Qubole®

White Paper

BIG DATA ENGINEERING FOR MACHINE LEARNING

AN INTRODUCTION TO COMMONLY USED ENGINES AND FRAMEWORKS

by Jorge Villamariona, Qubole Technical Marketing

TABLE OF CONTENTS

INTRODUCTION

Even as individual consumers of goods and services we get to experience the results of Machine Learning when it is used by the institutions we rely on to conduct our daily activities. We may have experienced a text message from a bank requiring verification right after the bank has paused a credit card transaction. Or, on the more positive side, an online travel site may have sent us an email that offers the perfect accommodations for our next personal or business trip.

What is much more difficult to appreciate is the work that happens behind the scenes to facilitate these experiences. Yes, data science teams build applications that lead to these outcomes, but their work relies on massive data sets supplied by data engineering teams. These data sets are in time leveraged to train, build and test data science models that in turn facilitate these outcomes.

Data engineering teams are nothing new, they have been around for several decades. Their role has been most recently extended from building data pipelines that only support traditional data warehouses to also building more technically demanding continuous data pipelines that feed today's applications that leverage Artificial Intelligence and Machine Learning algorithms.

These data pipelines must be **cost effective**, fast, and **reliable** regardless of the type of workload and use case. This document covers the most popular engines used to build these pipelines. It delineates the synergies between data engineering and data science teams. It provides insights on how Qubole customers build their pipelines leveraging the steps outlined below:

In the final section this document provides a guide to help decide which engine to use based on the business need and type of workload.

ENGINES FOR BUILDING MACHINE LEARNING (ML) DATA PIPELINES

Due to the diversity of data sources, and the volume of data that needs to be processed, traditional data processing tools fail to meet the performance and reliability requirements for modern machine learning and advanced analytics applications. The need to build reliable pipelines that could handle these workloads coupled with advances in distributed high performance computing gave rise to data lake processing engines such as Hadoop. Let's quickly review the different engines and frameworks often used in data engineering aimed at supporting ML efforts:

Apache Hadoop/Hive

Hive is an Apache open-source project built on top of Hadoop for querying, summarizing and analyzing large data sets using a SQL-like interface. It is noted for bringing the familiarity of relational technology to data lake processing with its Hive Query Language, as well as structures and operations comparable to those used by relational databases such as tables, joins and partitions. Apache **Hive** is used mostly for batch processing of large ETL jobs and batch SQL queries on very large data sets as well as exploration on large volumes of structured, semi-structured, and unstructured data. Hive includes a Metastore which provides schemas and statistics that are useful for optimizing queries and data exploration. **Hive is Distributed, Scalable, Fault Tolerant, Flexible** and it can persist large data sets on cloud file systems. It is important to note that Hive was created by the founders of Qubole while working at Facebook's data team. **Qubole** provides an enhanced, cloud-optimized, self-managing, and self-optimizing implementation of Apache Hive. Qubole's implementation of Hive leverages AIR (Alerts, Insights, Recommendations) and allows data teams to focus on generating business value from data rather than managing the platform. Qubole Hive seamlessly integrates with existing data sources and third-party tools, while providing best-in-class security.

Apache Spark

Spark is a general purpose open-source computational engine for Hadoop data. Spark provides a simple and expressive programming model that supports a wide range of applications, including ETL, machine learning, stream processing, and graph computation. **Spark is also Distributed, Scalable, Fault Tolerant, Flexible, and because of its in-memory processing capabilities, it is also Fast**. Spark natively supports a wide array of programming languages, including J**ava, Python, R, and Scala.** Spark includes native integration with a number of data persistence solutions such as HDFS. Because Spark is memory intensive, it may not be the most cost-effective engine for all use cases. Qubole's Spark implementation greatly improves the performance of Spark workloads with enhancements such as fast storage, distributed caching, advanced indexing, and metadata caching capabilities. Other enhancements include job isolation on multi-tenant clusters and SparkLens, an open source Spark profiler that provides insights into the Spark application.

Presto

Presto is an open-source SQL query engine developed by Facebook. Presto is used for running interactive analytic queries against data sources of all sizes ranging from gigabytes to petabytes. **Presto** was built to provide SQL query capabilities against disparate data sources, this allows Presto users to combine data from multiple data sources in one query. This is known as "Federated Queries". **Presto** is tuned like an MPP (Massively Parallel Processing) SQL engine and it is optimized for SQL execution Since the questions are often ad-hoc, there is some trial and error involved; arriving at the final results may involve a series of SQL queries. Presto offers connectors to work directly on files that reside on the file system (e.g. S3, Azure storage) and can join terabytes of data in seconds, or cache queries intermittently for rapid response upon later runs. Qubole **Presto** also expedites performance by reordering the execution sequence of query joins based on table statistics. By optimizing the response time of these queries, **Presto** accelerates the time to insight to better serve your business needs. **Presto is also Distributed, Scalable, and Flexible**. Presto is also Fast due to its in-memory processing and it can be ready in just a few minutes. Qubole has optimized Presto for the cloud. Qubole's enhancements allow for dynamic cluster sizing based on workload and termination of idle clusters — ensuring high reliability while reducing compute costs. Qubole's Presto clusters support multi-tenancy and provide logs and metrics to track performance of queries.

Airflow

While technically, not a big data engine, **Airflow** is an open-source tool to programmatically author, schedule and monitor data workflows. With **Airflow**, users can author workflows as directed acyclic graphs (DAGs) of tasks. A DAG is the set of tasks needed to complete a pipeline organized to reflect their relationships and interdependencies. **Airflow's** rich user interface makes it easy to visualize pipelines running in production, monitor progress, and troubleshoot issues when needed. It connects out-of-the-box with multiple data sources, it can alert via email or slack when a task completes or fails. Because workflows are defined as code, they become more maintainable, version-able, testable, and collaborative. **Airflow is Distributed, Scalable and Flexible** which makes it well suited to handle the orchestration of complex business logic. Qubole provides its users an enhanced, cloud-optimized version of Apache Airflow. Qubole provides single-click deployment of Airflow, automates cluster and configuration management, and includes dashboards to visualize Airflow DAGs.

Besides providing these scalable engines and frameworks it is important to mention that Qubole also partners with 3rd party ETL/ELT vendors that offer visual tools that facilitate building data pipelines. These tools are very powerful, and many teams leverage them to expedite the work with the engines listed above.

EXPLORE

EXPLORING YOUR DATA

Data engineering always starts with data exploration. Data exploration is inspecting the data in order to understand its characteristics and what it represents. At the end of the exploration stage data engineers should understand the structure of the data, its volume, its patterns as well as its quality (e.g. are all the records complete? are there duplicate records?). The learnings acquired during this stage will impact the amount and type of work that the data scientist will do during their data preparation phase.

Because SQL is a widely used standard for ad hoc queries it is a good tool for users to start interacting with structured datasets, especially larger structured datasets (in the terabytes) that need to be persisted. Because of its inexpensive storage and it is compatible with SQL, **Hadoop/Hive** is an excellent choice in this case.

Spark works well for data sets that require a programmatic approach -- e.g. a hierarchical tree needs to be traversed just to read the data such as is the case with the 835 and 837 file formats widely used in healthcare insurance processing. **Spark** also offers facilities (Python, R, Scala) that can be used to quickly understand the nature and statistical distribution of the data (such as **5 number summary**, etc.).

Explore

- **With SQL on larger datasets (petabytes)**
- **Using SQL or programmatic constructs**
- **With SQL when interactivity and response time is important**

On the other hand, **Presto** provides a quick and easy way to allow access to data from a variety of sources using industry standard SQL query language. Users don't have to learn any new complex language or tool, they can simply utilize existing tools with which they are already comfortable. For data exploration, **Presto** is very similar to **Hive** but **Presto** offers the advantage of speed because of its in-memory processing.

Because fault tolerance is not critical during this phase, all 3 engines can be used for data exploration. Thus, the decision regarding which engine to use will depend on the nature of your data, the use case and the team's skill set. **Hive** may be more economical for larger datasets while **Presto** offers a significant speed advantage for quick interactive exploratory queries to better understand the data. **Spark** may be a better option if a programmatic facility is best suited for exploration. The insert on the right summarizes how Qubole customers often leverage our engines for data exploration.

BUILD

BUILDING ROBUST DATA PIPELINES

Data pipelines carry and process data from the data sources to the BI and ML applications that take advantage of it. These pipelines consist of multiple steps: reading data, moving it from one system to the next, reformatting it, joining it with other data sources and also adding derived columns (feature engineering). These new derived fields could represent characteristics such as "Age" or "Age Group" derived from date of birth or "Deal Size" derived from the amount field on a sales record. These derived fields support different data science models as well as more in-depth BI. Data scientists refer to this step as data **preparation**. A data pipeline includes all of these steps, and its mission is to ensure that every step happens consistently and reliably for all data sets. Data engineers often distinguish between two different types of pipelines: batch and streaming. Let's take a quick look at each type of pipeline:

Pipelines for batch processing

Traditional data engineering workloads consist of mostly "well-defined" data sets often coming from applications supporting key business functions such as accounting, inventory management, CRM systems as well as "well-defined" semi-structured or unstructured files - such as server logs. By "well-defined" we mean that most of these data sets remain relatively static between the start and end time of the execution of a pipeline. For "well-defined" data sets batch data pipelines are sufficient and adequate to support BI and ML applications.

When persistence of large data sets is important, **Hive** offers diverse computational techniques and it is cost effective. On the other hand, **Spark** offers an in-memory computational engine that allows the creation of programs that can read data, build a pipeline, and export the results, and it may be the better choice if speed of processing is critical.

Pipelines for streaming data

With the emergence of big data and the internet of things (IOT), batch data pipelines while still necessary and relevant are no longer sufficient. Data engineers have to develop streaming data pipelines to deal with near-real time requirements of ML applications built on top these new data frameworks. Let's take a closer look at both types of pipelines.

Streaming data is generated on a continuous basis and often by several data sources, it is transmitted simultaneously and in relative small packets. Examples of streaming data include call detail records generated by mobile phones, logs from a web application such as an ecommerce site, social network updates and eGame activities. This data is processed sequentially and in near-real time and often while the data is still "in flight" (before it is persisted in a database). In many streaming use cases the value of the data decreases with time. For example, an alert from a machine on a factory floor calls for immediate attention or a potentially fraudulent bank transaction needs to be paused in order for it to be verified by the account owner before it completes. **Spark** offers the best facilities for streaming data pipelines.

Spark Streaming brings Apache Spark's language-integrated API to stream processing, allowing engineers to create streaming jobs the same way they write batch jobs. Spark Streaming supports Java, Scala and Python.

Please note that while **Spark** offers the best facilities for near-real time data streaming, micro-batching (i.e. batch processing with smaller time windows such as intervals of a few minutes) may be a workable and more economical option leveraging **Hive. Hive** is also effective for training ML models with large, representative and persisted datasets. The insert on the right outlines how Qubole customers use these engines for building data pipelines.

Build

- Batch Pipelines
- Streaming Pipelines via Micro-batching
- Training ML Models

- Batch Pipelines
- Near-real time streaming Pipelines
- Training ML Models

ORCHESTRATE

ORCHESTRATING DATA PIPELINES

Orchestration of data pipelines refers to the sequencing, coordination, scheduling and management of complex data pipelines from diverse sources with the aim of delivering data sets that are ready for consumption either by business intelligence applications and/or data science ML models that support data lake applications.

Efficient, cost-effective and well-orchestrated data pipelines help data scientists come up with better tuned and more accurate ML models because those models have been **trained** with complete data sets not just small samples.

Because **Airflow** is natively integrated to work with engines such as **Hive, Presto and Spark**, it is an ideal framework to orchestrate jobs running on any of these engines. Organizations are increasingly adopting **Airflow** to orchestrate their ETL/ELT jobs. At the time of this writing about 18% of all ETL/ELT jobs on the Qubole platform leverage **Airflow** and the number of DAGs is increasing 15% month over month. Also, with Qubole contributed features such as the **Airflow** QuboleOperator customers have the ability to submit commands to Qubole, thus giving them greater programmatic flexibility.

Airflow is an ideal solution for orchestration of workloads that follow the batch processing model as outlined in the insert on the right.

Orchestrate

- Batch Pipelines
- Orchestration of data pipelines refers to the sequencing, co-ordination, scheduling and management of complex data pipelines from diverse sources with the aim of delivering datasets ready for use

DELIVER

DELIVERING DATA SETS

Qubole's multi-engine platform allows data engineers to build, update and refine data pipelines in order to reliably and cost-effectively deliver those data sets on predefined schedules or on-demand. Qubole provides the ability to publish data through notebooks or templates and deliver the data to downstream advanced analytics and ML applications. The data delivery stage has the greatest impact on the **training**, deployment and operation of ML models as well as the applications built on top of them.

At Qubole we believe your use case should determine the delivery engine. For example, if the information is going to be delivered as a dashboard or the intention is to probe the resulting datasets with low-latency SQL queries then **Presto** would be the optimal choice because with **Presto** queries run faster than with **Spark** since there is no consideration for mid-query fault-tolerance. **Spark** on the other hand supports mid-query fault-tolerance and will recover in case of a failure but, actively planning for failure impacts **Spark's** query performance, especially in the absence of any technical hiccups.

Spark also offers excellent support for delivering streamed data, datasets resulting from long-running (batch) queries that may require fault tolerance, as well as any dataset that may require programmatic constructs for interpretation. Common output destinations for **Spark** could also be file systems, databases, dashboards, notebooks and certainly ML/Predictive applications.

If cost effective persistence of the result set is important then **Hadoop/Hive** is a great choice because it supports a number of traditional BI tools via ODBC and JDBC. Also, because its ability to store large volumes of data, **Hadoop/Hive** is an ideal platform for a persisted single view of an entity (customer, patient, asset, etc.) such as those offered by MDM systems.

All three engines offer distinctive advantages for data delivery. The insert on the right shows how Qubole customers often leverage these three engines in this phase.

At Qubole we believe your use case should determine the delivery engine.

Deliver

- Larger datasets (petabytes) that need persistence such as ML training datasets or the ones that underpin a single view of an entity such as Customer 360 systems.
- Large Scale data cleansing and enrichment processes

- Streaming Datasets in near-real time
- Programmatic (Scala, Python, R) treatment of data workload
- Training datasets

- Ad-hoc interactive SQL queries that return data quickly
- Federated SQL Queries joining multiple disparate system/ environment

TAKEAWAYS

Each one of the engines covered on this document provides distinctive advantages depending on the use case. Companies leveraging data lakes have a wide variety of use cases and for a given company or even within a department not all of the use cases will be known initially. **As companies become more mature and sophisticated in their data lake deployments, data engineers will discover more use cases and opportunities to leverage different engines.**

As companies become more mature and sophisticated in their big data deployments, data engineers will discover more use cases and opportunities to leverage different engines.

For these reasons, building data pipelines calls for a multi-engine platform with the ability to auto-scale such as the one offered by **Qubole**.

Qubole is an open, simple, and secure data lake platform for machine learning, streaming analytics, data exploration, and ad-hoc analytics. No other platform radically simplifies data management, data engineering and run-time services like Qubole. We enable reliable, secure data access and collaboration among users while reducing time to value, improving productivity, and lowering cloud data lake costs from day one. The Qubole Open Data Lake Platform:

- Provides a unified environment for creation and orchestration of multiple data pipelines to dynamically build trusted datasets for ad-hoc analytics, streaming analytics, and machine learning.
- Optimizes and rebalances underlying multi-cloud infrastructure for the best financial outcome while supporting the unmatched levels of concurrent users, and workloads at any point in time.
- Enables collaboration through workbench between data scientists, data engineers, and data analysts with shareable notebooks, queries, and dashboards.
- Improves the security, governance, reliability, and accessibility of data residing in data lakes.
- Provides APIs, and connectors to 3rd party tools such as Tableau, Looker for analytics, RStudio, H2O.ai for machine learning use cases.

Table 1 below encapsulates how Qubole customers apply our different engines to fulfill different stages of the data engineering function when building ML data pipelines.

EXPLORE	BUILD	ORCHESTRATE	DELIVER
			
• With SQL on Larger datasets	• Batch Pipelines • Training ML Models • Streaming Pipelines via Micro- batching	• Batch workflows • Orchestration of data pipelines refers to the sequencing, co-ordination, scheduling and management of complex data pipelines from diverse sources with the aim of delivering datasets ready for use	• Larger datasets requiring persistence • Large scale clensing and enriching processes
			
• With programmatic constructs • With SQL	• Batch Pipelines • Near-real time streaming Pipelines • Training ML Models		• Streaming datasets • Workloads requiring programmatic constructs • Spark SQL
			
• With SQL when interactivity and response time is important			• Ad-hoc interactive SQL queries that return data quickly • Federated SQL Queries joining multiple disparate system

Table 1 - Most Common Engine and Framework usage patterns

This is based on how Qubole customers leverage our engines and frameworks in their ETL/ELT processes.

Table 2 dives deeper into the characteristics and merits of each tool and it can help you decide which engine to use depending on your use case and data workload.

ENGINE OR FRAMEWORK	HIVE	SPARK	PRESTO
CHARACTERISTICS			
Response Time (interactive queries)	Slower	Faster[1]	Faster
ETL SLA Adherence	High	High	N/A
Fault Tolerance	High	High	Limited[2]
Choice of Language	SQL	Scala, R, Python, SQL	SQL
Type of Workload/ Use Case	Batch Processing Larger Data Sets (Petabytes)	Machine Learning Batch Processing Stream Processing Graph Processing Interactive Programming	Exploration Interactive Sql BI/Analytics

Table 2 - Engine Decision Guide

1. Spark Clusters take longer to start. 2. Qubole offers retry logic at query level

In addition to offering several engines that allow our customers to select the most adequate tool for each job, **Qubole is the only cloud data platform that delivers a multi-cloud, multi-engine, self-service machine learning and analytics architecture**. It automatically provisions, manages and optimizes cloud resources balancing cost, workloads, and performance requirements. Qubole's open data lake platform enables orchestration and execution of all types of data engineering tasks whether it is data **exploration, building** of data pipelines, **orchestration** or data **delivery**.

When building a house, you would choose different tools for different tasks, it is impossible to build a house using only one tool. **Similarly, when building data pipelines, you should choose the optimal big data engine by considering your specific use case and the specific business needs of your company or department.**

About Qubole

Qubole is revolutionizing the way companies activate their data — the process of putting data into active use across their organizations. With Qubole's cloud-native big data platform, companies exponentially activate petabytes of data faster, for everyone and any use case, while continuously lowering costs. Qubole overcomes the challenges of expanding users, use cases, and variety and volume of data while constrained by limited budgets and a global shortage of big data skills. Qubole offers the only platform that delivers freedom of choice, eliminating legacy lock in — use any engine, any tool, and any cloud to match your company's needs. Qubole investors include CRV, Harmony Partners, IVP, Lightspeed Venture Partners, Norwest Venture Partners, and Singtel Innov8.

For more information visit **www.qubole.com.**

469 El Camino Real, Suite 201
Santa Clara, CA 95050
(855) 423-6674 | info@qubole.com
WWW.QUBOLE.COM