Course Outline

Architecting Large-Scale Data Systems with Dask

Course Summary

Description

This class explores the best ways to leverage Dask within enterprise data architectures.

Most enterprises make heavy use of elements core to Dask (e.g., data manipulation and machine learning); activities external to Dask (e.g., using SQL for reporting and data extraction); and activities orthogonal to Dask but still critical to the success of the overall system (e.g., data storage). Moreover, staff and skillsets are often different across these areas.

We explore options and patterns for getting the best out of both Dask and non-Dask elements of the system.

Objectives

At the end of this course, students will understand:

- Where is Dask great? Where do we need additional tools?
- Integrating Dask with JVM-based data processing systems
- Best practices to allow your data team to excel with the skills they know best

Topics

- Introduction
- Integrating Data
- Data Processing, ETL, and Feature Engineering
- Data Output
- Additional Goals, Challenges, and Opportunities

Audience

This course is intended for those who manage data storage.

Prerequisites

The following prerequisites are required for this course:

- Python, basic level
- JVM/Hadoop/Spark/Kafka ecosystem, basic level
- Large-scale data storage patterns, basic level
- Understanding of ML concepts and workflow, basic level

Duration

One day

Course Outline

Architecting Large-Scale Data Systems with Dask

Course Outline

I. Introduction
- A. About Dask and Coiled Computing: Making scale-out computing easier
- B. Dask, the 30-000-foot view: scheduler, APIs, infrastructure support components
- C. Photographic negative: what's not in Dask
- D. Overview of an integration flow from data warehouse to reports or ML models

II. Integrating Data
- A. Finding your data: Hive metastore and alternatives
- B. Ingesting data: formats and locations
- C. Options for SQL access to data
- D. Distributed caching
- E. Consuming streaming data

III. Data Processing, ETL, and Feature Engineering
- A. Dask support
- B. Shuffling and other data transfer
- C. External Python integrations
- D. Custom functions / business logic
- E. Checking compatibility vs. ANSI SQL, SparkSQL, HiveQL
- F. Index pros/cons
- G. ML Modeling – orientation/overview
- H. Implementing custom algorithms

IV. Data Output
- A. Output artifacts
- B. ML Models
- C. Reports for human or business system consumption
- D. ETL writes into another datastore
- E. Output to a streaming or message-oriented middleware system
- F. Transactional/safe writes – present and future

V. Additional Goals, Challenges, and Opportunities
- A. Orchestration
- B. Resilient streaming systems
- C. ML serving/scoring systems
- D. GPU / heterogeneous compute integration
- E. Monitoring, management, and debugging interfaces
- F. End-user notebook integration
- G. Q & A