

Lesson 3.2: Spark as a Connector

DISTRIBUTED COMPUTING WITH SPARK SQL

Spark as a Connector



Conor Murphy
Data Scientist
Databricks

UC DAVIS
Continuing and Professional Education

Slide 2: Welcome Back!



Welcome Back!

Data architecture and design principles

Slide 3: Learning Objectives



Learning Objectives

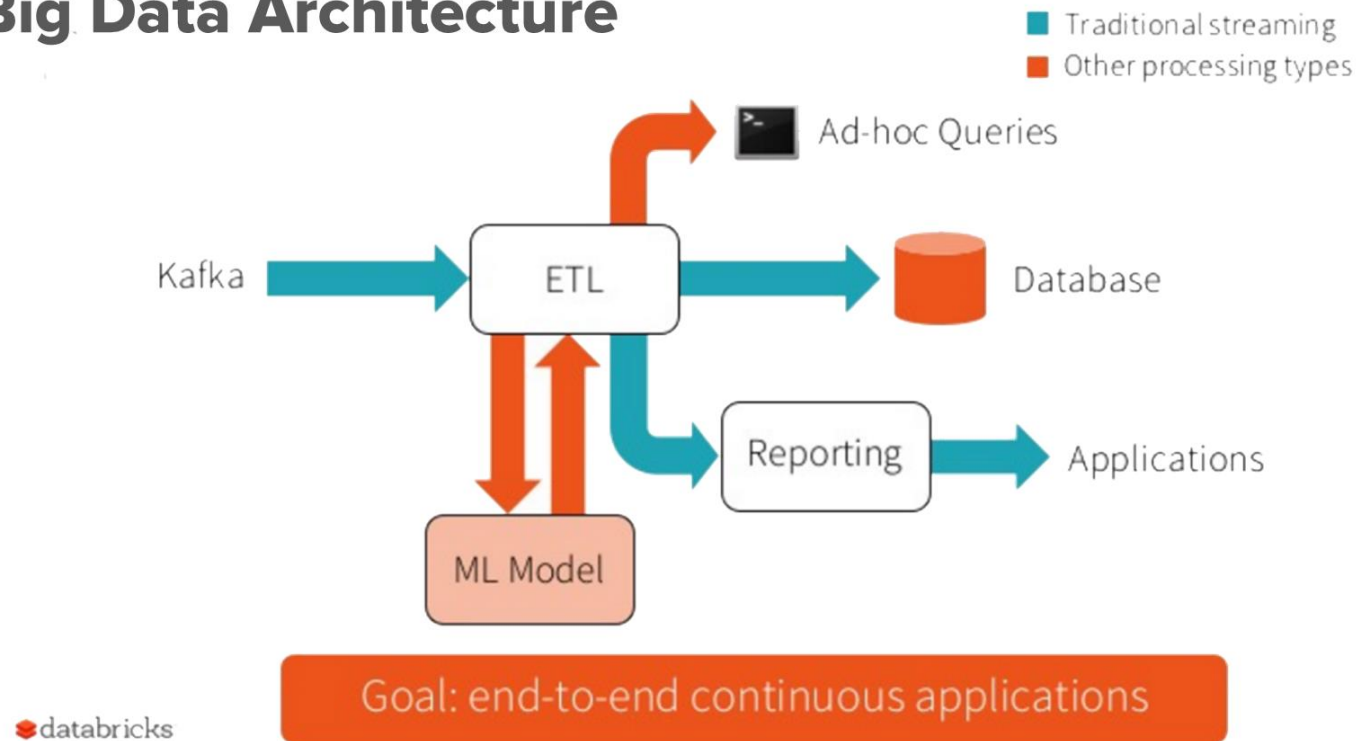
Contextualize demands on data applications

Identify popular frameworks

Describe design principles for scaling your queries

Slide 4: Big Data Architecture

Big Data Architecture



Slide 5: Architecture Needs: Scalability

Architecture Needs: Scalability

A scalable way to receive data

Producers publish to topics

Consumers consume from topics

Apache Kafka, Amazon Kinesis, Azure Event Hub



Slide 6: Architecture Needs: ETL Process

Architecture Needs: ETL Process

Extract – Transform – Load

Raw data → usable data



Slide 7: Other Architecture Needs

Other Architecture Needs

Transactional database

Ad hoc queries

Machine learning

Reporting

Scalable data lake



Slide 8: Decoupled Storage and Compute

Decoupled Storage and Compute

No data stays on the cluster

Resources are elastic

Updates are easy



Slide 9: IO vs CPU Bound Problems

IO vs. CPU Bound Problems

IO is network transfer (data transfer)

CPU is computation (processing data)

Data transfer is normally the bottleneck for tasks

Spark solves this by colocating your storage

Data Sources

Traditional databases like Postgres, SQL Server, and MySQL

NoSQL databases/documents stores like MongoDB

Distributed databases like Cassandra and Redshift

Blob stores like S3 and the Azure Blob

Message brokers like Kafka and Kinesis

Data warehouses like Hive

File types like CSV, Parquet, and Avro

Slide 11: Online Analytical Processing (OLAP)

Online Analytical Processing (OLAP)

Reporting

Ad hoc analysis

Blob stores

Data warehouses



Slide 12: Online Transaction Processing (OLTP)



Online Transaction Processing (OLTP)

Databases

Web traffic

Streaming