databricks

Delta Lake Features

databricks

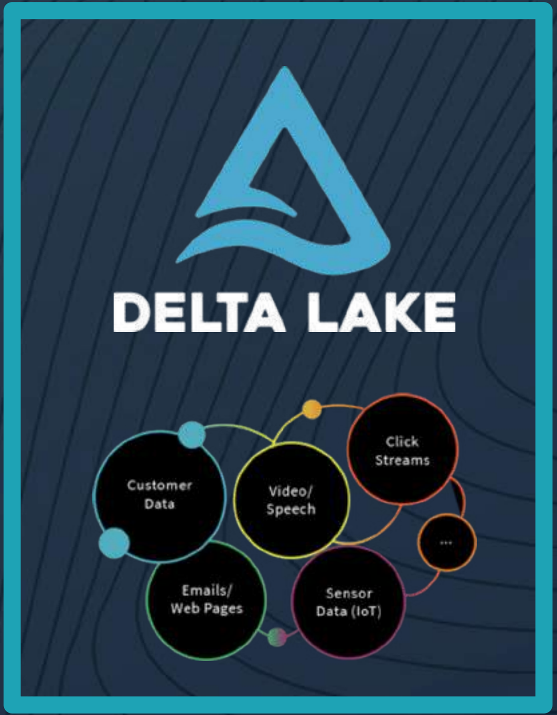
Delta Lake

Open Format Based on Parquet

With Transactions

Apache Spark APIs

Delta Lake ready for Analytics



Reliability

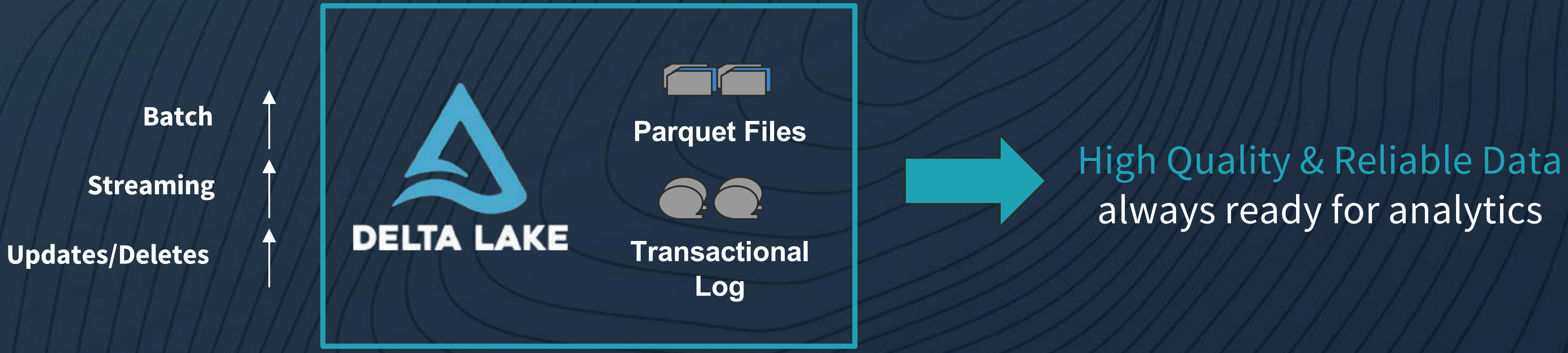
→

Performance

Data Science & ML

- Recommendation Engines
- Risk, Fraud Detection
- IoT & Predictive Maintenance
- Genomics & DNA Sequencing

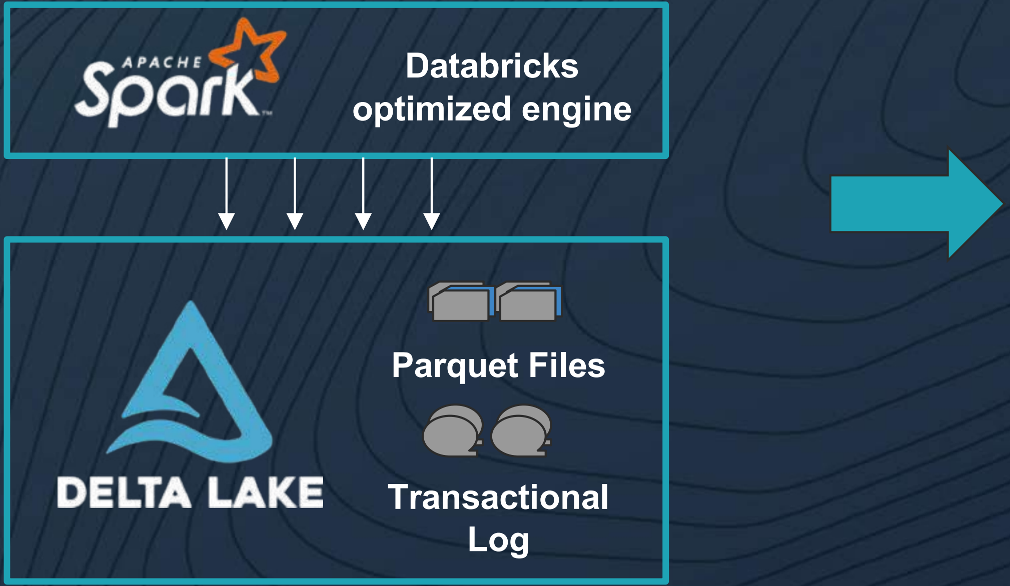
Delta Lake ensures data reliability



Key Features

- ACID Transactions
- Schema Enforcement
- Unified Batch & Streaming
- Time Travel/Data Snapshots

Delta Lake optimizes performance



Highly Performant queries at scale

Key Features

- Indexing
- Compaction
- Data skipping
- Caching

Get started with Delta using Spark APIs

Instead of **parquet**...

```
CREATE TABLE ...
USING parquet
...

dataframe
     .write
     .format("parquet")
     .save("/data")
```

... simply say **delta**

```
CREATE TABLE ...
USING delta
…

dataframe
     .write
     .format("delta")
     .save("/data")
```

Use Delta with Existing Parquet Tables

Step 1: Convert `Parquet` to `Delta` Tables

```
CONVERT TO DELTA parquet.`path/to/table` [NO STATISTICS]
[PARTITIONED BY (col_name1 col_type1, col_name2 col_type2, ...)]
```

Step 2: Optimize Layout for Fast Queries

```
OPTIMIZE events
WHERE date >= current_timestamp() - INTERVAL 1 day
ZORDER BY (eventType)
```

Upsert/Merge fine-grained Updates

```
MERGE INTO customers -- Delta table
USING updates
ON customers.customerId = source.customerId
WHEN MATCHED THEN
    UPDATE SET address = updates.address
WHEN NOT MATCHED
    THEN INSERT (customerId, address) VALUES (updates.customerId,
updates.address)
```

Time Travel

Reproduce experiments & reports

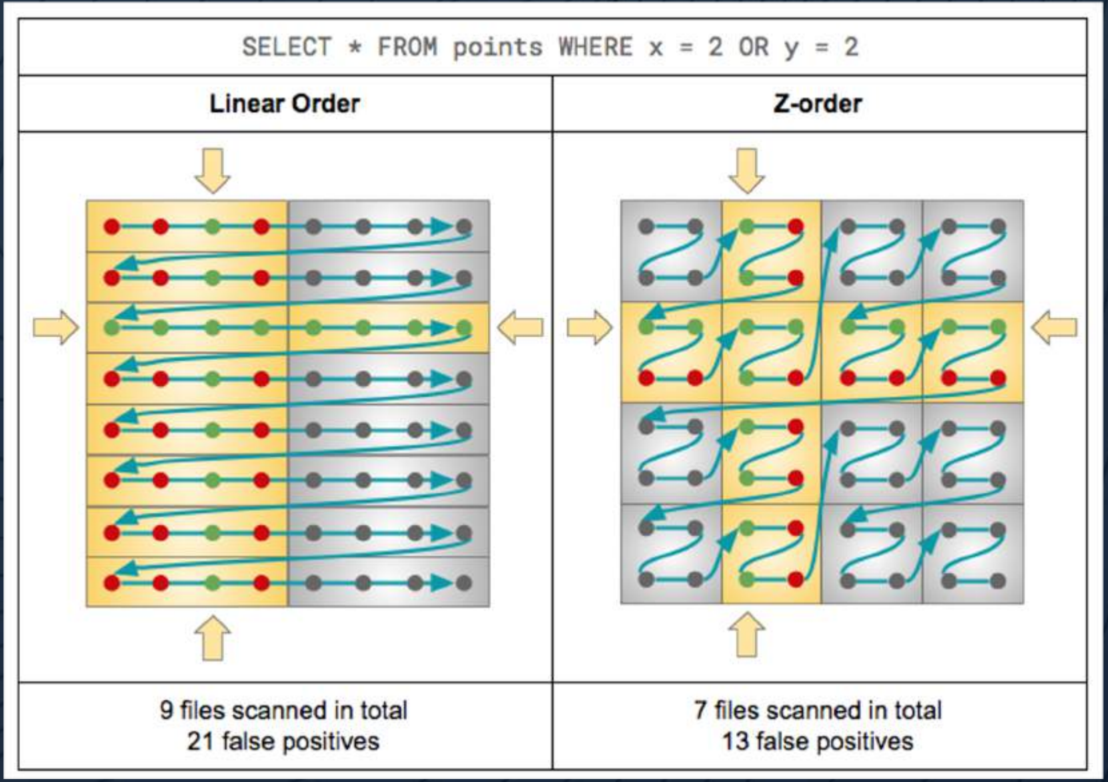
```
SELECT count(*) FROM events
TIMESTAMP AS OF timestamp

SELECT count(*) FROM events
VERSION AS OF version

spark.read.format("delta").option("timestampAsOf",
timestamp_string).load("/events/")
```

Rollback accidental bad writes

```
INSERT INTO my_table
    SELECT * FROM my_table TIMESTAMP AS OF
    date_sub(current_date(), 1)
```

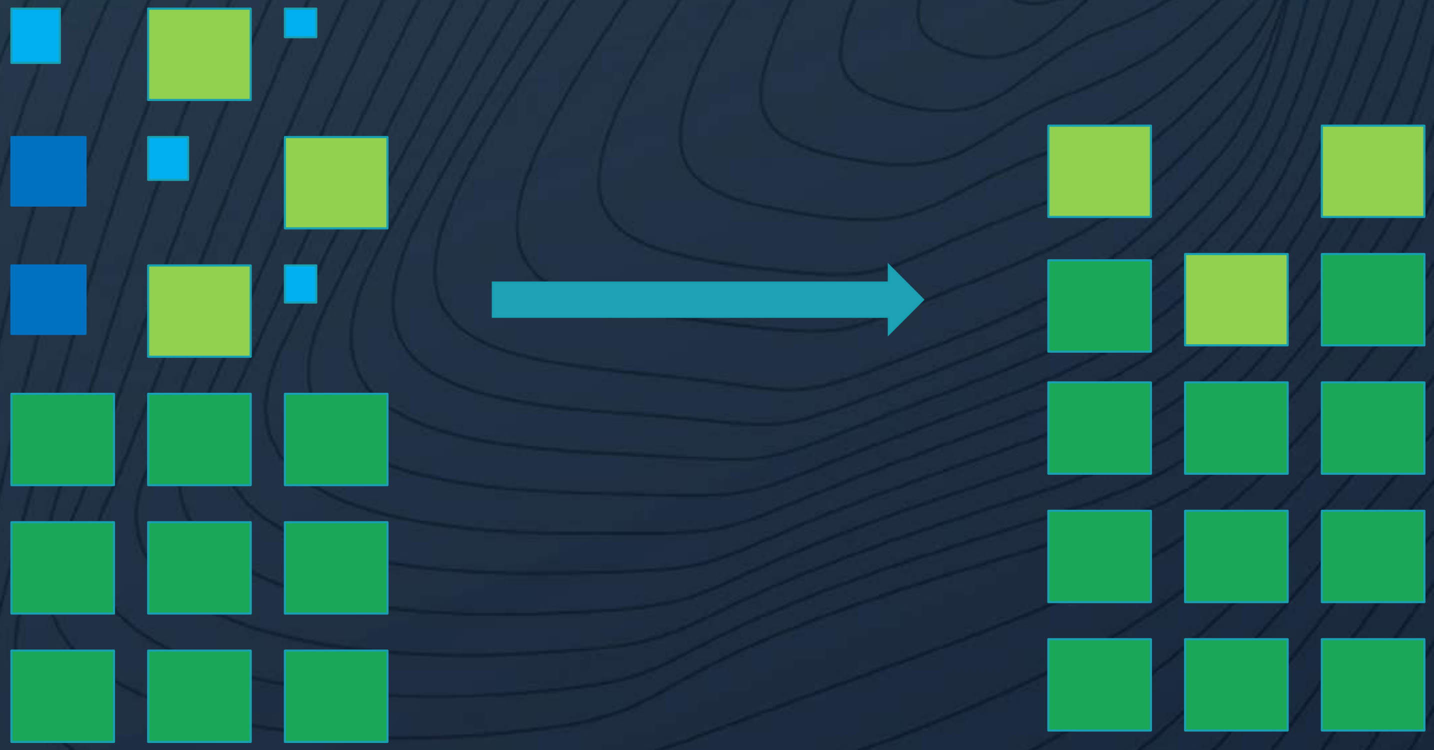
Optimizing data layout - Z-Ordering

```
OPTIMIZE events
WHERE date >=
    current_timestamp() -
    INTERVAL 1 day
ZORDER BY (eventType)
```



How OPTIMIZE works

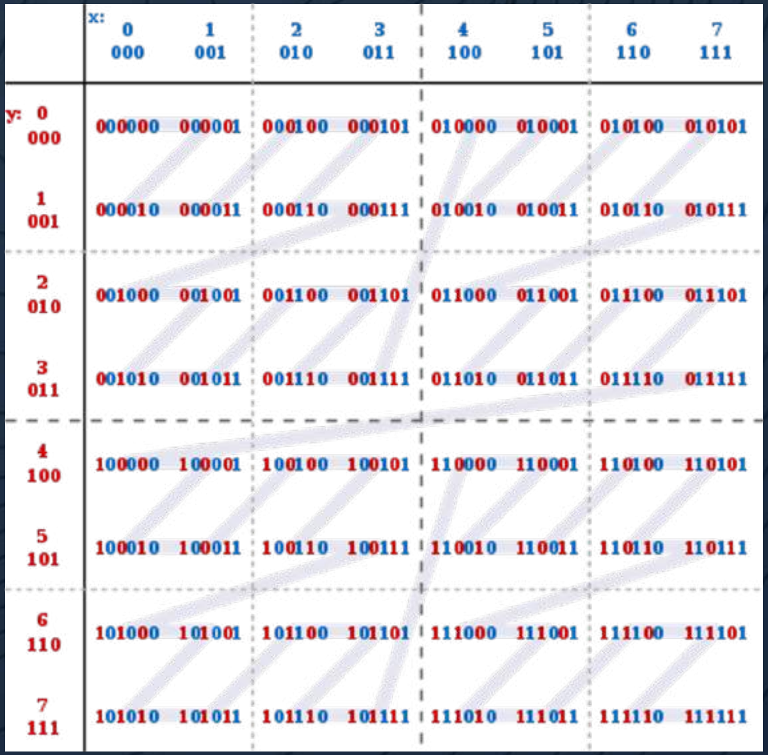
OPTIMIZE my_table



How ZORDER BY works

OPTIMIZE my_table ZORDER BY (col1, col2)

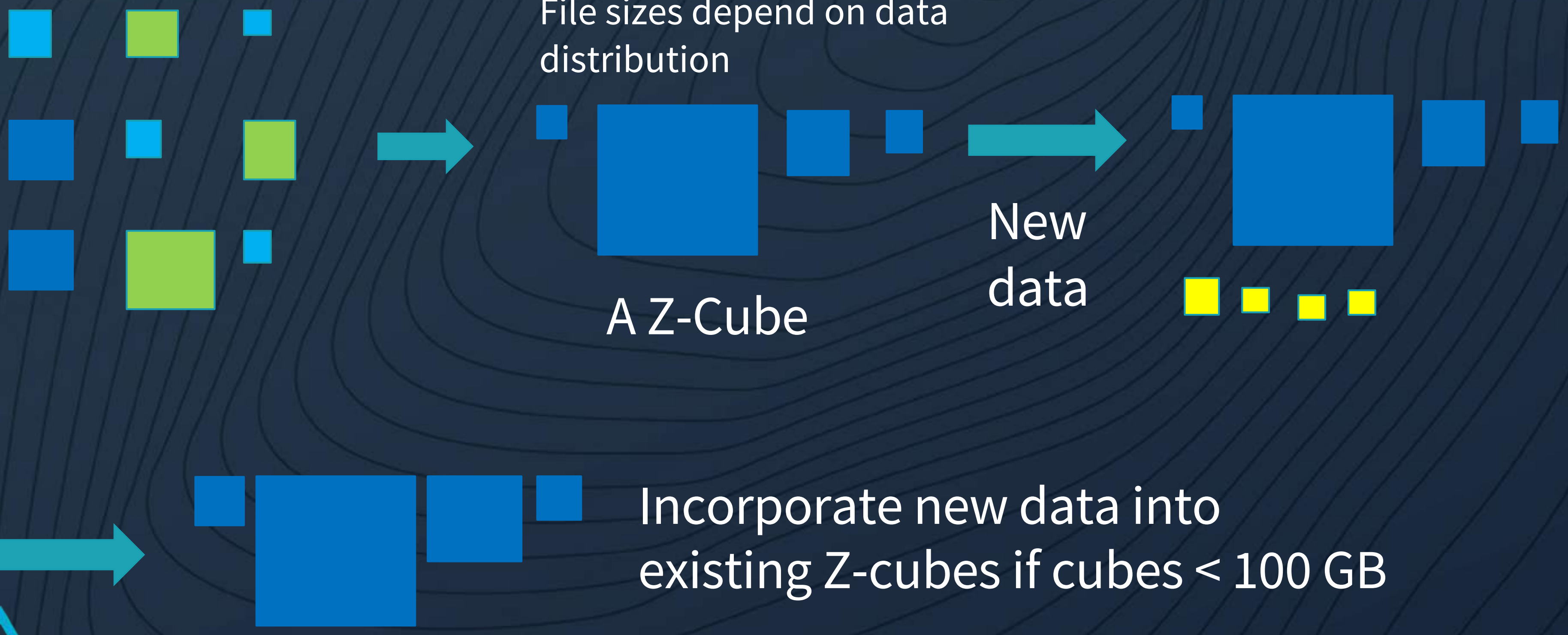
- Range partitions data
- Each box is a separate file



y: \ x:	0 000	1 001	2 010	3 011	4 100	5 101	6 110	7 111
0 000	000000	000001	000100	000101	010000	010001	010100	010101
1 001	000010	000011	000110	000111	010010	010011	010110	010111
2 010	001000	001001	001100	001101	011000	011001	011100	011101
3 011	001010	001011	001110	001111	011010	011011	011110	011111
4 100	100000	100001	100100	100101	110000	110001	110100	110101
5 101	100010	100011	100110	100111	110010	110011	110110	110111
6 110	101000	101001	101100	101101	111000	111001	111100	111101
7 111	101010	101011	101110	101111	111010	111011	111110	111111

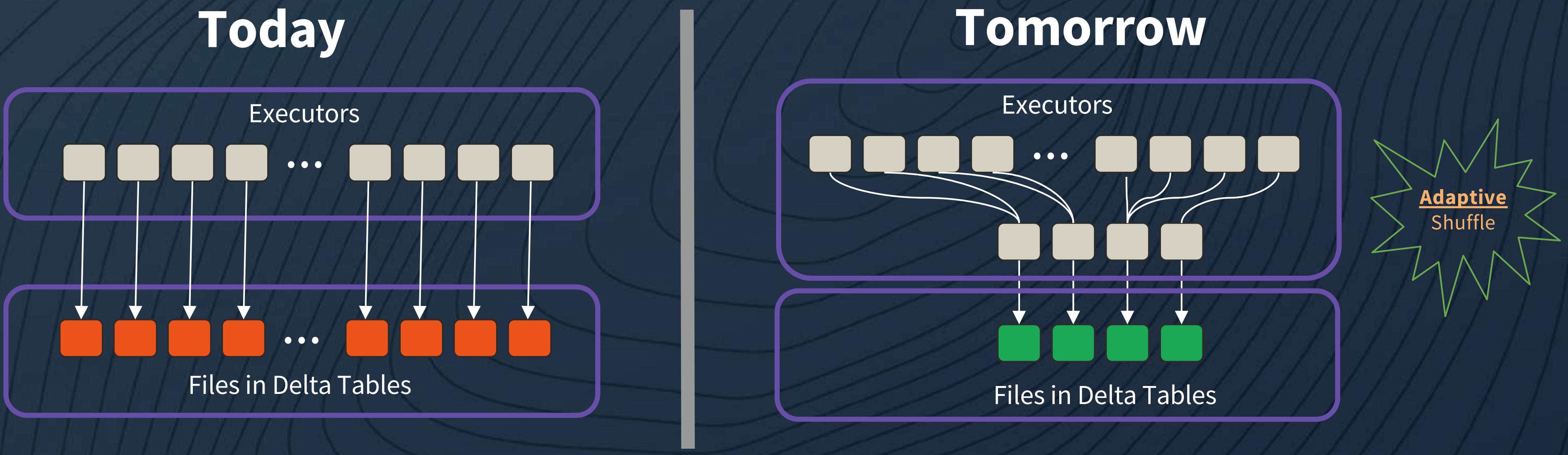
How Incremental ZORDER BY works

OPTIMIZE my_table ZORDER BY (col1, col2)



Incorporate new data into existing Z-cubes if cubes < 100 GB

Auto Optimize of Delta Tables

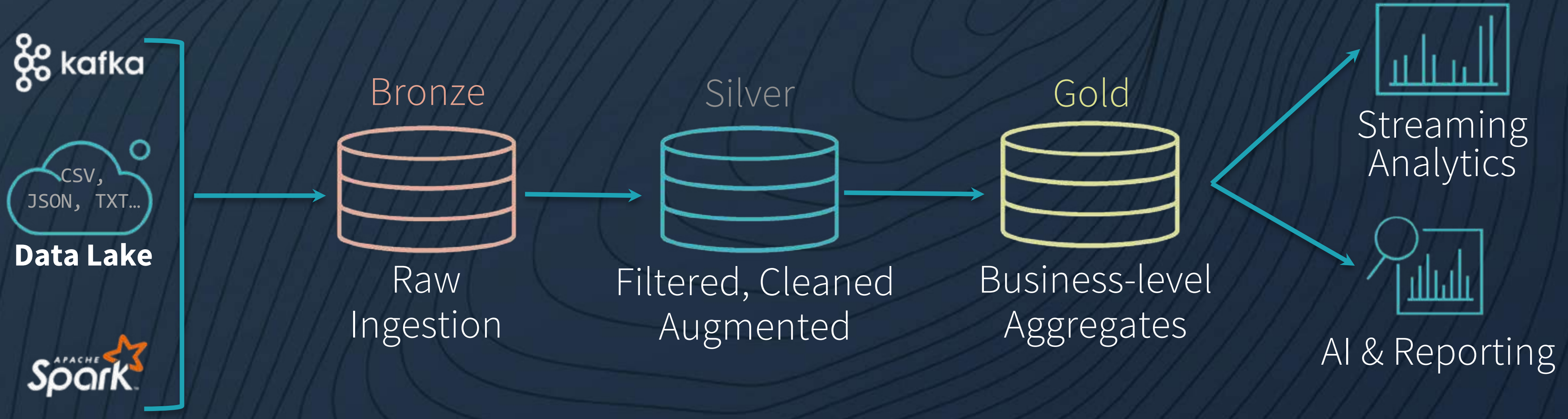


Efficient Writes
(Avoid IO failures due to many file writes)

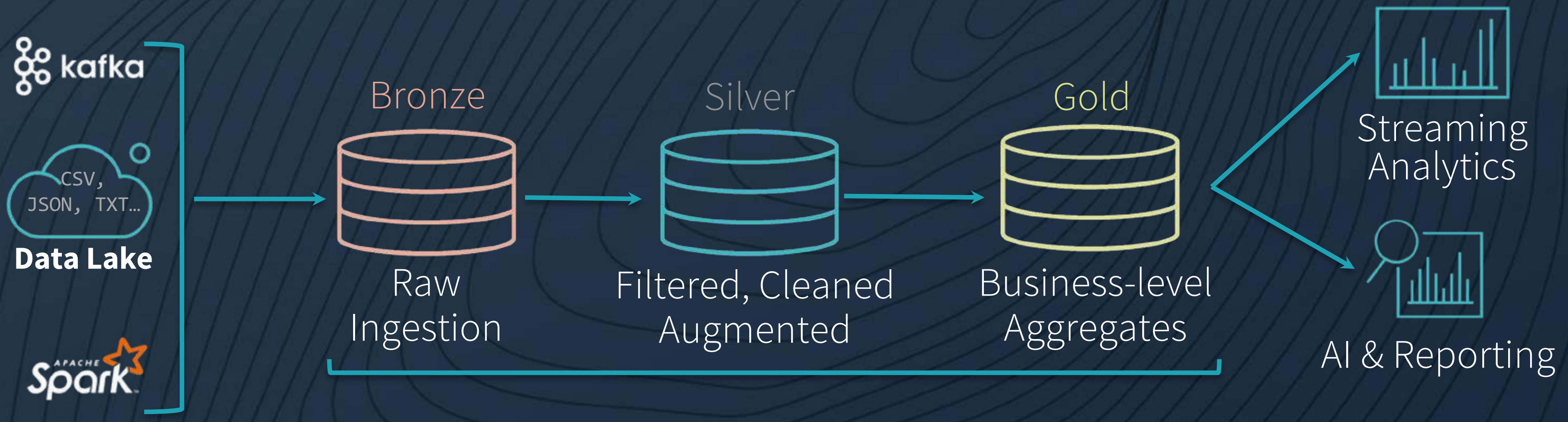
Fast Reads

Save Cloud Costs
(Avoid listing large # of files)

The DELTA LAKE

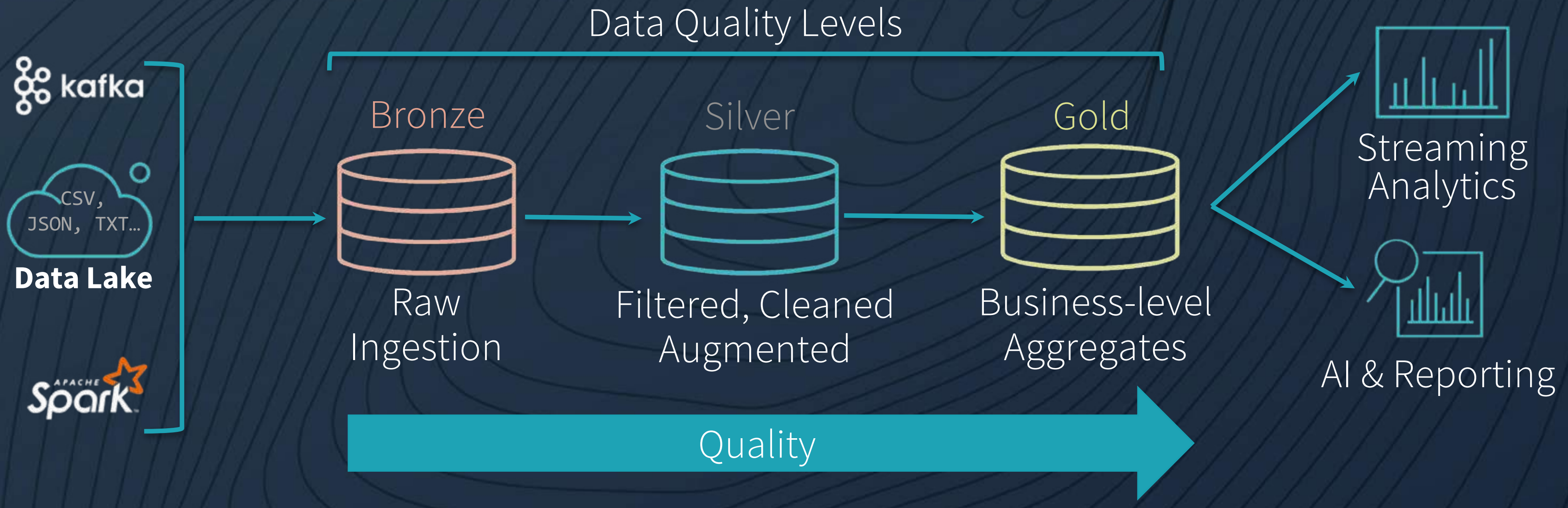


The DELTA LAKE

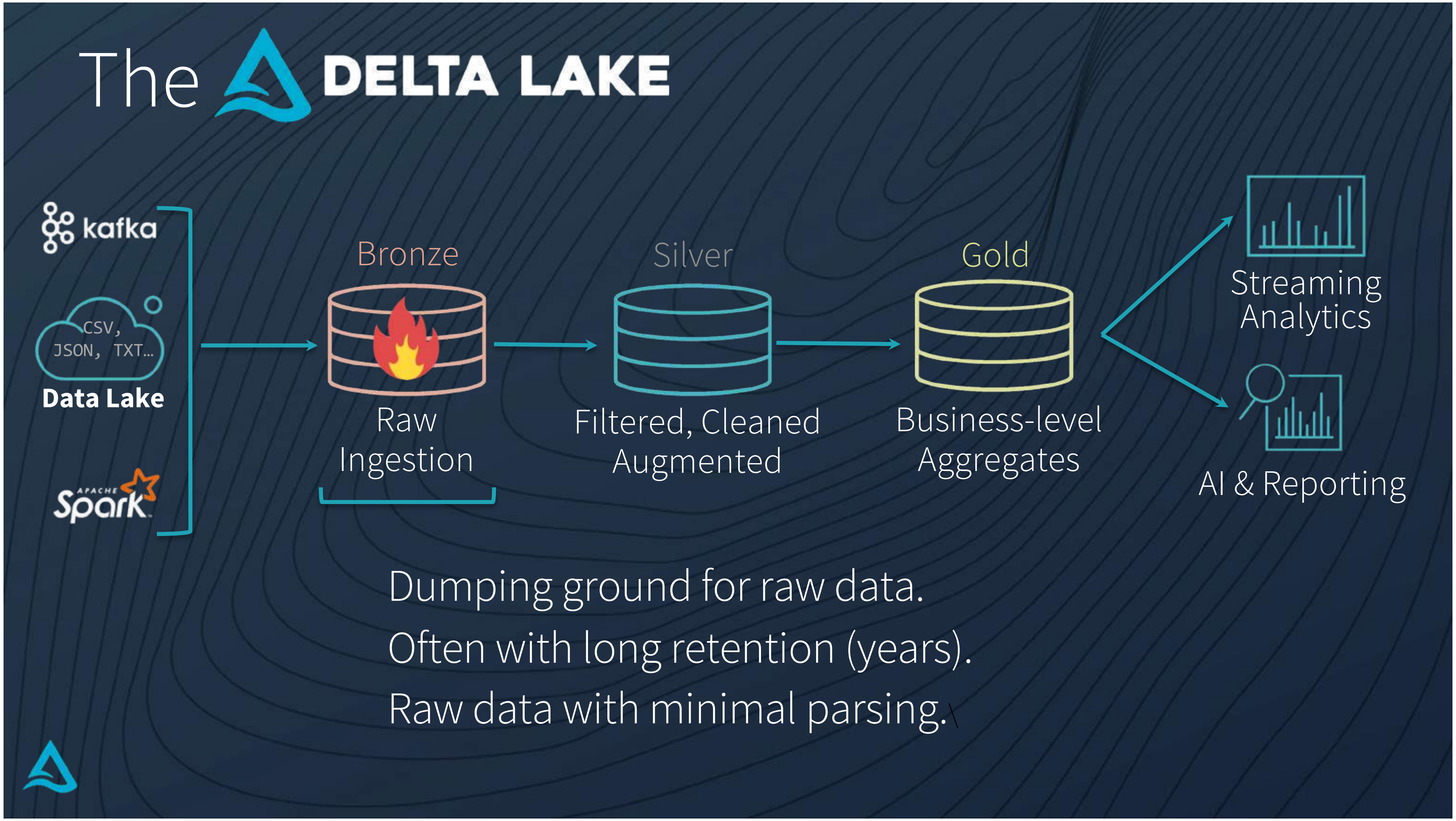


- Full ACID Transactions
- Open Source (Apache License)
- Powered by Apache Spark

The DELTA LAKE



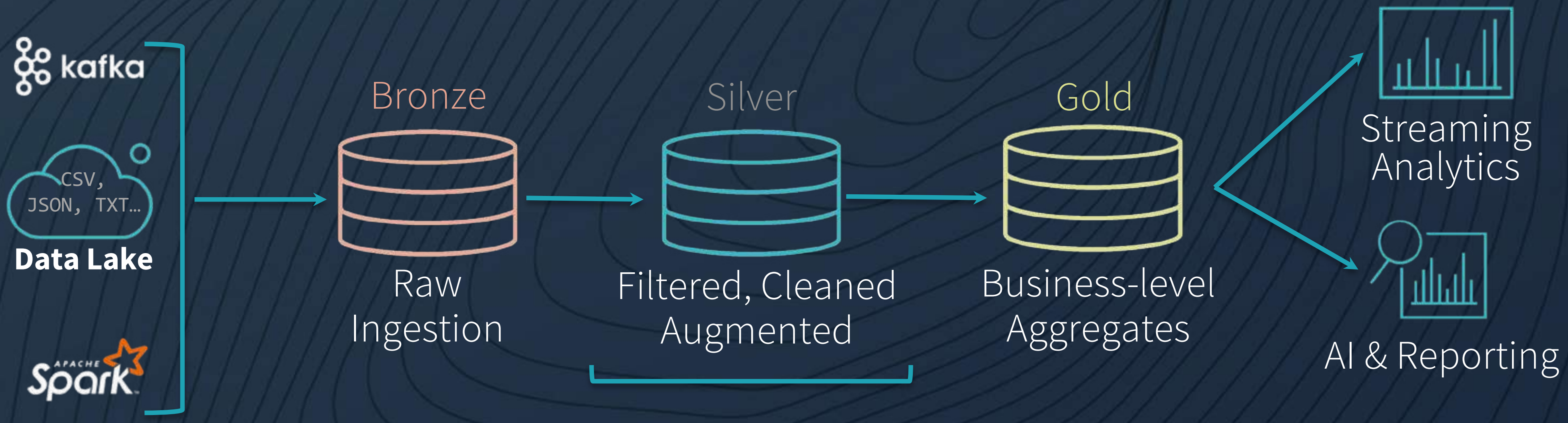
Delta Lake allows you to *incrementally* improve the quality of your data until it is ready for consumption.



The DELTA LAKE

Dumping ground for raw data.
Often with long retention (years).
Raw data with minimal parsing.

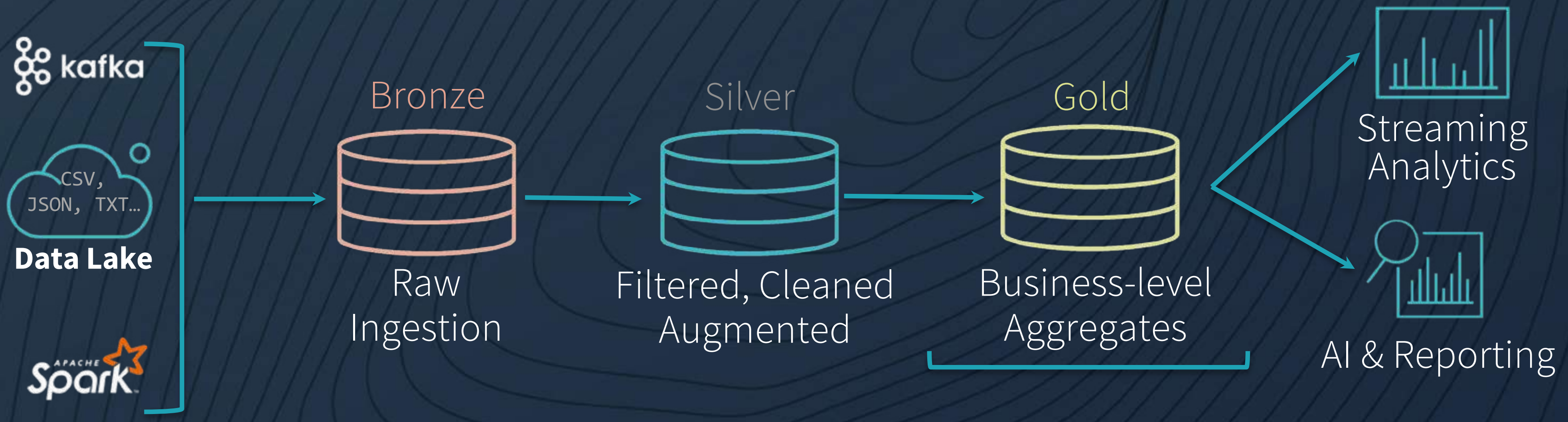
The DELTA LAKE



Intermediate data with some cleanup applied.

Queryable for easy debugging!

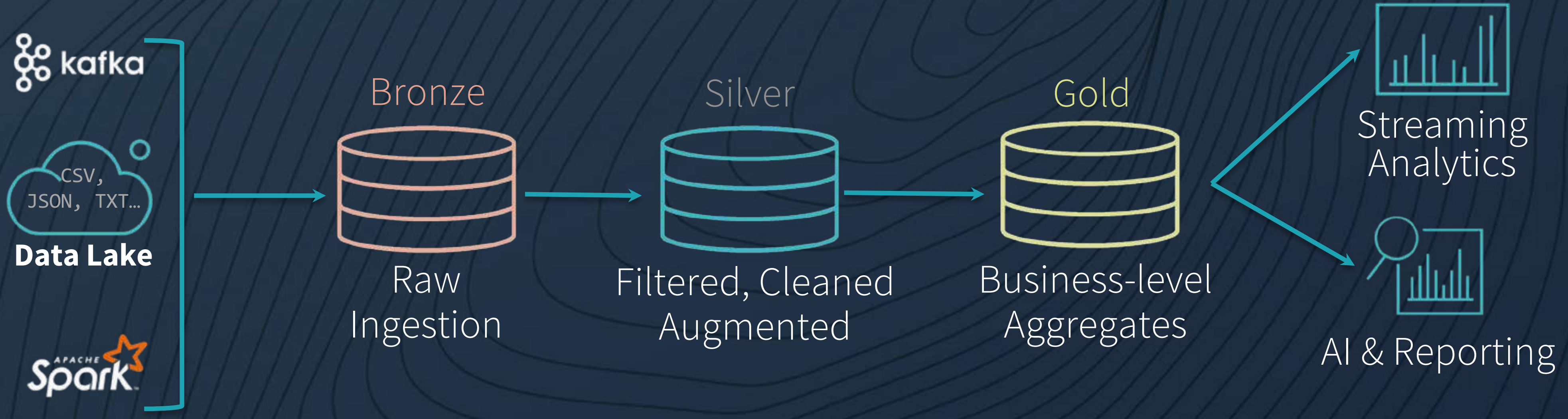
The DELTA LAKE



Clean data, ready for consumption.

Read with Spark or Presto.

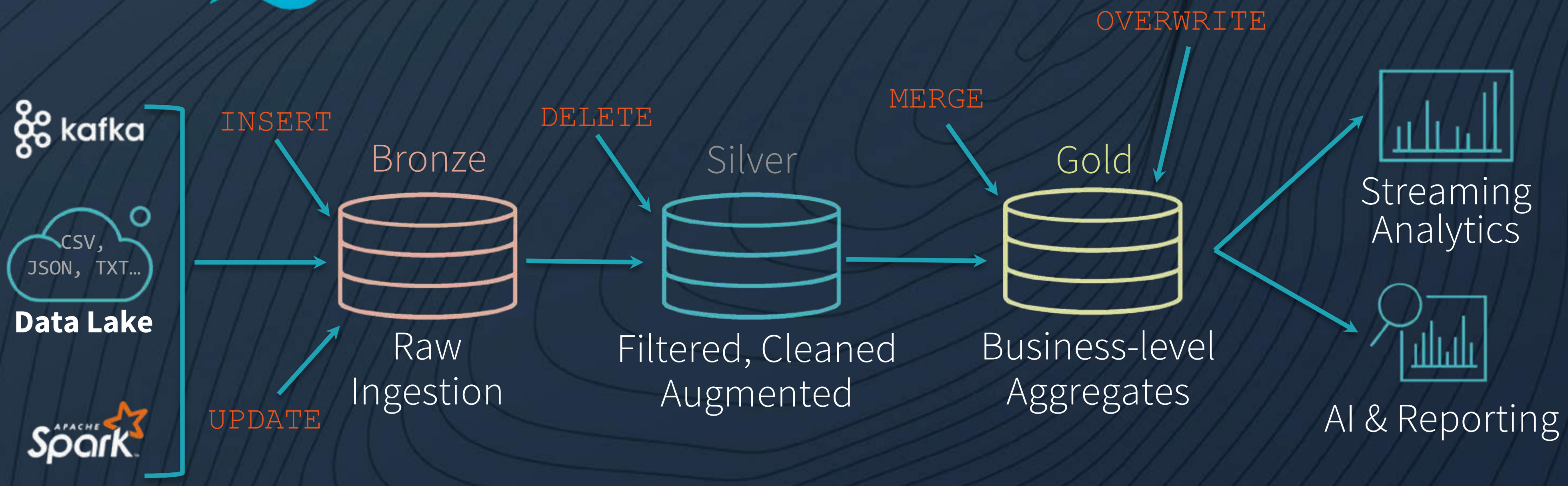
The DELTA LAKE



Streams move data through the Delta Lake

- Low-latency or manually triggered
- Eliminates management of schedules and jobs

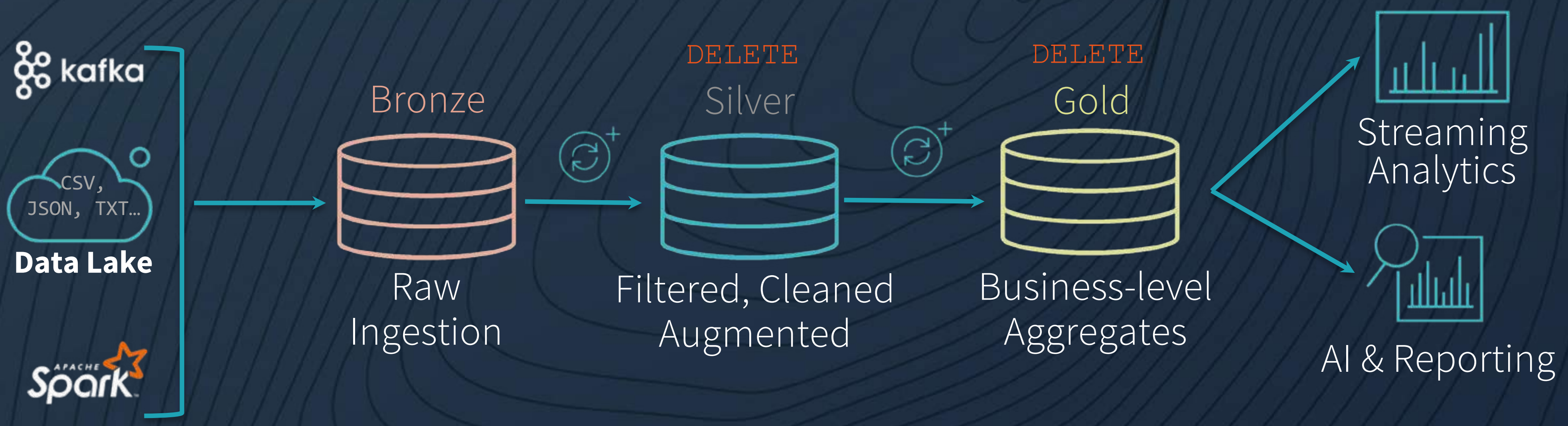
The DELTA LAKE



Delta Lake also supports batch jobs and standard DML

- Retention
- Corrections
- GDPR
- UPSERTS

The DELTA LAKE



Easy to recompute when business logic changes

- Clear tables
- Restart streams