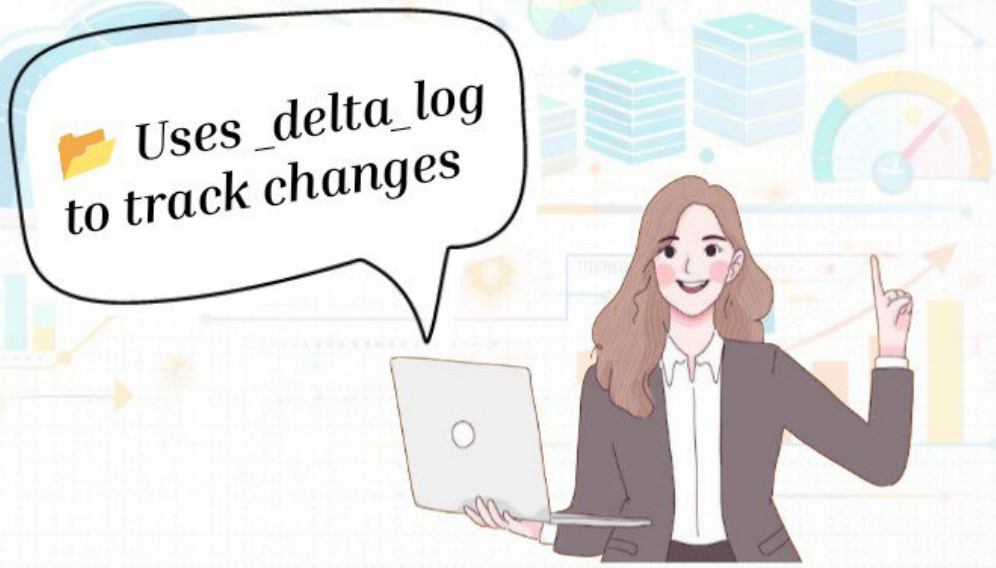
Day 4 – Delta Lake Introduction

What is Delta Lake?

- *Open-source storage layer*
- *Built on top of Parquet*
- *Adds reliability to data lakes*

🧠 *Key Idea*

Delta Lake = Parquet + Transactions + Metadata



Why Delta Lake?

❌ Problems with Traditional Data Lakes

- No transactions
- Duplicate data

Schema issues

- Hard to update/delete data

✅ Delta Lake Solves This

- ACID transactions
- Schema enforcement
- Updates & deletes
- Time travel support

ACID Transactions

🔐 What is ACID?

- Atomicity – All or nothing
- Consistency – Valid data only
- Isolation – Safe concurrent operations
- Durability – Data is permanent

✅ In Delta Lake

- No partial writes
- Safe concurrent reads & writes

Schema Enforcement

🛡️ Data Protection Feature

- Delta Lake enforces table schema
- Invalid data → insert fails

✅ Benefits

- Prevents bad data
- Ensures data quality

Avoids silent errors

Schema Enforcement

What happens?

If your Delta table schema is:

```
id INT, name STRING
```

And you try to insert:

```
id STRING, name STRING
```

🚫 *Insert fails*

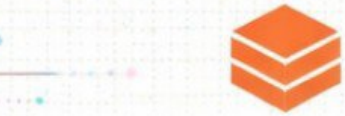
Why this is good?

- *Prevents bad data*
- *Avoids silent data issues*
- *Keeps tables clean*

Delta Lake vs Parquet

Feature	Parquet	Delta Lake
Storage format	✅	✅
ACID transactions	❌	✅
Schema enforcement	❌	✅
Updates & Deletes	❌	✅
Time travel	❌	✅

👉 *Delta Lake = Parquet + reliability*



✨ Takeaways

- *Delta Lake makes data lakes production-ready*
- *ACID transactions are critical for reliability*
- *Schema enforcement protects data quality*
- *Delta is far more powerful than plain Parquet*