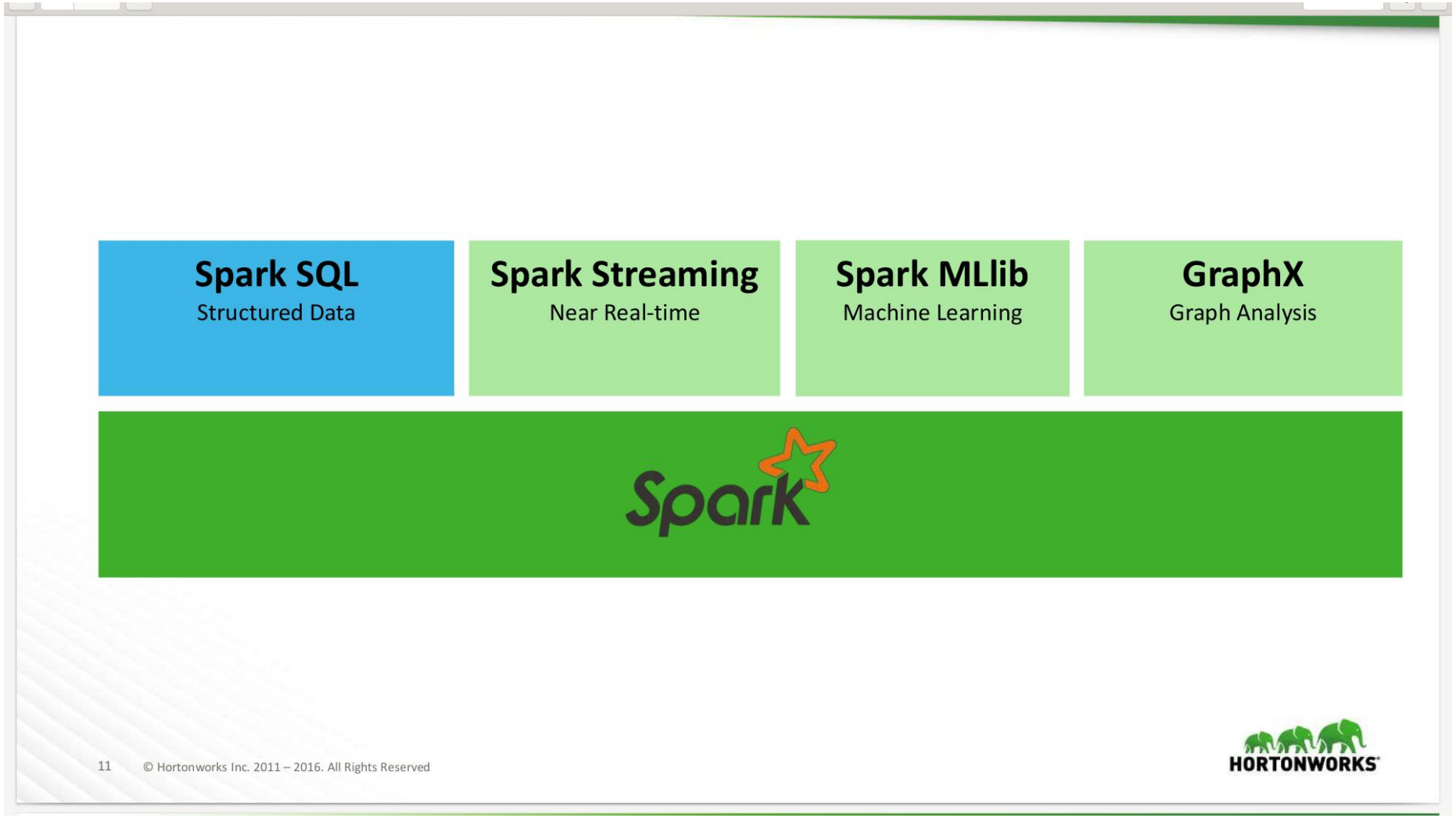
Spark Core Concepts

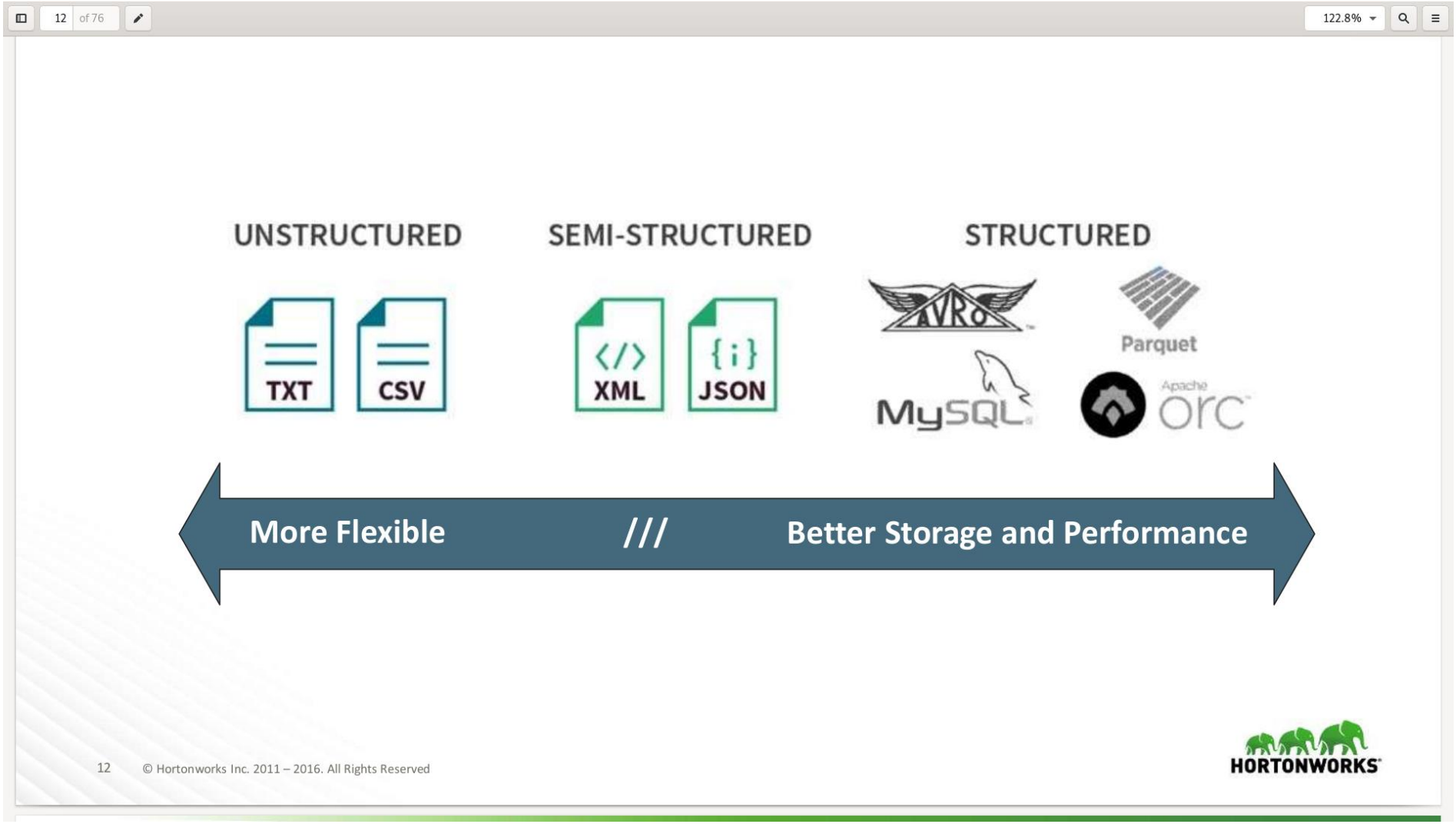
"Slide Smoothie"

- https://www.slideshare.net/Hadoop_Summit/apache-spark-crash-course-98521732
- https://www.slideshare.net/databricks/improving-spark-sql-at-linkedin
- https://www.slideshare.net/databricks/jumpstart-on-apache-spark-22-on-databricks
- https://www.slideshare.net/databricks/introducing-databricks-delta
- https://www.slideshare.net/BrandonBerlinrut/chug-building-a-data-lake-in-azure-with-spark-and-databricks
- https://www.slideshare.net/sawjd/data-con-la-2019-big-data-modeling-with-spark-sql-make-data-valuable-by-jayesh-patel

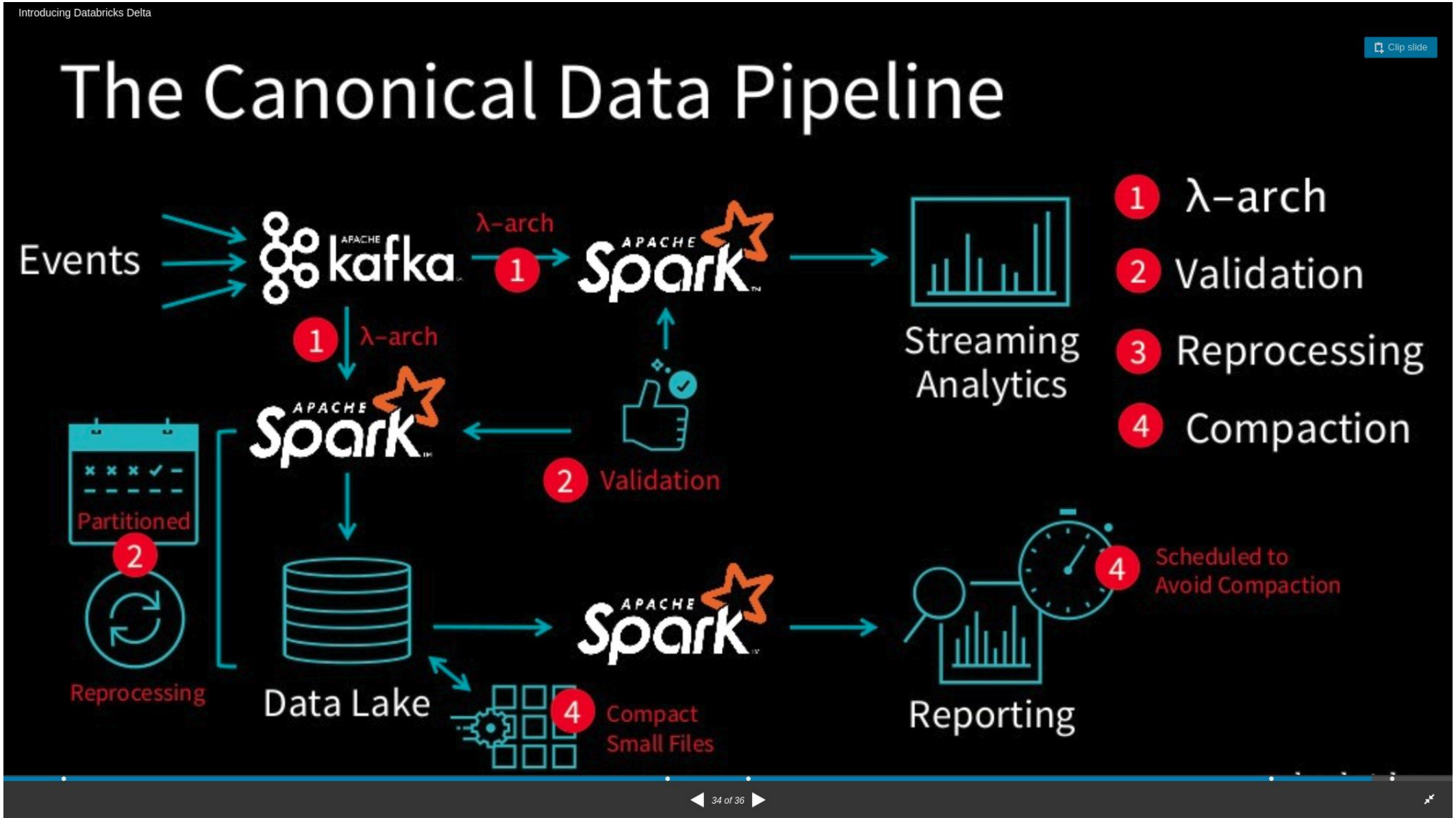
Spark family picture, SQL is king for IOC



Source(s) of data



Databricks Delta Storage



Layered data distribution

Creating Layers in the Lake

Clip slide

Raw ("Bronze")

- Stores all source data in its raw format.
- Is always append-only.
- Allows users to rapidly access data and prototype new pipelines/queries/models.
- Often uses lifecycle management and cold storage for cost savings.

Integrated ("Silver")

- Data is lightly processed to a standardized format and convention.
- Data types can be applied.
- File formats/compression applied (ORC, Parquet, etc.).
- Can be append-only or potentially have other load strategies.

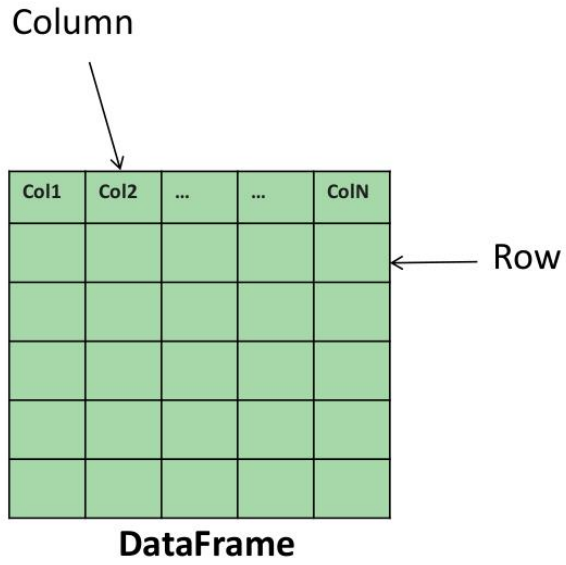
Curated ("Gold")

- Data must be manually modeled and curated for use.
- May resemble a star or snowflake schema with heavily normalized data.
- Can feed data into an EDW or potentially replace an EDW all together.
- May contain the output of AI/ML models.

Spark workhorse: Distributed Dataframes

DataFrames

- **Distributed collection** of **data organized** into ***named columns***
- Conceptually equivalent to a table in **relational DB** or a **data frame in R/Python**
- API available in Scala, Java, Python, and R



Data is described as a DataFrame with **rows, columns,** and a **schema**

HORTONWORKS

DataFrames in Spark

- Distributed collection of data grouped into named columns (i.e. RDD with schema)
- Domain-specific functions designed for common tasks
  - Metadata
  - Sampling
  - Project, filter, aggregation, join, …
  - UDFs
- Available in Python, Scala, Java, and R (via SparkR)

Spark SQL

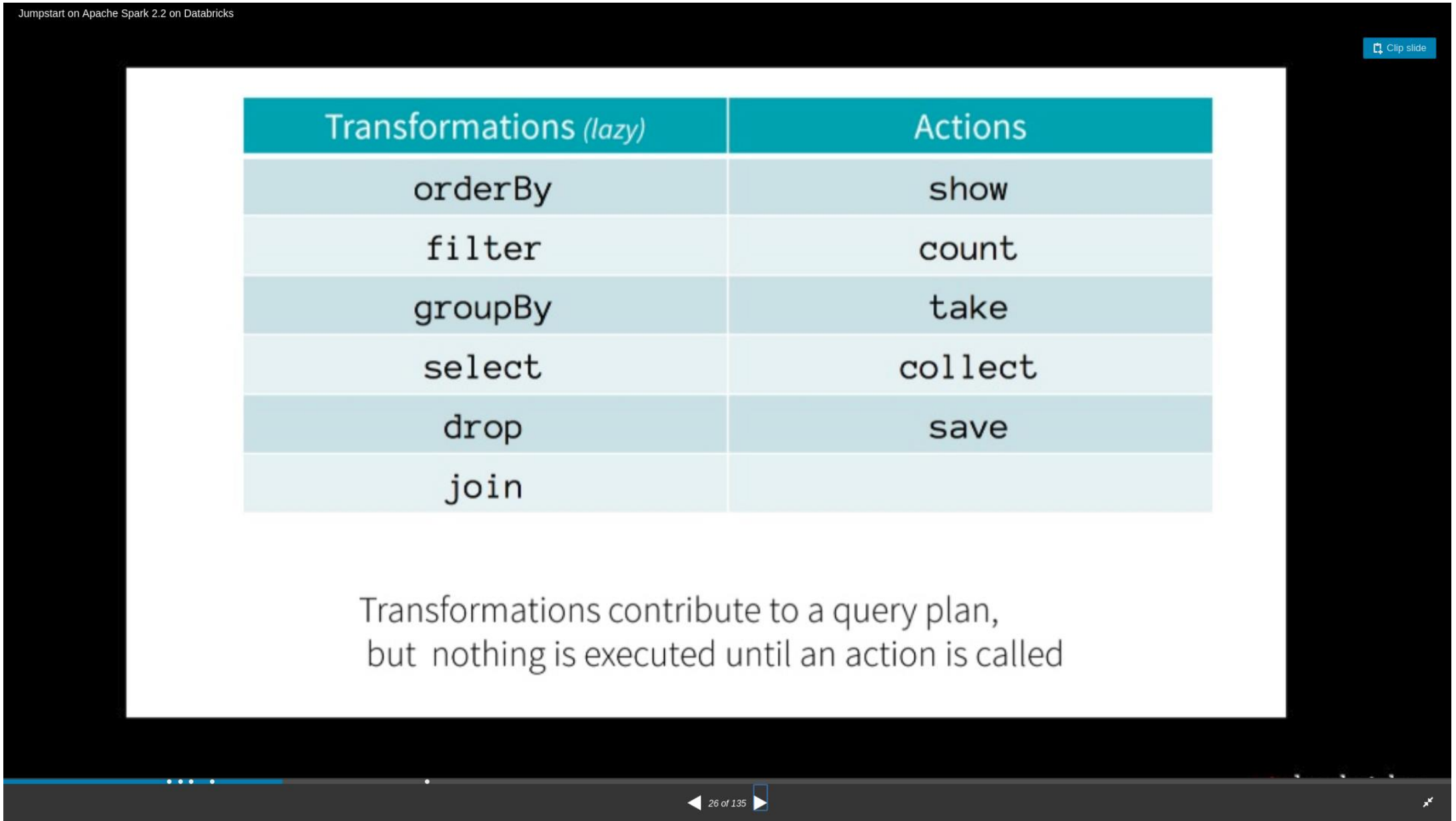
- Supports HiveQL and SQL.
- Offers standard functions, aggregation and window functions for Dataframes

```
display(df.groupBy('protocol_type')
          .count()
          .orderBy('count', ascending=False))
```

```
protocols = sqlContext.sql("""
                           SELECT protocol_type, count(*) as freq
                           FROM connections
                           GROUP BY protocol_type
                           ORDER BY 2 DESC
                           """)
display(protocols)
```

SQL 2003

Spark is lazy

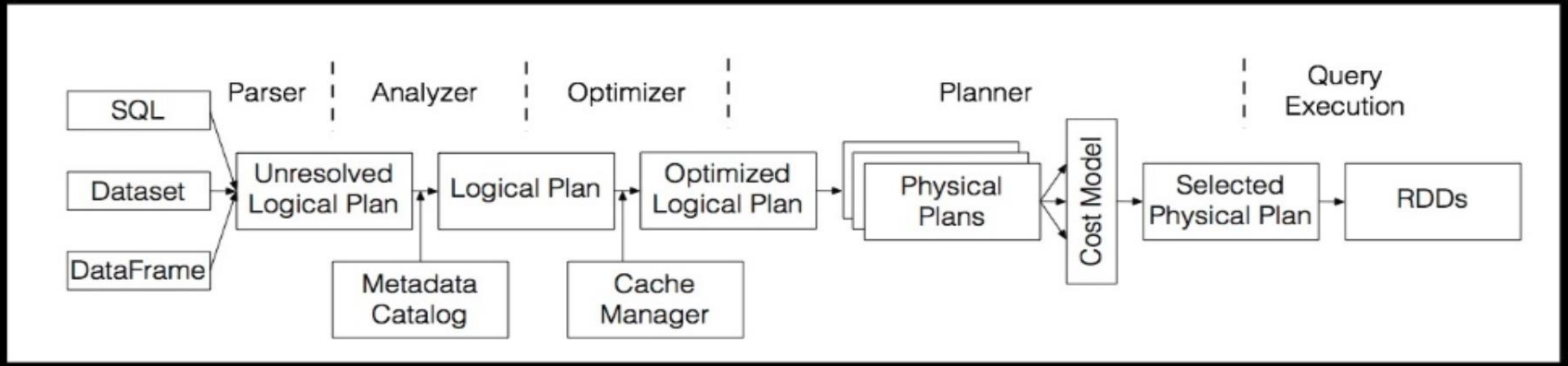


Deep Dive into Spark SQL with Advanced Performance Tuning with Xiao Li & Wenchen Fan

Clip slide

Spark SQL

A compiler from queries to RDDs.



https://www.slideshare.net/databricks/deep-dive-into-spark-sql-with-advanced-performance-tuning-with-xiao-li-wenchen-fan

How to use those tables? Well…