**This is just to help during my lightning talk**

**Basically, a dialog, a script.**

**\*\*Notes:  
*- Basically a delta lake is an open-source protocol for reading and writing files to cloud storage. Then delta lake, utilizes delta tables that sore data within a folder directory as parquet files, then this architecture adds delta logs stored as json files alongside the parquet files.***

***- ACID transactions refer to: Atomicity, entire transactions completes; Consistency, Data follows rules, or it will be rolled back; Isolation, one transaction completed before the other start; Durability, Data is saved in a persistent state once is completed.***

***- Lineage refers to the full journey of data from the source to the destinations. It shows the conversions or transformations or processes applied to it.***

***- a brief explanation of databricks. Databricks is a unified data analytics platform built on Apache Spark. Basically, it combines data engineering, data science, machine learning and analytics on a single platform powered by Lakehouse architecture and it exists to empower data teams to collaborate, and analyze and innovate at scale.***

**SLIDE** **1** — Presentation (10 seconds | ~20 words)

My name is Leonardo Trevizo. Today we’re going to discuss a really interesting topic- *and that is How to avoid turning your data lake into a data swamp*

**SLIDE 2** — Introduction (15 seconds | ~40 words)

This presentation focuses on how Delta Lake, an open-source storage layer developed by Databricks, helps prevent common pitfalls in data lake architecture. We'll see how it brings structure, reliability, and governance to big data environments.

**SLIDE 3** — Agenda (10 seconds | ~25 words)

Alright, here’s a quick roadmap for what we’ll be covering in the next 7 minutes.

We’ll start by looking at the problems that arise when working with traditional data lakes.

**SLIDE 4** — The reality of raw data lakes (30 seconds | ~65 words)

What problems does a data lake has? What problems is delta lake solving?

Before Delta Lake, working with raw data lakes felt like building on quicksand.

There were no transactions, no consistency, just loose files.

Updating or deleting data safely is Risky.

Schemas evolved manually, so are prone to human error.

And perhaps worst of all: no version control, it means no easy way to track or recover changes. This made data engineering unreliable and frustrating.

***SLIDE 5*** *— What is Delta Lake? (1:10 minutes | ~150 words)*

*Delta Lake solves all these issues by acting as a transactional storage layer on top of your existing data lake.*

*Now with Delta Lake, data is stored in Parquet files, and every operation is tracked in transaction logs.*

*This adds ACID compliance, version control, and schema enforcement to what was previously a chaotic environment.*

*You can now safely perform inserts, updates, and deletes and merge.*

*You can even time travel across table versions.*

*Schema changes are now controlled and validated.*

*Performance also improves through features like file compaction and caching and vacuum.*

*What this means for data engineers is that your data lake stops being a passive storage dump and becomes an active, queryable, trustworthy platform.*

*In short, Delta Lake brings structure, safety, and speed to data lakes without sacrificing in flexibility or cost.*

**SLIDE 6** — Where does Delta Lake sit? + Unity Catalog relation (1:00 minute | ~130 words)

Where does Delta Lake sit? You may feel overwhelmed by this table, but the only purpose is compare with other technologies, and to know where Delta Lake sits.

Delta Lake sits right between the raw flexibility of a data lake and the structured reliability of a data warehouse.

You get versioning and ACID transactions like a warehouse, but maintain the scalability and cost benefits of a data lake.

This middle layer makes it ideal for modern analytics pipelines.

And here’s where Unity Catalog -the main idea for my last presentation- comes in.

If Delta Lake gives your data consistency and performance, Unity Catalog adds governance and security.

It manages metadata across all Delta tables, enforces access policies, and tracks lineage.

Together, Delta Lake and Unity Catalog create a trusted data foundation.

You gain both control and discoverability, something essential for collaboration and compliance.

The result is a true data Lakehouse architecture that combines the best of both worlds.

**SLIDE 7** — How a data engineer uses Delta Lake features (1:05 minutes | ~140 words)

Let’s look at how Delta Lake works in action.

In this example, we’re merging new orders into our orders\_2025 table using SQL.

Delta Lake lets us do this with one command that updates existing rows and inserts new ones.

You can see the results below: 55 rows affected in total—20 updated, 35 inserted, zero deleted.

All of this is transactionally safe. And if anything had gone wrong, we could roll back.

This isn’t just a data modification—it’s a controlled, trackable operation that fits perfectly into modern CI/CD pipelines for data.

This level of precision and traceability would be very difficult to implement in a traditional data lake.

Delta Lake makes these operations intuitive and reliable—critical for data engineers responsible for production pipelines.

***SLIDE 8*** *— Describe commands (1:05 minutes | ~140 words)*

*But what about visibility?*

*Delta Lake gives us powerful metadata tools through commands like DESCRIBE HISTORY.*

*This example shows the full version history of the orders\_2025 table.*

*Each operation, whether it is a CREATE, WRITE, or a MERGE command, is logged with a timestamp, user, and execution details.*

*You can see exactly who changed what and when.*

*This is especially helpful for debugging and auditing. Basically, with one command, you gain total clarity over your data’s evolution.*

*And it’s all native to Delta Lake.*

**SLIDE 9** — Versioning (20 seconds | ~45 words)

Here we’re looking at how Delta Lake lets you query data at a specific point in time.

Here, we’re selecting from version 1 of our orders table. But you cannot only query previous versions, you can revert changes as well.

This makes data reproducibility and rollback not just possible, but effortless.

***SLIDE 10*** *— Real benefits + Conclusion (45 seconds | ~100 words)*

*So why does all of this matter to us as data engineers?*

*With Delta Lake, we create reliable tables that downstream teams can trust.*

*No more costly “overwrite everything” jobs.*

*Our datasets become auditable and reproducible.*

*Collaboration becomes smoother because teams are working on consistent data with clear history.*

*Delta Lake doesn’t just solve technical problems—it improves how teams work with data.*

*And when combined with Unity Catalog, it creates a governed, secure environment that scales with your business.*

*In conclusion: Delta Lake is not just a layer—it’s a foundation for building modern data platforms.*

**SLIDE 11** — Acknowledgements (10 seconds | ~20 words)

Thank you all for your time and attention.

Special thanks to my instructors, mentors and colleagues, for their support and the feedback.

• • • •