Introduction to Big Data with Spark and Hadoop

Glossary: Introduction to Big Data with Spark and Hadoop

Welcome! This alphabetized glossary contains many of the terms in this course. This comprehensive glossary also includes additional industry-recognized terms not used in course videos. These terms are essential for you to recognize when working in the industry, participating in user groups, and in other professional certificate programs.

Estimated reading time: 20 minutes

Term	Definition
Aggregating data	Aggregation is a Spark SQL process frequently used to present aggregated statistics. Commonly used aggregation functions such as count(), avg(), max(), and others are built into DataFrames. Users can also perform aggregation programmatically using SQL queries and table views.
AIOps	Implies the use of artificial intelligence to automate or enhance IT operations. It helps collect, aggregate, and work with large volumes of operations data. It also helps to identify events and patterns in large or complex infrastructure systems. It allows quick diagnosis of the root cause of issues so that users can report or fix them automatically.
Amazon Simple Storage Service (Amazon S3)	An object store interface protocol that Amazon invented. It is a Hadoop component that understands the S3 protocol. S3 provides an interface for Hadoop services, such as IBM Db2 Big SQL, to consume S3-hosted data.
Analyze data using printSchema	In this phase, users examine the schema or the DataFrame column data types using the print schema method. It is imperative to note the data types in each column. Users can apply the select() function to examine data from a specific column in detail.
Anomaly detection	A process in machine learning that identifies data points, events, and observations that deviate from a data set's normal behavior. Detecting anomalies from time series data is a pain point that is critical to address for industrial applications.
Apache	This open-source HTTP server implements current HTTP standards to be highly secure, easily configurable, and highly extendible. The Apache Software License by the Apache Software Foundation builds and distributes it.
Apache Cassandra	It is a scalable, NoSQL database specifically designed to not have a single point of failure.
Apache HBase	A robust NoSQL datastore that efficiently manages storage and computation resources independently of the Hadoop ecosystem.
Apache Mesos	General-purpose cluster manager with additional benefits. Using Mesos has some advantages. It can provide dynamic partitioning between Spark and other big data frameworks and scalable partitioning between multiple Spark instances. However, running Spark on Apache Mesos might require additional setup, depending on your configuration requirements.
Apache Nutch	An extensible and scalable web crawler software product used to aggregate data from the web.
Apache Spark	An open-source, distributed computing framework designed for processing large-scale data and performing data analytics. It provides libraries for various data processing tasks, including batch processing, real-time stream processing, machine learning, and graph processing. Spark is known for its speed and ease of use, making it a popular choice for big data applications.
Apache Spark Architecture	Consists of the driver and the executor processes. The cluster comprises the cluster manager and worker nodes. The Spark Context schedules tasks for the cluster, and the cluster manager manages the cluster's resources.
Apache Spark Cluster Modes	Apache Spark offers various cluster modes for distributed computing, including Standalone, YARN (Yet Another Resource Negotiator), and Apache Mesos. Each mode has specific characteristics and setup complexities.
Apache Spark User Interface	Provides valuable insights, organized on multiple tabs, about the running application. The Jobs tab displays the application's jobs, including job status. The Stages tab reports the state of tasks within a stage. The Storage tab shows the size of any RDDs or DataFrames that persisted to memory or disk. The Environment tab includes any environment variables and system properties for Spark or the JVM . The Executor tab displays a summary that shows memory and disk usage for any executors in use for the application. Additional tabs display based on the type of application in use.
Apache ZooKeeper	A centralized service for maintaining configuration information to maintain healthy links between nodes. It provides synchronization across distributed applications. It is also used to track server failure and network partitions by triggering an error message and then repairing the failed nodes.
Application dependency issues	Spark applications can have many dependencies, including application files such as Python script files, Java JAR files, and even required data files. Applications depend on the libraries used and their dependencies. Dependencies must be made available on all cluster nodes, either by preinstallation, including the dependencies in the spark-submit script bundled with the application, or as additional arguments.
Application programming interface (API)	Set of well-defined rules that help applications communicate with each other. It functions as an intermediary layer for processing data transfer between systems, allowing companies to open their application data and functionality to business partners, third-party developers, and other internal departments.
Application resource issues	CPU cores and memory can become an issue if a task is in the scheduling queue and the available workers do not have enough resources to run the tasks. As a worker finishes a task, the CPU and memory are freed, allowing the scheduling of another task. However, if the application asks for more resources that can ever become available, the tasks might never be run and eventually time out. Similarly, suppose that the executors are running long tasks that never finish. In that case, their resources never become available, which also causes future tasks to never run, resulting in a timeout error. Users can readily access these errors when they view the UI or event logs.
Application-id	A unique ID that Spark assigns to each application. These log files appear for each executor and driver process that the application runs.
Big data	Data sets whose type or size supersedes the ability of traditional relational databases to manage, capture, and process the data with low latency. Big data characteristics include high volume, velocity, and variety.
Big data analytics	Uses advanced analytic techniques against large, diverse big data sets that include structured, semi-structured, and unstructured data from different sources and sizes, from terabytes to zettabytes. It helps companies gain insights from the data collected by IoT devices.
Big data programming tools	Programming tools are the final component of big data commercial tools. These programming tools perform large-scale analytical tasks and operationalize big data. They also provide all necessary functions for data collection, cleaning, exploration, modeling, and visualization. Some popular tools you can use for programming include R, Python, SQL, Scala, and Julia.
Block	Minimum amount of data written or read, and also offers fault tolerance. The default block size can be 64 MB or 128 MB, depending on the user's system configuration. Each file stored need not take up the storage of the preconfigured block size.
Bootstrap data set (BSDS)	A Virtual Storage Access Method (VSAM) key-sequenced data set (KSDS) used to store the essential information required by IBM MQ (message queuing). The BSDS typically includes an inventory of all active and archived log data sets known to IBM MQ. IBM MQ uses this inventory to track active and archived log data sets. The BSDS plays a critical role in the proper functioning and management of IBM MQ, ensuring the integrity and availability of log data sets within the messaging system.
Business intelligence (BI)	Encompasses various tools and methodologies designed to convert data into actionable insights efficiently.
Catalyst	Within Spark's operational framework, it employs a pair of engines, namely Catalyst and Tungsten, in a sequential manner for query enhancement and execution. Catalyst's primary function involves deriving an optimized physical query plan from the initial logical query plan. This

Term	Definition
	optimization process entails implementing a range of transformations such as predicate pushdown, column pruning, and constant folding onto the logical plan.
Catalyst phases	Catalyst analyzes the query, DataFrame, unresolved logical plan, and Catalog to create a logical plan in the Analysis phase. The logical plan evolves into an optimized logical plan in the logical optimization phase. It is the rule-based optimization step of Spark SQL. Rules such as folding, pushdown, and pruning are applicable here. Catalyst generates multiple physical plans based on the logical plan in the physical planning phase. A physical plan describes computation on data sets with specific definitions explaining how to conduct the computation. A cost model then selects the physical plan with the least cost. This explains the cost-based optimization step. Code generation is the final phase. In this phase, the Catalyst applies the selected physical plan and generates Java bytecode to run on the nodes.
Catalyst query optimization	Catalyst Optimizer uses a tree data structure and provides the data tree rule sets in the background. Catalyst performs the following four high-level tasks to optimize a query: analysis, logical optimization, physical planning, and code generation.
Classification algorithms	A type of of machine learning algorithm that helps computers learn how to categorize things into different groups based on patterns they find in data.
Cloud computing	Allows customers to access infrastructure and applications over the internet without needing on-premises installation and maintenance. By leveraging cloud computing, companies can utilize server capacity on-demand and rapidly scale up to handle the extensive computational requirements of processing large data sets and executing complex mathematical models.
Cloud providers	Offer essential infrastructure and support, providing shared computing resources that encompass computing power, storage, networking, and analytical software. These providers also offer software as a service model featuring specific solutions, enabling enterprises to efficiently gather, process, and visualize data. Prominent examples of cloud service providers include AWS, IBM, GCP, and Oracle.
Cluster management framework	It handles the distributed computing aspects of Spark. It can exist as a stand-alone server, Apache Mesos, or Yet Another Resource Network (YARN). A cluster management framework is essential for scaling big data.
Clusters	Refer to groups of servers that are managed collectively and participate in workload management. You can have nodes within a cluster, typically individual physical computer systems with distinct host IP addresses. Each node in a cluster can run one or more application servers. Clusters are a fundamental concept in distributed computing and server management, allowing for the efficient allocation of resources and the scalability of applications and services across multiple server instances.
Command-line interface (CLI)	Used to enter commands that enable users to manage the system.
Committer	Most open-source projects have formal processes for contributing code and include various levels of influence and obligation to the project: Committer, contributor, user, and user group. Typically, committers can modify the code directly.
Commodity hardware	Consists of low-cost workstations or desktop computers that are IBM-compatible and run multiple operating systems such as Microsoft Windows, Linux, and DOS without additional adaptations or software.
Compute interface	It is a shared boundary in computing against which two or more different computer system components exchange information.
Containerization	Implies Spark applications are more portable. It makes it easier to manage dependencies and set up the required environment throughout the cluster. It also supports better resource sharing.
Cost-based optimization	Cost is measured and calculated based on the time and memory that a query consumes. Catalyst optimizer selects a query path that results in minimal time and memory consumption. As queries can use multiple paths, these calculations can become quite complex when large data sets are part of the calculation.
Creating a view in Spark SQL	It is the first step in running SQL queries in Spark SQL. It is a temporary table used to run SQL queries. Both temporary and global temporary views are supported by Spark SQL. A temporary view has a local scope. Local scope implies that the view exists within the current Spark session on the current node. A global temporary view exists within the general Spark application. This view is shareable across different Spark sessions.
DAGScheduler	As Spark acts and transforms data in the task execution processes, the DAGScheduler facilitates efficiency by orchestrating the worker nodes across the cluster. This task-tracking makes fault tolerance possible, as it reapplies the recorded operations to the data from a previous state.
Data engineering	It is a prominent practice that entails designing and building systems for collecting, storing, and analyzing data at scale. It is a discipline with applications in different industries. Data engineers use Spark tools, including the core Spark engine, clusters, executors and their management, Spark SQL, and DataFrames.
Data ingestion	The first stage of big data processing. It is a process of importing and loading data into IBM® WatsonX.data. You can use the Ingestion jobs tab from the Data manager page to load data securely and easily into WatsonX.data console.
Data science	Discipline that combines math and statistics, specialized programming, advanced analytics, artificial intelligence (AI), and machine learning with specific subject matter expertise to unveil actionable insights hidden in the organization's data. These insights can be used in decision-making and strategic planning.
Data sets	Created by extracting data from packages or data modules. They can be used to gather a customized collection of items that you use frequently. As users update their data set, dashboards, and stories that use the data set are also kept updated.
Data validation	The practice of verifying the integrity, quality, and correctness of data used within Spark applications or data processing workflows. This validation process includes checking data for issues such as missing values, outliers, or data format errors. Data validation is crucial for ensuring that the data being processed in Spark applications is reliable and suitable for analysis or further processing. Various techniques and libraries, such as Apache Spark's DataFrame API or external tools, can be employed to perform data validation tasks in Spark environments.
Data warehouse	Stores historical data from many different sources so users can analyze and extract insights from it.
DataFrame operations	Refer to a set of actions and transformations that can be applied to a DataFrame, which is a two-dimensional data structure in Spark. Data within a DataFrame is organized in a tabular format with rows and columns, similar to a table in a relational database. These operations encompass a wide range of tasks, including reading data into a DataFrame, performing data analysis, executing data transformations (such as filtering, grouping, and aggregating), loading data from external sources, and writing data to various output formats. DataFrame operations are fundamental for working with structured data efficiently in Spark.
DataFrames	Data collection categorically organized into named columns. DataFrames are conceptually equivalent to a table in a relational database and similar to a dataframe in R or Python, but with greater optimizations. They are built on top of the SparkSQL RDD API. They use RDDs to perform relational queries. Also, they are highly scalable and support many data formats and storage systems. They are developer-friendly, offering integration with most big data tools via Spark and APIs for Python, Java, Scala, and R.
Declarative programming	A programming paradigm that a programmer uses to define the program's accomplishment without defining how it needs to be implemented. The approach primarily focuses on what needs to be achieved, rather than advocating how to achieve it.
Directed acyclic graph (DAG)	Conceptual representation of a series of activities. A graph depicts the order of activities. It is visually presented as a set of circles, each representing an activity, some connected by lines, representing the flow from one activity to another.
distinct ([numTasks])	It helps in finding the number of varied elements in a data set. It returns a new data set containing distinct elements from the source data set.
Distributed computing	A system or machine with multiple components located on different machines. Each component has its own job, but the components communicate with each other to run as one system for the end user.
Driver	Receives query statements submitted through the command line and sends the query to the compiler after initiating a session.

Term	Definition
Driver memory	Refers to the memory allocation designated for the driver program of a Spark application. The driver program serves as the central coordinator of tasks, managing the distribution and execution of Spark jobs across cluster nodes. It holds the application's control flow, metadata, and the results of Spark transformations and actions. The driver memory's capacity is a critical factor that impacts the feasibility and performance of Spark applications. It should be configured carefully to ensure efficient job execution without memory-related issues.
Driver program	It can be run in either client or cluster mode. In client mode, the application submitter (such as a user machine terminal) launches the driver outside the cluster. In cluster mode, the driver program is sent to and run on an available Worker node inside the cluster. The driver must be able to communicate with the cluster while it is running, whether it is in client or cluster mode.
Dynamic configuration	Refers to a practice employed in software development to avoid hardcoding specific values directly into the application's source code. Instead, critical configuration settings, such as the location of a master server, are stored externally and are adjustable without modifying the application's code.
Environment tab	Entails several lists to describe the environment of the running application. These lists include the Spark configuration properties, resource profiles, properties for Hadoop, and the current system properties.
Environment variables	Spark Application configuration method in which environment variables are loaded on each machine, so they can be adjusted on a per-machine basis if hardware or installed software differs between the cluster nodes.
Executor	Utilizes a set portion of local resources as memory and compute cores, running one task per available core. Each executor manages its data caching as dictated by the driver. In general, increasing executors and available cores increases the cluster's parallelism. Tasks run in separate threads until all cores are used. When a task finishes, the executor puts the results in a new RDD partition or transfers them back to the driver. Ideally, limit utilized cores to the total cores available per node.
Executor memory	Used for processing. If caching is enabled, additional memory is used. Excessive caching results in out-of-memory errors.
Executors tab	A component of certain distributed computing frameworks and tools used to manage and monitor the execution of tasks within a cluster. It typically presents a summary table at the top that displays relevant metrics for active or terminated executors. These metrics may include task-related statistics, data input and output, disk utilization, and memory usage. Below the summary table, the tab lists all the individual executors that have participated in the application or job, which may include the primary driver. This list often provides links to access the standard output (stdout) and standard error (stderr) log messages associated with each executor process. The Executors tab serves as a valuable resource for administrators and operators to gain insights into the performance and behavior of cluster executors during task execution.
Extended Hadoop Ecosystem	Consists of libraries or software packages commonly used with or installed on top of the Hadoop core.
Extract, load, and transform (ELT)	It emerged because of big data processing. All the data resides in a data lake. A data lake is a pool of raw data for which the data purpose is not predefined. In a data lake, each project forms individual transformation tasks as required. It does not anticipate all the transformation requirements usage scenarios as in the case of ETL and a data warehouse. Organizations opt to use a mixture of ETL and ELT.
Extract, transform, and load (ETL) process	A systematic approach that involves extracting data from various sources, transforming it to meet specific requirements, and loading it into a data warehouse or another centralized data repository.
Fault tolerance	A system is fault-tolerant if it can continue performing despite parts failing. Fault tolerance helps to make your remote-boot infrastructure more robust. In the case of OS deployment servers, the whole system is fault-tolerant if the OS deployment servers back up each other.
File system	An all-comprehensive directory structure that includes a root (/) directory and other directories and files under it. It is confined to a logical volume. The complete information about the file system is centralized in the /etc/filesystems file.
filter (*func*)	It helps in filtering the elements of a data set basis its function. The filter operation is used to selectively retain elements from a data set or DataFrame based on a provided function (*func*). It allows you to filter and extract specific elements that meet certain criteria, making it a valuable tool for data transformation and analysis.
flatmap (*func*)	Similar to map (*func*) can map each input item to zero or more output items. Its function should return a Seq rather than a single item.
Flume	A distributed service that collects, aggregates, and transfers big data to the storage system. Offers a simple yet flexible architecture that streams data flows and uses an extensible data model, allowing online analytic applications.
For-loop	Extends from a FOR statement to an END FOR statement and executes for a specified number of iterations, defined in the FOR statement.
Functional programming (FP)	A style of programming that follows the mathematical function format. Declarative implies that the emphasis of the code or program is on the "what" of the solution as opposed to the "how to" of the solution. Declarative syntax abstracts out the implementation details and only emphasizes the final output, restating "the what." We use expressions in functional programming, such as the expression f of x, as mentioned earlier.
Hadoop	An open-source software framework that provides dependable distributed processing for large data sets through the utilization of simplified programming models.
Hadoop Common	Fundamental part of the Apache Hadoop framework. It refers to a collection of primary utilities and libraries that support other Hadoop modules.
Hadoop Distributed File System (HDFS)	A file system distributed on multiple file servers, allowing programmers to access or store files from any network or computer. It is the storage layer of Hadoop. It works by splitting the files into blocks, creating replicas of the blocks, and storing them on different machines. It is built to access streaming data seamlessly. It uses a command-line interface to interact with Hadoop.
Hadoop Ecosystem	It splits big data analytics processing tasks into smaller tasks. The small tasks are performed in conjunction using an algorithm (e.g., MapReduce) and then distributed across a Hadoop cluster (i.e., nodes that perform parallel computations on big data sets).
Hadoop Ecosystem stages	The four main stages are: Ingest, store, process, analyze, and access.
HBase	A column-oriented, non-relational database system that runs on top of the Hadoop Distributed File System (HDFS). It provides real-time wrangling access to the Hadoop file system. It uses hash tables to store data in indexes and allow for random data access, making lookups faster.
High-throughput	Throughput quantifies the data processed in a timeframe. The target system needs robust throughput for heavy workloads with substantial data changes from the source database to prevent latency spikes. Performance objectives are frequently outlined with throughput targets. High throughput is achieved when most messages are delivered successfully, whereas low successful delivery rates indicate poor throughput and network performance.
Hive	A data warehouse infrastructure employed for data querying and analysis, featuring an SQL-like interface. It facilitates report generation and utilizes a declarative programming language, enabling users to specify the data they want to retrieve.
Hive client	Hive provides different drivers for communication depending on the type of application. For example, for Java-based applications, it uses JDBC drivers, and other types of applications will use ODBC drivers. These drivers communicate with the servers.
Hive server	Used to execute queries and enable multiple clients to submit requests. It is built to support JDBC and ODBC clients.
Hive services	Client interactions are done through the Hive services. Any query operations are done here. The command-line interface acts as an interface for the Hive service. The driver takes in query statements, monitors each session's progress and life cycle, and stores metadata generated from the query statements.
Hive tables	Spark supports reading and writing data stored in Apache Hive.

Term	Definition
Hive Web Interface	A web-based user interface that interacts with Hive through a web browser. It offers a graphical user interface (GUI) used to browse tables, execute Hive queries, and manage Hive resources.
HMaster	The master server that monitors the region server instances. It assigns regions to region servers and distributes services to different region servers. It also manages any changes that are made to the schema and metadata operations.
Hue	An acronym for Hadoop user experience. It allows you to upload, browse, and query data. Users can run Pig jobs and workflow in Hue. It also provides an SQL editor for several query languages, like Hive and MySQL.
Hybrid cloud	Unifies and combines public and private cloud and on-premises infrastructure to create a single, cost-optimal, and flexible IT infrastructure.
IBM Analytics Engine	Works with Spark to provide a flexible, scalable analytics solution. It uses an Apache Hadoop cluster framework to separate storage and compute by storing data in object storage such as IBM Cloud Object Storage. This implies users can run compute nodes only when required.
IBM Spectrum Conductor	A multitenant platform for deploying and managing Spark and other frameworks on a cluster with shared resources. This enables multiple Spark applications and versions to be run together on a single large cluster. Cluster resources can be divided up dynamically, avoiding downtime. IBM Spectrum Conductor also provides Spark with enterprise-grade security.
IBM Watson	Creates production-ready environments for AI and machine learning by providing services, support, and holistic workflows. Reducing setup and maintenance saves time so that users can concentrate on training Spark to enhance its machine-learning capabilities. IBM Cloud Pak for Watson AIOps offers solutions with Spark that can correlate data across your operations toolchain to bring insights or identify issues in real time.
Immutable	This type of object storage allows users to set indefinite retention on the object if they are unsure of the final duration of the retention period or want to use event-based retention. Once set to indefinite, user applications can change the object retention to a finite value.
Impala	A scalable system that allows nontechnical users to search for and access the data in Hadoop.
Imperative programming paradigm	In this software development paradigm, functions are implicitly coded in every step used in solving a problem. Every operation is coded, specifying how the problem will be solved. This implies that pre-coded models are not called on.
In-memory processing	The practice of storing and manipulating data directly in a computer's main memory (RAM), allowing for faster and more efficient data operations compared to traditional disk-based storage.
InputSplits	Created by the logical division of data. They serve as an input to a single Mapper job.
Internet of Things (IoT)	A system of physical objects connected through the internet. A "thing or device" can include a smart device in our homes or a personal communication device such as a smartphone or computer. These collect and transfer massive amounts of data over the internet without manual intervention by using embedded technologies.
Iterative process	An approach to continuously improving a concept, design, or product. Creators produce a prototype, test it, tweak it, and repeat the cycle to get closer to the solution.
JAR (Java Archive)	A standard file format used to package Java classes and related resources into a single compressed file. JAR files are commonly used to bundle Java libraries, classes, and other assets into a single unit for distribution and deployment.
Java	Technology equipped with a programming language and a software platform. To create and develop an application using Java, users are required to download the Java Development Kit (JDK), available for Windows, macOS, and Linux.
Java virtual machines (JVMs)	The platform-specific component that runs a Java program. At run time, the VM interprets the Java bytecode compiled by the Java Compiler. The VM is a translator between the language and the underlying operating system and hardware.
JavaScript Object Notation (JSON)	A simplified data-interchange format based on a subset of the JavaScript programming language. IBM Integration Bus provides support for a JSON domain. The JSON parser and serializer process messages in the JSON domain.
JDBC client	Component in the Hive client, which allows Java-based applications to connect to Hive.
Job details	Provides information about the different stages of a specific job. The timeline displays each stage, where the user can quickly see the job's timing and duration. Below the timeline, completed stages are displayed. In the parentheses beside the heading, users will see a quick view that displays the number of completed stages. Then, view the list of stages within the job and job metrics, including when the job was submitted, input or output sizes, the number of attempted tasks, the number of succeeded tasks, and how much data was read or written because of a shuffle.
Jobs tab	Commonly found in Spark user interfaces and monitoring tools, it offers an event timeline that provides key insights into the execution flow of Spark applications. This timeline includes crucial timestamps such as the initiation times of driver and executor processes, along with the creation timestamps of individual jobs within the application. The Jobs tab serves as a valuable resource for monitoring the chronological sequence of events during Spark job execution.
JSON data sets	Spark infers the schema and loads the data set as a DataFrame.
Kubernetes (K8s)	A popular framework for running containerized applications on a cluster. It's an open-source system that is highly scalable and provides flexible deployments to the cluster. Spark uses a built-in native Kubernetes scheduler. It is portable, so it can be run in the same way on cloud or on-premises.
Lambda calculus	A mathematical concept that implies every computation can be expressed as an anonymous function that is applied to a data set.
Lambda functions	Calculus functions, or operators. These are anonymous functions that enable functional programming. They are used to write functional programming code.
List processing language (Lisp)	The functional programming language that was initially used in the 1950s. Today, there are many functional programming language options, including Scala, Python, R, and Java.
Loading or exporting the data	In the ETL pipeline's last step, data is exported to disk or loaded into another database. Also, users can write the data to the disk as a JSON file or save the data into another database, such as a Postgres (PostgresSQL) database. Users can also use an API to export data to a database, such as a Postgres database.
Local mode	Runs a Spark application as a single process locally on the machine. Executors are run as separate threads in the main process that calls 'spark-submit'. Local mode does not connect to any cluster or require configuration outside a basic Spark installation. Local mode can be run on a laptop. That's useful for testing or debugging a Spark application, for example, testing a small data subset to verify correctness before running the application on a cluster. However, being constrained by a single process means local mode is not designed for optimal performance.
Logging configuration	Spark Application configuration method in which Spark logging is controlled by the log4j defaults file, which dictates what level of messages, such as info or errors, are logged to the file or output to the driver during application execution.
Low latency data access	A type of data access allowing minimal delays, not noticeable to humans, between an input processed and corresponding output offering real-time characteristics. It is crucial for internet connections using trading, online gaming, and voice over IP.
Machine data	Refers to information generated by various sources, including Internet of Things (IoT) sensors embedded in industrial equipment as well as weblogs that capture user behavior and interactions.
Machine learning	A full-service cloud offering that allows developers and data scientists to collaborate and integrate predictive capabilities with their applications.
Map	MapReduce converts a set of data into another set of data and the elements are fragmented into tuples (key or value pairs).
map (*func*)	It is an essential operation capable of expressing all transformations needed in data science. It passes each element of the source through a function func, thereby returning a newly formed distributed data set.

Term	Definition
MapReduce	A program model and processing technique used in distributed computing based on Java. It splits the data into smaller units and processes big data. It is the first method used to query data stored in HDFS. It allows massive scalability across hundreds or thousands of servers in a Hadoop cluster.
Meta store	Stores the metadata, the data, and information about each table, such as the location and schema. The meta store, file system, and job client, in turn, communicate with Hive storage and computing to perform the following: Metadata information from tables is stored in some databases and query results, and data loaded from the tables is stored in a Hadoop cluster on HDFS.
Modular development	Techniques used in job designs to maximize the reuse of parallel jobs and components and save user time.
Multiple related jobs	Spark application can consist of many parallel and often related jobs, including multiple jobs resulting from multiple data sources, multiple DataFrames, and the actions applied to the DataFrames.
Node	A single independent system responsible for storing and processing big data. HDFS follows the primary and secondary concept.
NoSQL databases	NoSQL databases are built from the ground up to store and process vast amounts of data at scale and support a growing number of modern businesses. NoSQL databases store data in documents rather than relational tables. Types of NoSQL databases include pure document databases, key-value stores, wide-column databases, and graph databases such as MongoDB, CouchDB, Cassandra, and Redis.
Open Database Connectivity (ODBC) client	Component in the Hive client, which allows applications based on the ODBC protocol to connect to Hive.
Open-source software	Not only is the runnable version of the code free, but the source code is also completely open, meaning that every line of code is available for people to view, use, and reuse as needed.
Optimizer	Performs transformations on the execution and splits the tasks to help speed up and improve efficiency.
Parallel computing	A computing architecture in which multiple processors execute different small calculations fragmented from a large, complex problem simultaneously.
Parallel programming	It resembles distributed programming. It is the simultaneous use of multiple compute resources to solve a computational task. Parallel programming parses tasks into discrete parts solved concurrently using multiple processors. The processors access a shared pool of memory, which has control and coordination mechanisms in place.
Parallelization	Parallel regions of program code executed by multiple threads, possibly running on multiple processors. Environment variables determine the number of threads created and calls to library functions.
Parquet	Columnar format that is supported by multiple data processing systems. Spark SQL allows reading and writing data from Parquet files, and Spark SQL preserves the data schema.
Parser	A program that interprets the physical bit stream of an incoming message and creates an internal logical representation of the message in a tree structure. The parser also regenerates a bit stream for an outgoing message from the internal message tree representation.
Partitioning	This implies dividing the table into parts depending on the values of a specific column, such as date or city.
Persistent cache	Information is stored in "permanent" memory. Therefore, data is not lost after a system crash or restart, as if it were stored in cache memory.
Pig Hadoop component	Famous for its multi-query approach, it analyzes large amounts of data. It is a procedural data flow language and a procedural programming language that follows an order and set of commands.
Price analytics	Helps understand market segmentation, identify the best price points for a product line, and perform margin analysis for maximum profitability.
Primary node	Also known as the name node, it regulates file access to the clients and maintains, manages, and assigns tasks to the secondary node. The architecture is such that per cluster, there is one name node and multiple data nodes, the secondary nodes.
Properties	Spark Application configuration method in which Spark properties are used to adjust and control most application behaviors, including setting properties with the driver and sharing them with the cluster.
Python	Easy-to-learn, high-level, interpreted, and general-purpose dynamic programming language focusing on code readability. It provides a robust framework for building fast and scalable applications for z/OS, with a rich ecosystem of modules to develop new applications the same way you would on any other platform.
R	An open-source optimized programming language for statistical analysis and data visualization. Developed in 1992, it has a robust ecosystem with complex data models and sophisticated tools for data reporting.
Rack	The collection of about forty to fifty data nodes using the same network switch.
Rack awareness	When performing operations such as read and write, the name node maximizes performance by choosing the data nodes closest to themselves. This could be done by selecting data nodes on the same rack or nearby racks. It is used to reduce network traffic and improve cluster performance. To achieve rack awareness, the name node keeps the rack ID information.
RDD actions	It is used to evaluate a transformation in Spark. It returns a value to the driver program after running a computation. An example is the reduce action that aggregates the elements of an RDD and returns the result to the driver program.
RDD transformations	It helps in creating a new RDD from an existing RDD. Transformations in Spark are deemed lazy as results are not computed immediately. The results are computed after evaluation by actions. For example, map transformation passes each element of a data set through a function. This results in a new RDD.
Read	In this operation, the client will send a request to the primary node to acquire the location of the data nodes containing blocks. The client will read files closest to the data nodes.
Read the data	When reading the data, users can load data directly into DataFrames or create a new Spark DataFrame from an existing DataFrame.
Reduce	Job in MapReduce that uses output from a map as an input and combines data tuples into small sets of tuples.
Redundancy	Duplication of data across multiple partitions or nodes in a cluster. This duplication is implemented to enhance fault tolerance and reliability. If one partition or node fails, the duplicated data on other partitions or nodes can still be used to ensure that the computation continues without interruption. Redundancy is critical in maintaining data availability and preventing data loss in distributed computing environments like Spark clusters.
Region	The basic building element and most negligible unit of the HBase cluster, consisting of column families. It contains multiple stores, one for each column family and has two components: HFile and MemStore .
Region servers	These servers receive read and write requests from the client. They assign the request to a region where the column family resides. They serve and manage regions present in a distributed cluster. The region servers can communicate directly with the client to facilitate requests.
Relational database	Data is organized into rows and columns collectively, forming a table. The data is structured across tables, joined by a primary or a foreign key.
Relational Database Management System (RDBMS)	Traditional RDBMS is used to maintain a database and uses the structured query language known as SQL. It is suited for real-time data analysis, like data from sensors. It allows for as many read-and-write operations as a user may require. It can handle up to terabytes of data. It enforces that the schema must verify loading data before it can proceed. It may not always have built-in support for data partitioning.
Replication	The process of creating a copy of the data block. It is performed by rack awareness as well. It is done by ensuring data node replicas are in different racks. So, if a rack is down, users can obtain the data from another rack.

Term	Definition
Replication factor	Defined as the number of times you make a copy of the data block. Users can set the number of copies they want, depending on their configuration.
Resilient Distributed Datasets (RDDs)	A fundamental abstraction in Apache Spark that represents distributed collections of data. RDDs allow you to perform parallel and fault-tolerant data processing across a cluster of computers. RDDs can be created from existing data in storage systems (like HDFS), and they can undergo various transformations and actions to perform operations like filtering, mapping, and aggregating. The "resilient" aspect refers to resilient distributed datasets (RDDs) ability to recover from node failures, and the "distributed" aspect highlights their distribution across multiple machines in a cluster, enabling parallel processing.
Scala	A general-purpose programming language that supports both object-oriented and functional programming. The most recent representative in the family of programming languages. Apache Spark is written mainly in Scala, which treats functions as first-class citizens. Functions in Scala can be passed as arguments to other functions, returned by other functions, and used as variables.
Scalability	The ability of a system to take advantage of additional resources, such as database servers, processors, memory, or disk space. It aims at minimizing the impact on maintenance. It is the ability to maintain all servers efficiently and quickly with minimal impact on user applications.
Schema	It is a collection of named objects. It provides a way to group those objects logically. A schema is also a name qualifier; it provides a way to use the same natural name for several objects and to prevent ambiguous references to those objects.
Secondary node	This node is also known as a data node. There can be hundreds of data nodes in the HDFS that manage the storage system. They perform read and write requests at the instructions of the name node. They also create, replicate, and delete file blocks based on instructions from the name node.
Semi-structured data	Semi-structured data (e.g., JSON, CSV, XML) is the "bridge" between structured and unstructured data. It does not have a predefined data model and is more complex than structured data, yet easier to store than unstructured data.
Sentiment analysis	Utilizes social media conversations to gain insights into consumer opinions about a product. It is used to develop effective marketing strategies and establish customer connections based on their sentiments and preferences.
Serialization	Required to coordinate access to resources that are used by more than one program. An example of why resource serialization is needed occurs when one program is reading from a data set and another program needs to write to the data set.
Shuffle	Phase in which interim map output from mappers is transferred to reducers. Every reducer fetches interim results for all values associated with the same key from multiple nodes. This is a network-intensive operation within the Hadoop cluster nodes.
Social data	Comes from the likes, tweets and retweets, comments, video uploads, and general media that are uploaded and shared via the world's favorite social media platforms. Machine-generated data and business-generated data are data that organizations generate within their own operations.
Spark Application	A Spark application refers to a program or set of computations written using the Apache Spark framework. It consists of a driver program and a set of worker nodes that process data in parallel. Spark applications are designed for distributed data processing, making them suitable for big data analytics and machine learning tasks.
Spark Cluster Manager	Communicates with a cluster to acquire resources for an application to run. It runs as a service outside the application and abstracts the cluster type. While an application is running, the Spark Context creates tasks and communicates to the cluster manager what resources are needed. Then the cluster manager reserves executor cores and memory resources. Once the resources are reserved, tasks can be transferred to the executor processes to run.
Spark Configuration Location	Located under the "conf" directory in the installation. By default, there are no preexisting files after installation, however, Spark provides a template for each configuration type with the filenames shown here. Users can create the appropriate file by removing the '.template' extension. Inside the template files are sample configurations for standard settings. They can be enabled by uncommenting.
Spark Context	Communicates with the Cluster Manager. It is defined in the Driver, with one Spark Context per Spark Application.
Spark Core	Often popularly referred to as "Spark." The fault-tolerant Spark Core is the base engine for large-scale parallel and distributed data processing. It manages memory and task scheduling. It also contains the APIs used to define RDDs and other datatypes. It parallelizes a distributed collection of elements across the cluster.
Spark data persistence	Also known as caching data in Spark. Ability to store intermediate calculations for reuse. This is achieved by setting persistence in either memory or both memory and disk. Once intermediate data is computed to generate a fresh DataFrame and cached in memory, subsequent operations on the DataFrame can utilize the cached data instead of reloading it from the source and redoing previous computations. This feature is crucial for accelerating machine learning tasks that involve multiple iterations on the same data set during model training.
Spark driver program	A program that functions as software situated on the primary node of a machine. It defines operations on RDDs, specifying transformations and actions. To simplify, the Spark driver initiates a SparkContext linked to a designated Spark Master. Furthermore, it transfers RDD graphs to the Master, the location from which the stand-alone cluster manager operates.
Spark History server	Web UI where the status of running and completed Spark jobs on a provisioned instance of Analytics Engine powered by Apache Spark, is displayed. If users want to analyze how different stages of the Spark job are performed, they can view the details in the Spark history server UI.
Spark jobs	Computations that can be executed in parallel. The Spark Context divides Jobs into Tasks to be executed on the Cluster.
Spark logging	Controlled using log4j and the configuration is read through "conf/log4j-properties." Users can adjust a log level to determine which messages (such as debug, info, or errors) are shown in the Spark logs.
Spark memory management	Spark memory stores the intermediate state while executing tasks such as joining or storing broadcast variables. All the cached and persisted data will be stored in this segment, specifically in the storage memory.
Spark ML	Spark's machine learning library for creating and using machine learning models on large data sets across distributed clusters.
Spark RDD persistence	Optimization technique that saves the result of RDD evaluation in cache memory. Using this technique, the intermediate result can be saved for future use. It reduces the computation overhead.
Spark Shell	Available for Scala and Python, giving you access to Spark APIs for working with data as Spark jobs. Spark Shell can be used in local or cluster mode, with all options available.
Spark Shell Environment	When Spark Shell starts, the environment automatically initializes the SparkContext and SparkSession variables. This means you can start working with data immediately. Expressions are entered in the shell and evaluated in the driver. Entering an action on a shell DataFrame generates Spark jobs that are sent to the cluster to be scheduled as tasks.
Spark Shuffle	Performed when a task requires other data partitions. It marks the boundary between stages.
Spark SQL memory optimization	The primary aim is to improve the run-time performance of a SQL query by minimizing the query time and memory consumption, thereby helping organizations save time and money.
Spark SQL	A Spark module for structured data processing. Users can interact with Spark SQL using SQL queries and the DataFrame API. Spark SQL supports Java, Scala, Python, and R APIs. Spark SQL uses the same execution engine to compute the result independently of the API or language used for computation. Developers can use the API to help express a given transformation. Unlike the basic Spark RDD API, Spark SQL includes a cost-based optimizer, columnar storage, and code generation to perform optimizations that equip Spark with information about the structure of data and the computation in process.
Spark Stages	Represents a set of tasks an executor can complete on the current data partition. Subsequent tasks in later stages must wait for that stage to be completed before beginning execution, creating a dependency from one stage to the next.

Term	Definition
Spark Standalone	Included with the Spark installation. It is best for setting up a simple cluster. There are no additional dependencies required to configure and deploy. Spark Standalone is specifically designed to run Spark and is often the fastest way to get a cluster up and running applications.
Spark Standalone cluster	Has two main components: Workers and the Master. The workers run on cluster nodes. They start an executor process with one or more reserved cores. There must be one master available which can run on any cluster node. It connects workers to the cluster and keeps track of them with heartbeat polling. However, if the master is together with a worker, do not reserve all the node's cores and memory for the worker.
Spark tasks	Tasks from a given job operate on different data subsets called partitions and can be executed in parallel.
SparkContext	When a Spark application is being run, as the driver program creates a SparkContext, Spark starts a web server that serves as the application user interface. Users can connect to the UI web server by entering the hostname of the driver followed by port 4040 in a browser once that application is running. The web server runs for the duration of the Spark application, so once the SparkContext stops, the server shuts down, and the application UI is no longer accessible.
Spark-submit	Spark comes with a unified interface for submitting applications called the 'spark-submit' script found in the 'bin/' directory. 'Spark-submit' can be used for all supported cluster types and accepts many configuration options for the application or cluster. 'Unified interface' means you can switch from running Spark in local mode to cluster by changing a single argument. 'Spark-submit' works the same way, irrespective of the application language. For example, a cluster can run Python and Java applications simultaneously by passing in the required files.
SQL Procedural code	A set of instructions written in a programming language within an SQL database environment. This code allows users to perform more complex tasks and create custom functions, procedures, and control structures, enabling them to manipulate and manage data in a more controlled and structured manner.
SQL queries in Spark SQL	Spark SQL allows users to run SQL queries on Spark DataFrames.
Sqoop	An open-source product designed to transfer bulk data between relational database systems and Hadoop. It looks in the relational database and summarizes the schema. It generates MapReduce code to import and export data. It helps develop any other MapReduce applications that use the records stored in HDFS.
Stages tab	Displays a list of all stages in the application, grouped by the current state of either completed, active, or pending. This example displays three completed stages. Click the Stage ID Description hyperlinks to view task details for that stage.
Static configuration	Settings that are written programmatically into the application. These settings are not usually changed because they require modifying the application itself. Use static configuration for something that is unlikely to be changed or tweaked between application runs, such as the application name or other properties related to the application only.
Storage tab	Displays details about RDDs that have been cached or persisted to memory and written to disk.
Streaming	Implies HDFS provides a constant bitrate when transferring data, rather than having the data transferred in waves.
Streaming analytics	Help leverage streams to ingest, analyze, monitor, and correlate data from real-time data sources. They also help to view information and events as they unfold.
String data type	It is the IBM® Informix® ESQL/C data type that holds character data that is null-terminated and does not contain trailing blanks.
Structured data	Structured data, typically categorized as quantitative data, is highly organized and easily decipherable by machine learning algorithms. Developed by IBM in 1974, structured query language (SQL) is the programming language used to manage structured data.
Syntax error	If this error is detected while processing a control statement, the remaining statement is skipped and not processed. Any operands in the portion of the statement preceding the error are processed.
toDS() function	Converts data into a typed data set for efficient and type-safe operations in PySpark.
Transactional Data	Generated from all the daily transactions that take place both online and offline, such as invoices, payment orders, storage records, and delivery receipts.
Transform the data	In this step of the ETL pipeline, users plan for required data set transformations, if any. The transformation aims at retaining only the relevant data. Transformation techniques include data filtering, merging with other data sources, or performing columnar operations. Columnar operations include actions such as multiplying each column by a specific number or converting data from one unit to another. Transformation techniques can also be used to group or aggregate data. Many transformations are domain-specific data augmentation processes. The effort needed varies with the domain and the data.
Tungsten	Catalyst and Tungsten are integral components of Spark's optimization and execution framework. Tungsten is geared toward enhancing both CPU and memory performance within Spark. Unlike Java, which was initially designed for transactional applications, it seeks to bolster these aspects by employing methods more tailored to data processing within the Java Virtual Machine (JVM). To achieve optimal CPU performance, it also adopts explicit memory management, employs cache-friendly data structures through STRIDE-based memory access, supports on-demand JVM bytecode, minimizes virtual function dispatches, and capitalizes on CPU register placement and loop unrolling.
Uber-JAR	An Uber-JAR is a single Java Archive (JAR) file that contains not only the application code but also all its dependencies, including transitive ones. The purpose of an Uber-JAR is to create a self-contained package that can be easily transported and executed within a computing cluster or environment.
Unified memory	Unified regions in Spark shared by executor memory and storage memory. If executor memory is not used, storage can acquire all the available memory, and vice versa. If the total storage memory usage falls under a certain threshold, executor memory can discard storage memory. Due to complexities in implementation, storage cannot evict executor memory.
Unstructured data	Information lacking a predefined data model or not fitting into relational tables.
User code	Made up of the driver program, which runs in the driver process, and the functions and variables serialized that the executor runs in parallel. The driver and executor processes run the application user code of an application passed to the Spark-submit script. The user code in the driver creates the SparkContext and creates jobs based on operations for the DataFrames. These DataFrame operations become serialized closures sent throughout the cluster and run on executor processes as tasks. The serialized closures contain the necessary functions, classes, and variables to run each task.
Variety	The diversity of data or the various data forms that need to be stored. Variety is one of the four main components used to describe the dimensions of big data.
Velocity	The speed at which data arrives. Velocity is one of the four main components used to describe the dimensions of big data.
Veracity	The certainty of data, as with a large amount of data available, makes it difficult to determine if the data collected is accurate. Veracity is one of the four main components used to describe the dimensions of big data.
Volume	The increase in the amount of data stored over time. Volume is one of the four main components used to describe the dimensions of big data.
Worker	Cluster node that can launch executor processes to run tasks.
Worker node	A unit in a distributed system that performs tasks and processes data according to instructions from a central coordinator.
Workflows	Include jobs created by SparkContext in the driver program. Jobs in progress run as tasks in the executors, and completed jobs transfer results back to the driver or write to disk.
Write	In this operation, the Name node ensures that the file does not exist. If the file exists, the client gets an IO Exception message. If the file does not exist, the client is given access to start writing files.

Term	Definition
Yet Another Resource Negotiator (YARN)	Serves as the resource manager bundled with Hadoop and is typically the default resource manager for numerous big data applications, such as HIVE and Spark. While it remains a robust resource manager, it's important to note that more contemporary container-based resource managers, such as Kubernetes, are gradually emerging as the new standard practices in the field.

Author(s)

- Niha Ayaz Sultan
- Rashi Kapoor

Skills Network