Introduction to Emerging Big Data Technologies

Estimated time: 30 minutes

Introduction

In the evolving world of big data, new technologies are emerging to address the growing need for faster, more scalable, and efficient data processing. Traditional systems like Hadoop and Spark have been the backbone of big data processing, but modern technologies such as Apache Pulsar, Apache Druid, and PrestoDB are gaining popularity for their unique capabilities. This reading will explore these technologies and compare their features to Hadoop and Spark, providing you with a broader understanding of the current big data landscape.

Objectives

By the end of this reading, learners will be able to:

1. Explain the core functionalities and use cases of modern big data technologies
2. Differentiate traditional and emerging solutions
3. Recognize the specific scenarios where each technology—Pulsar for messaging and streaming, Druid for real-time analytics, and PrestoDB for distributed querying—is best suited
4. Evaluate the scalability, performance, and latency trade-offs of these emerging technologies compared to traditional big data systems
5. Determine the right technology stack for real-world applications

---

1. Apache Pulsar

Overview

Apache Pulsar is an open-source, cloud-native, distributed messaging and streaming platform developed by Yahoo in 2016. It was built to handle high-throughput, low-latency messaging for large-scale, globally distributed systems. Pulsar combines features of both message queueing systems (like Kafka) and real-time streaming platforms, making it highly versatile for modern data architectures.

Key features

- **Unified messaging model:** Pulsar supports both message queueing and real-time streaming in a single system. This reduces the need to integrate multiple solutions for different use cases.
- **Multi-tenancy and isolation:** Pulsar architecture is designed with multi-tenancy in mind, allowing multiple clients or teams to share the same Pulsar cluster while maintaining strong isolation between their workloads. This makes Pulsar ideal for large organizations with various use cases.
- **Geo-replication:** Pulsar natively supports geo-replication, allowing seamless data replication across multiple geographic locations, which is critical for building globally distributed systems.
- **Topic compaction and retention:** Pulsar supports topic compaction, which keeps only the most recent values for each key, and message retention policies that can store messages for defined periods.
- **Scalability and performance:** Pulsar separates compute and storage (via Apache BookKeeper), allowing horizontal scaling with minimal impact on performance. Its architecture ensures low-latency message delivery and can scale to millions of topics.

Use cases

- **Real-time analytics:** Pulsar is well-suited for capturing real-time events and streaming them to downstream analytics engines like Apache Druid or Apache Spark.
- **Event-driven architectures:** Pulsar ability to handle both message queueing and streaming makes it ideal for event-driven applications such as microservices.
- **Data pipeline orchestration:** It can serve as the backbone of complex data pipelines, ensuring real-time, fault-tolerant message delivery.

Limitations

- Learning curve for deployment and operation
- Limited community and ecosystem compared to Kafka

---

2. Apache Druid

Overview

Apache Druid is a real-time analytics database designed for rapid ingestion and querying of event-driven data. Originally developed by Metamarkets (later acquired by Snap Inc.), Druid excels at handling high volumes of time-series data, such as logs, metrics, and clickstreams.

Key features

- **Columnar data storage:** Druid stores data in a columnar format, which allows for fast aggregation and filtering of large data sets. This structure is optimized for analytical queries that scan large amounts of data.
- **Real-time data ingestion:** Druid can ingest streaming data from sources like Apache Kafka or Pulsar and provide near-instant query results. Its ability to combine streaming and batch data ingestion makes it highly versatile.
- **Time-series analysis:** Druid architecture is optimized for time-based data, making it ideal for use cases like time-series data analysis, trend monitoring, and anomaly detection.
- **Horizontal scalability:** Druid can scale horizontally by adding more nodes, allowing it to handle petabytes of data efficiently.
- **Optimized for OLAP:** Druid is purpose-built for Online Analytical Processing (OLAP) workloads, which involve complex queries that aggregate large datasets.

Use cases

- **Interactive dashboards:** Druid is commonly used for powering interactive analytics dashboards where users need to query large datasets in real time.
- **Fraud detection:** Its ability to analyze large volumes of real-time data makes it suitable for detecting patterns of fraud or suspicious activity.

- **Operational analytics:** Businesses use Druid to monitor and analyze metrics from their operational systems, such as monitoring system logs or user activity.

Limitations

- Complexity in setup and configuration
- Limited flexibility for non-time-series data

---

3. PrestoDB

Overview

PrestoDB is an open-source distributed SQL query engine, developed by Facebook, that allows querying across large data sets stored in a variety of systems such as HDFS, S3, MySQL, and NoSQL stores. Presto is designed to execute complex queries at interactive speeds, making it an ideal solution for businesses looking for fast querying without moving data.

Key features

- **Federated queries:** Presto can query data from multiple sources, combining data sets from different systems into a single query. This eliminates the need for ETL (Extract, Transform, Load) processes, making it easier to work with disparate data sources.
- **Interactive query speeds:** Presto is optimized for running SQL queries with low-latency, making it a popular choice for business intelligence (BI) tools where fast querying of large datasets is essential.
- **Pluggable architecture:** Presto offers extensible architecture that allows for integration with a wide range of data sources through custom connectors.
- **Scalability:** Presto can scale horizontally across clusters, enabling it to handle large data sets spread across different storage systems.

Use cases

- **Data lake querying:** Presto is frequently used to query data in distributed data lakes such as those built on top of HDFS or AWS S3.
- **Interactive analytics:** Business intelligence and reporting tools use Presto to provide users with fast, interactive access to large data sets.
- **Ad hoc data exploration:** Data scientists and engineers use Presto for exploratory data analysis across multiple data sources.

Limitations

- Not designed for transaction processing or streaming workloads
- Requires careful tuning for large-scale queries

Comparison with Hadoop and Spark

	Feature	Apache Hadoop	Apache Spark	Apache Pulsar	Apache Druid	PrestoDB
1	**Core functionality**	Batch processing	Batch + Stream processing	Message queueing + Streaming	Real-time analytics database	SQL query engine
2	**Primary use case**	Large-scale ETL	Real-time and batch analytics	Event-driven microservices	Real-time + historical queries	Interactive querying
3	**Data processing**	Disk-based, MapReduce	In-memory processing	Stream and message processing	Real-time ingestion	SQL queries across sources
4	**Latency**	High	Low (in-memory)	Low	Low (optimized for OLAP)	Low
5	**Scalability**	High but requires tuning	High	Very high (geo-replication)	Horizontally scalable	Horizontally scalable
6	**Data ingestion**	Batch	Batch + Stream	Stream	Batch + Stream	Batch
7	**Strengths**	High fault-tolerance, mature ecosystem	Fast, in-memory computing, ML libraries	Unified messaging and streaming	Time-series analysis, fast aggregations	Fast interactive queries across multiple data sources
8	**Weaknesses**	High latency, not suitable for real-time	Complex for small-scale tasks	Complex configuration, smaller ecosystem	Limited non-time-series flexibility	Requires tuning for optimal performance

---

Conclusion

While traditional big data platforms like Hadoop and Spark remain foundational, the growing complexity and real-time requirements of modern data systems have led to the rise of specialized tools like Apache Pulsar, Apache Druid, and PrestoDB. These technologies provide solutions for real-time analytics, event-driven architectures, and fast querying across distributed data, making them essential for organizations facing modern data challenges. Understanding these emerging tools is beneficial for building efficient, scalable, and future-proof data systems.

Author

Rajashree Patil

Skills Network