ROI TRAINING
MAXIMIZE YOUR TRAINING INVESTMENT

Spark Program

CHAPTER 10:
THE SPARK ECOSYSTEM

Chapter Objectives

In this chapter, we will:

- Look at other products in the big data space
  - Hive/HCatalog
  - Sqoop
  - Flume
  - Kafka
  - Storm
  - Flink

The Big Data Ecosystem

- Big data started with Hadoop
  - HDFS for storage
  - YARN to run Java MapReduce programs
- Hive and Pig joined later to allow you to write simpler scripts instead of Java code
  - HQL (Hive Query Language) is based on SQL
    - HCatalog is a subset of Hive that just handles the storage of metadata for table definitions
    - Hive could also be configured to use Spark for the backend instead of MapReduce
  - Pig Latin is its own language but is similar to Python
    - More designed for ETL operations than analytical queries
- Impala is a bridge between Hive and Spark
  - Uses the same HQL language
  - Caches more intermediate results in memory for higher performance
  - Doesn't cache as much as Spark and doesn't have all the other features Spark has

Sqoop

- Sqoop is a tool designed to efficiently move data between SQL sources and HDFS
- If you need to do a mass export of data from a SQL database into HDFS, you can set up jobs in Sqoop to parallel load it to multiple nodes simultaneously
- It is useful to export large SQL tables occasionally, but if you need to query the current data that is in the table, you can do so directly with Spark
  - But be aware that doing so may take a lot of bandwidth to get it into the cluster
- Typically, you export large tables and do periodic incremental updates of new data to refresh what is in HDFS, but it is difficult to account for changes to the records, whereas new records are easy to append to the end of a folder

Flume

- Flume is another single-use tool like Sqoop
- Instead of moving data in and out of SQL, it is designed to deal specifically with log files
- Typical use case would be to aggregate the log files of multiple web servers in a farm to consolidate them, scan them for keywords, aggregate them for performance metrics, etc.

Kafka

- Kafka is a streaming platform designed to create reliable, scalable, clusters to handle basic message queue-type operations
- It is designed to:
  - Create a publish/subscribe metaphor
  - Store the streams in a fault-tolerant and durable manner
  - Process the streams as they occur
- Can be used as just a bridge to stream data from a source that publishes it to a subscriber that receives it
  - We used Spark as a subscriber of Kafka streams in the Spark Streaming chapter
- Could also be used to process the streams itself

Storm

- Another stream processing engine that is clustered, fault tolerant, and reliable
- Does basically the same things that Spark Streaming does, but predates it a bit
- You build the solutions in Java using Storm APIs instead of using Python or Scala like you would in Spark

Flink

- Yet another stream processing engine that is clustered, fault tolerant, and reliable
- Built from the ground up as a stream processing engine, whereas Spark was built as a batch engine first, and streaming added later
- Could also do batch operations, but was designed primarily for streaming
- Has Java and Python APIs
- Lacks an interactive command-line environment like PySpark

Chapter Summary

In this chapter, we have:

- Looked at other products in the big data space
  - Hive/HCatalog
  - Sqoop
  - Flume
  - Kafka
  - Storm
  - Flink