rbusa	racct	rhcur	entity
1	6120	0000002690	2022	008	OPS	0000141005	TWD	6120_OP
2	6120	0000002340	2022	008	OPS	0000141005	TWD	6120_OP
3	6120	0000002120	2022	008	OPS	0000141005	TWD	6120_OP
4	6120	0000002160	2022	008	OPS	0000141005	TWD	6120_OP
5	6120	0000002020	2022	008	OPS	0000141005	TWD	6120_OP
6	6120	0000002850	2022	008	OPS	0000141005	TWD	6120_OP
7	6120	0000002360	2022	008	OPS	0000141005	TWD	6120_OP

1 2 >

2.06 s | 31 rows returned

Refreshed 7 days ago

Microsoft Azure | databricks | Search CTRL + P | has-aa-dev-hdc-dbw | pietros@[redacted]

Data Science & Engi...

- New
- Workspace
- Repos
- Recents
- Data
- Compute
- Workflows
- Partner Connect
- Menu options

Data Explorer

Add | HDC Dev Warehouse Pro S

Data

- trf_longview__entities_hp
- trf_longview__entities_hp_time_de
- trf_longview__gl_default
- trf_longview__gl_local
- trf_longview__gl_tusdb
- trf_longview__gl_union_all
- trf_longview__lv_accounts_flatteni
- trf_longview__lv_accounts_flatteni
- trf_longview__lv_entities_flattenin
- trf_longview__lv_entities_flattenin
- trf_longview__lv_product_flattenir
- trf_longview__lv_product_flattenir
- trf_longview__lv_product_level_hie
- trf_longview__net_revenue_nrp
- trf_longview__net_revenue_nrp_tir
- trf_longview__product_flattening
- trf_longview__product_flattening_
- trf_longview__product_flattening_
- trf_longview__product_flattening_

Delta Sharing NEW

External Locations

Catalogs › hdc_dev › dbt_transform ›

hdc_dev.dbt_transform.trf_longview__lv_entities_flattening_time_dependent

View | advanalytics-svc@na.hasbro.com | Add comment | Create

Columns | Sample Data | Details | Permissions | History | Lineage | Insights New

Filter columns...

Column	Type	Comment
segment	string	
sub_segment	string	
region1	string	
region2	string	
region3	string	
entity	string	
segment_desc	string	
sub_segment_desc	string	
region1_desc	string	
region2_desc	string	
region3_desc	string	
datanode	string	

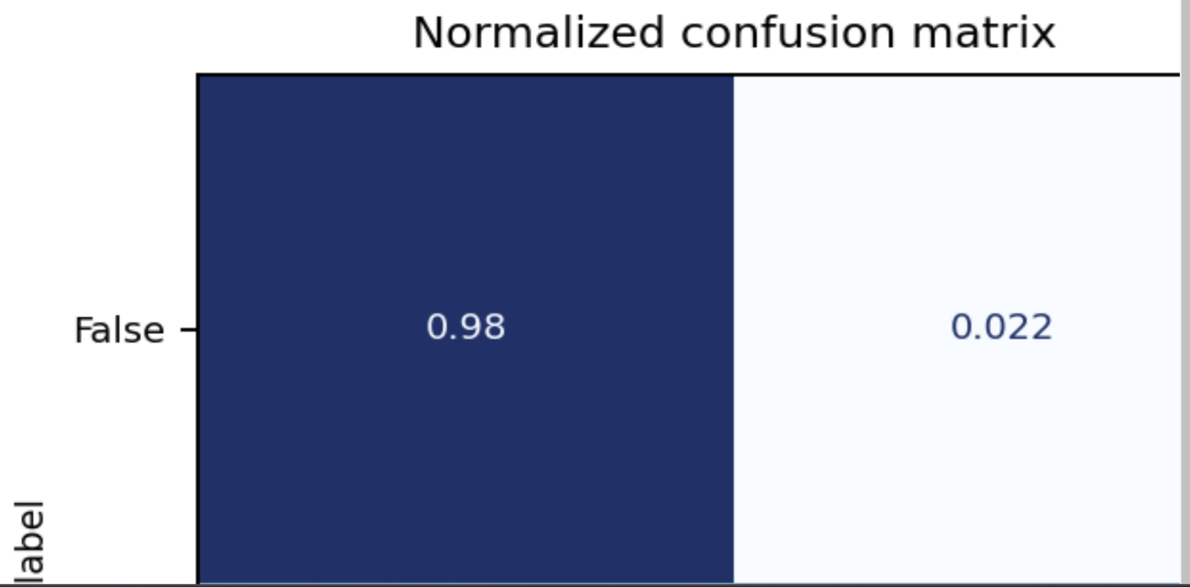
Name	Value	Actions
estimator_class	sklearn.ensemble._gb.GradientBoostingClassifier	
estimator_name	GradientBoostingClassifier	

Name | Value | Add

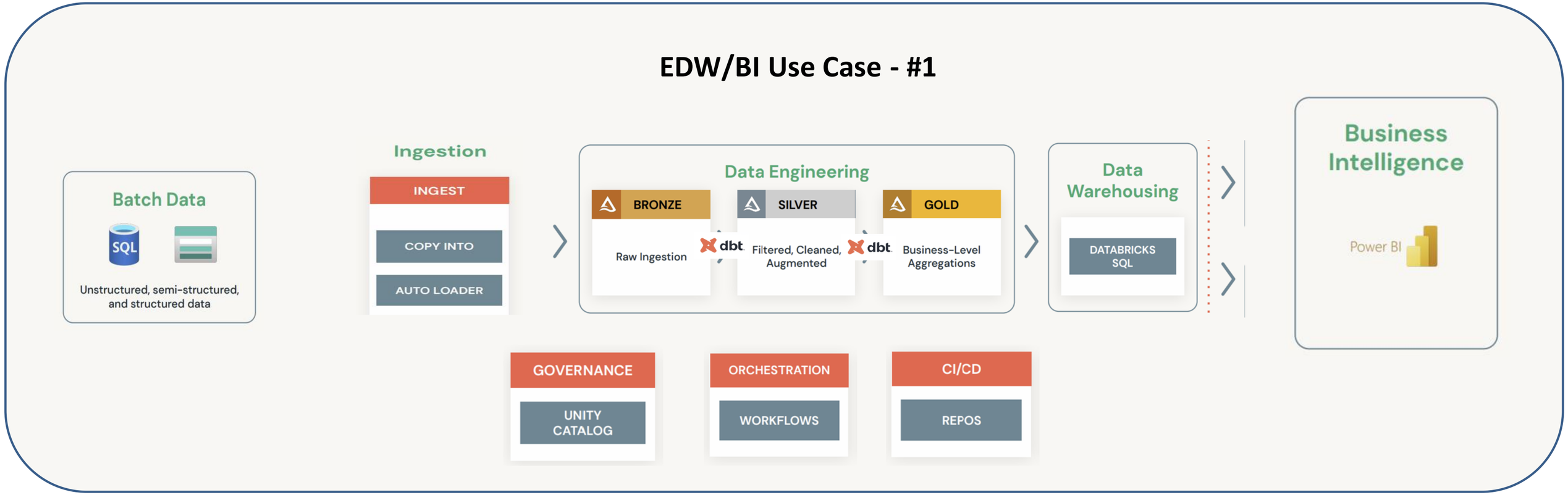
Artifacts

- model
  - MLmodel
  - conda.yaml
  - model.pkl
  - python_env.yaml
  - requirements.txt
- training_confusion_matrix.png
- training_precision_recall_curve.png
- training_roc_curve.png

Full Path:dbfs:/databricks/mlflow-tracking/1815222646294122/7e9d1e34d46346efba44022ba4ab...
Size: 23.86KB



EDW/BI Use Case - #1



AI/ML - Use Case #2