Hadoop for Finance Essentials

With the exponential growth of data and many enterprises crunching more and more data every day, Hadoop as a data platform has gained a lot of popularity. Financial businesses want to minimize risks and maximize opportunities, and Hadoop, largely dominating the big data market, plays a major role.

This book will get you started with the fundamentals of big data and Hadoop, enabling you to get to grips with solutions to many top financial big data use cases including regulatory projects and fraud detection. It is packed with industry references and code templates, and is designed to walk you through a wide range of Hadoop components.

By the end of the book, you'll understand a few industry leading architecture patterns, big data governance, tips, best practices, and standards to successfully develop your own Hadoop based solution.

Who this book is written for

This book is perfect for developers, analysts, architects or managers who would like to perform big data analytics with Hadoop for the financial sector. This book is also helpful for technology professionals from other industry sectors who have recently switched or like to switch their business domain to financial sector. Familiarity with big data, Java programming, database and data warehouse, and business intelligence would be beneficial.

What you will learn from this book

- Learn about big data and Hadoop fundamentals including practical finance use cases
- Walk through Hadoop-based finance projects with explanations of solutions, big data governance, and how to sustain Hadoop momentum
- Develop a range of solutions for small to large-scale data projects on the Hadoop platform
- Learn how to process big data in the cloud
- Present practical business cases to management to scale up existing platforms at enterprise level

Prices do not include local sales tax or VAT
In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 1 'Big Data Overview'
- A synopsis of the book’s content
- More information on Hadoop for Finance Essentials

About the Author

Rajiv Tiwari is a hands-on freelance big data architect with over 15 years of experience across big data, data analytics, data governance, data architecture, data cleansing / data integration, data warehousing, and business intelligence for banks and other financial organizations.

He is an electronics engineering graduate from IIT, Varanasi, and has been working in England for the past 10 years, mostly in the financial city of London, UK.

He has been using Hadoop since 2010, when Hadoop was in its infancy with regards to the banking sector.

He is currently helping a tier 1 investment bank implement a large risk analytics project on the Hadoop platform.

Rajiv can be contacted on his website at http://www.bigdatacloud.net or on Twitter at @bigdataoncloud.
Hadoop for Finance Essentials

Data has been increasing at an exponential rate and organizations are either struggling to cope up or rushing to take advantage by analyzing it. Hadoop is an excellent open source framework, which addresses this big data problem.

I have used Hadoop within the financial sector for the last few years but could not find any resource or book that explains the usage of Hadoop for finance use cases. The best books I have ever found are again on Hadoop, Hive, or some MapReduce patterns, with examples on counting words or Twitter messages in all possible ways.

I have written this book with the objective of explaining the basic usage of Hadoop and other products to tackle big data for finance use cases. I have touched base on the majority of use cases, providing a very practical approach.

What This Book Covers

Chapter 1, Big Data Overview, covers the overview of big data, its landscape, and technology evolution. It also touches base with the Hadoop architecture, its components, and distributions. If you know Hadoop already, just skim through this chapter.

Chapter 2, Big Data in Financial Services, extends the big data overview from the perspective of a financial organization. It will explain the story of the evolution of big data in the financial sector, typical implementation challenges, and different finance use cases with the help of relevant tools and technologies.

Chapter 3, Hadoop in the Cloud, covers the overview of big data in cloud and a sample portfolio risk simulation project with end-to-end data processing.

Chapter 4, Data Migration Using Hadoop, talks about the most popular project of migrating historical trade data from traditional data sources to Hadoop.

Chapter 5, Getting Started, covers the implementation project of a very large enterprise data platform to support various risk and regulatory requirements.

Chapter 6, Getting Experienced, gives an overview of real-time analytics and a sample project to detect fraudulent transactions.

Chapter 7, Scale It Up, covers the topics to scale up the usage of Hadoop within your organization, such as enterprise data lake, lambda architecture, and data governance. It also touches base with few more financial use cases with brief solutions.

Chapter 8, Sustain the Momentum, talks about the Hadoop distribution upgrade cycle and wraps up the book with best practices and standards.
Everything an organization or an individual does has a digital footprint that can be used or analyzed. Quite simply, big data refers to the usage of an ever increasing volume of data.

In this chapter, we will provide an overview of big data and Hadoop with the help of following topics:

• What is big data?
• Big data technology evolution
• Big data landscape
• Careers in big data
• Hadoop architecture
• The Hadoop jungle explained
• Hadoop distributions

**What is big data?**

There are many definitions of big data provided by different consultancies and IT providers. Here is a shortlist of two of the best definitions.

The first and the coolest definition is "data that is so big it cannot be processed using traditional data processing tools and applications".

Second, the most professionally accepted definition is the famous 3V one—"3Vs (volume, variety, and velocity) are the three defining properties or dimensions of big data. Volume refers to the amount of data, variety refers to the number of types of data, and velocity refers to the speed of data processing."
Big Data Overview

Data volume
There is an ongoing debate on data volume—exactly what can be classified as big data? But unfortunately there is no such defined rule to classify it. For a small company that deals with gigabytes of data, terabytes are considered big. For a large company that deals with terabytes of data, petabytes are considered big. But, in any case, we are talking of data at least more than a terabyte.

The size of data is growing at an exponential rate for both individuals and organizations. Also, no one wants to throw away data, especially when the price of hard disks has been dropping consistently. In addition to a desire to store endless data, corporations also want to analyze it. The data is in different forms—structured data such as massive transactional data, semistructured data such as documents, and unstructured data such as images and videos.

Data velocity
Until around 5 years ago, organizations used to extract, transform, and load (ETL) large amounts of data in daily batches into data warehouses or data marts with business intelligence and analytics tools on top of that. Now, with more real-time data sources such as messages, social media, and transactions, the processing of data in daily batches does not add much value. Business is now much more competitive and increasingly online. The increased speed of data generation and business needs has increased data velocity to as fast as real time.

With memory and hard disks getting cheaper and machines faster as every single day passes, the expectation to see real-time data analytics has never been greater than it is now.

Data variety
Earlier, data was mostly in the form of structured databases, which were relatively easier to process using traditional data integration and analytics tools.

But now businesses want to process heterogeneous types of data—anything from Excel tables and databases to pure text, pictures, videos, web data, GPS data, sensor data, documents, mobile messages, or even papers that can be scanned and transformed to electronic format.

Organizations must adapt to the new data formats and be able to process and analyze them.
Big data technology evolution
We will tell you the big data evolution story with Hadoop as the central theme.

History
Although it may feel like Hadoop is just a few years old going by the market buzz, in truth it is more than 10 years old. In 2003, Yahoo was doing a project called Nutch to address their web-search performance and scalability issue.

In 2003 and 2004, Google published its paper on Google's filesystem and MapReduce, respectively.

Nutch's developers started making the change using the design from Google's papers, which helped the process of scaling to more than just one machine. The solution worked to a very large extent, but still it wasn't a web-scale solution.

In 2006, Yahoo hired Doug Cutting to improve the web-search solution. He liaised with Apache to develop an open source project and took out the storage and processing parts from Nutch to create a generic reusable framework. Interestingly, Doug named the open source project Hadoop after his son's toy elephant and even he couldn't have thought that the name would become the top buzzword in the world of information technology a few years later. Doug's team improved the Hadoop framework and released a truly web-scale solution in 2008.

Yahoo kept on migrating its applications onto Hadoop and by 2011 it was running its search engine on 42,000 nodes containing hundreds of petabytes of data storage.

Current
Almost every large web giant, such as Facebook, Google, Microsoft, and eBay, is exploiting web data using Hadoop as its framework. According to an IDC survey, 63.3 percent of all organizations have either deployed or are planning to deploy Hadoop in the next year. The remaining 36.6 percent are planning to implement it as well but may take more than a year to do so.

There are numerous Hadoop distribution companies such as Cloudera and HortonWorks that make the deployment and management of Hadoop a lot easier. They also provide software support and help in the integration of Hadoop into an organization's IT landscape.
Future
IDC research forecasts data storage will see the biggest growth at 53.4 percent compound annual growth rate for at least another few years.

They also forecast that the big data technology and services market will grow at 27 percent compound annual growth rate to $32.4 billion through 2017—about six times the growth rate of the overall information and communication technology market.

At the moment, there are at least dozens of start-ups with very unique products and they have secured hundreds of millions of venture capital.

However you define it, though, there's no denying that big data will be a huge market and will attract the majority of technology expenditure for an organization.

The big data landscape
We will discuss the big data components responsible for functions such as storage, resource management, governance, processing, and analysis. Most of these big data components are packaged into an enterprise-grade-supported Hadoop distribution, which will be discussed later in more detail.
Chapter 1

Storage
Data storage is where your raw data lives. It is a reliable, fault-tolerant distributed filesystem that contains structured and unstructured data.

The data is stored either on a distributed on-premise filesystem, such as Hadoop Distributed Filesystem (HDFS), or a cloud-based system, such as Amazon S3. The data is also stored in NoSQL databases, such as HBase or Cassandra on Hadoop storage.

To move data into a big data ecosystem, there are data integration tools, such as Flume and Sqoop, or web-service interfaces, such as REST and SOAP for Amazon S3.

NoSQL
NoSQL databases are non-relational, distributed, generally open source, and horizontally scalable.

The term NoSQL is slightly misleading as NoSQL databases do support SQL operations. The term is now popularly defined as Not Only SQL.

Some other characteristics are schema-free, easy replication support, integrated caching, simple API, and eventually consistent/BASE (not ACID).

In comparison to relational databases, NoSQL databases have superior performance and are more scalable if you need to handle the following:

- Large volumes of structured, semistructured, and unstructured data
- Quick iterative requirements involving frequent code changes
- High growth business requiring efficient and scale-out architecture

NoSQL database types
There are mainly four types of NoSQL databases:

- Document databases: They pair each key with a document. Documents can be simple or complex data structures and contain key-value pairs, key-array pairs, or even nested documents; examples of these databases are MongoDB and CouchDB.
- Graph databases: They are used to store graphical relationships such as social connections; examples of these databases are Neo4j and HyperGraphDB.
Big Data Overview

- **Key-value databases**: They are the simplest of all NoSQL-type databases. Every single item is stored as an attribute name (or key) and corresponding value. Some key-value stores, such as Redis, allow each value to have a data type such as integer, which may be useful. Document databases can also function as a key-value database, but they are more targeted at complex structures; examples of these databases are Riak and Voldemort.

- **Wide-column databases**: They are highly optimized for queries over large datasets. They store columns of data together instead of rows, such as Cassandra and HBase.

It is a flooded market with at least dozens of NoSQL vendors, each claiming superiority over the others. Most of the databases within their type follow similar architecture and development methodologies and so it is not uncommon that organizations stick to a few of the vendors only.

**Resource management**

Effective resource management is a must, especially when there will be multiple applications running, fighting for the computing and data resources. Resource Managers such as YARN and Mesos manage allocation, de-allocation, and efficient utilization of your compute/data resources.

There is also a collection of tools to manage the workflow, provisioning, and distributed coordination; for example, Oozie and Zookeeper.

**Data governance**

Data governance is all about taking care of the data, ensuring that the metadata (information about data) is recorded properly and the data is only accessed by authorized people and systems. There is a collection of tools to manage metadata, authentication, authorization, security, privacy settings, and lineage. The tools used are Apache Falcon, HCatalog, Sentry, and Kerberos.

**Batch computing**

Batch computing is an efficient way to process large data volumes, as the data is collected over a period of time and then processed. The MapReduce, Pig, and Hive scripts are used for batch processing.

Typically, a batch process will have persistent-disk-based data storage for input, output, and intermediate results.

Examples include end-of-day risk metrics calculation and historical trade analytics.
Chapter 1

Real-time computing
Real-time computing is low-latency data processing and usually a subsecond response. Spark and Storm are the popular ones for real-time processing.

Typically, a batch process will have in-memory processing with continuous data input, but it doesn't necessarily need persistent storage for output and intermediate results.

Examples include live processing of tweets and stock prices, fraud detection, and system monitoring.

Data integration tools
Data integration tools bring data in and out of Hadoop ecosystems. In addition to tools provided by Apache or Hadoop distributions, such as Flume and Sqoop, there are many premium vendors such as Informatica, Talend, Syncsort, and Datameer, which are a one-stop shop for the majority of data integration needs. As many data integration tools are user-interface-based and use fourth-generation languages, they are easy to use and will keep your Hadoop ecosystem free from complicated MapReduce low-level programs.

Machine learning
Machine learning is the development of models and algorithms that learn from the input data and improve it based on feedback. The program is driven by the input data and doesn't follow explicit instructions. The most popular suite of machine learning libraries is from Mahout, but it's not uncommon to program using the Spark MLlib library or the Java-based custom MapReduce.

A few examples are speech recognition, anomaly or fraud detection, forecasting, and recommending products.

Business intelligence and virtualization
Hadoop distributions can be combined with different business intelligence and data visualization vendors that can connect to the underlying Hadoop platform to produce management and analytical reports.

There are many vendors in this space and nowadays almost all leading BI tools provide connectors to Hadoop platforms. The most trendy ones are Tableau, SAS, Splunk, Qlikview, and Datameer.
Careers in big data

Gartner predicted that the roles directly related to big data are predicted to be around 4.4 million across the world by 2015, which could be anything from basic server maintenance to high-end data science innovation.

As more companies want to make use of analytics on their big data, it looks more likely that the number of roles will only increase going forward. 2015 is likely to see a marked increase in the number of people involved within big data directly as the market has started maturing and now more businesses have proven business benefits.

The technical skills required are very diverse—server maintenance, low-level MapReduce programming, NoSQL database administration, data analytics, graphic designers for visualizations, data integration, data science, machine learning, and business analysis. Even the non-technical roles—project management, front office staff such as finance, trading, marketing, and sales teams to analyze the results—will need retraining with the usage of the new set of analytics and visualization tools.

There are very few people with skills on big data or Hadoop and the demand is very high. So, obviously, they are generally paid higher than the market average. The pay packages of Hadoop roles will be better if:

- The skills are new and scarce
- The skills are comparably difficult to learn
- Technical skills such as MapReduce are combined with business domain knowledge

The skills required are so diverse that you can choose a career in a subject that you are passionate about.

Some of the most popular job roles are:

- **Developers**: They develop MapReduce jobs, design NoSQL databases, and manage Hadoop clusters. Job role examples include Hadoop, MapReduce, or Java developer and Hadoop administrator.

- **Architects**: They have the breadth and depth of knowledge across a wide range of Hadoop components, understand the big picture, recommend the hardware and software to be used, and lead a team of developers and analysts if required. Job role examples include big data architect, Hadoop architect, senior Hadoop developer, and lead Hadoop developer.

- **Analysts**: They apply business knowledge with mathematical and statistical skills to analyze data using high-level languages such as Pig, Hive, SQL, and BI tools. Job role examples include business analysts and business intelligence developers.
• **Data scientists**: They have the breadth and depth of knowledge across a wide range of analytical and visualization tools, possess business knowledge, understand the Hadoop ecosystem, know a bit of programming, but have expertise in mathematics and statistics and have excellent communication skills.

People with development skills in Java, C#, relational databases, and server administration will find it easier to learn Hadoop development, get hands-on with a few projects, and choose to either specialize in a tool or programming language. They can also learn more skills across a variety of different components and take on architect/technical lead roles.

People with analysis skills who already understand the business processes and integration with technology can learn a few high-level programming languages such as Pig, R, Python, or BI tools. Experience in BI tools and high-level programming works best with good business domain knowledge.

Although I am trying here to divide this into two simple career paths—development and analysis—in the real world there is a significant overlap in all the mentioned job roles.

As long as you have excellent development and analysis skills and are also ready to learn mathematics and business (either formal education or experience), there is nothing stopping you from becoming the "data scientist"—claimed to be the sexiest job for the 21st century by Harvard Business Review.

### Hadoop architecture

This section will not provide in-depth knowledge of the Hadoop architecture, but only a high-level overview so we can understand the following chapters without much difficulty.

```
[ 🧠 For detailed knowledge on this subject, I recommend a study of the book Hadoop: The Definitive Guide, Tom White, O'Reilly Media. ]
```

### HDFS cluster

• A node is just a computer containing data that is based on nonenterprise, inexpensive commodity hardware. So, in the following figure, we have **Node 1, Node 2, Node 3**, and so on.
A rack is normally a collection of 10 or more nodes physically stored together and connected to the same network switch. So, network latency between any two nodes in a rack is lower than the latency between two nodes on different racks.

A cluster is a collection of racks as shown in the following figure:

- HDFS is designed to be fault tolerant through replication of data. If one component fails, another copy will be used.
- A file from a non-Hadoop system will be broken into multiple blocks and each block will be replicated on HDFS in a distributed form. Hadoop works best with very large files, as it is designed for streaming or sequential data access rather than random access.
- Blocks have a default size of 128 megabytes each since Hadoop 2.1.0. Newer versions of Hadoop are moving towards higher default block sizes. The block size can be configured depending on an application's needs.
- A 350 megabyte file with a 128 megabyte block size consumes three blocks, but the third block does not consume a full 128 megabytes. The blocks fit well with the replication design, which allows HDFS to be fault tolerant and available on commodity hardware.
• As shown in the following figure, each block is replicated to multiple nodes. For example, Block 1 is stored on Node 1 and Node 2. Block 2 is stored on Node 1 and Node 3. And Block 3 is stored on Node 2 and Node 3. This allows for node failure without data loss. If Node 1 crashes, Node 2 still runs and has Block 1’s data. In this example, we are only replicating data across two nodes, but you can set replication to be across many more nodes by changing Hadoop’s configuration or even setting the replication factor for each individual file.

MapReduce V1
Before Hadoop Version 2.2, MapReduce was referred to as MapReduce V1 and had a different architecture:

• MapReduce programs are of two types—a map transformation or a reduce transformation
• A MapReduce job will be executing as parallel tasks on different nodes–the map job first, and then the results of the map job will be processed by the reduce job
There are two main types of nodes. They are classified as HDFS or MapReduce nodes:

- An HDFS node is either a NameNode or DataNode.
- A MapReduce node is either a JobTracker or a TaskTracker.
- The client generally communicates with a JobTracker. However, it can also communicate with the NameNode or any of the DataNodes.
- There is only one NameNode in the cluster. The DataNodes store all the distributed blocks, but the metadata for those files are stored on the NameNode. This means that NameNode is a single point of failure and therefore deserves the best enterprise hardware for maximum reliability. As NameNode keeps the entire filesystem metadata in memory, it is recommended to buy as much RAM as possible.
- An HDFS cluster has many DataNodes. The DataNodes store blocks of data. When a client requests to read a file from HDFS, the client finds out from the NameNode which blocks make up that file and on which DataNodes those blocks are stored. The client can now directly read the blocks from the corresponding DataNodes. These DataNodes are inexpensive enterprise hardware and the replication is provided in the software layer instead of the hardware layer.
- As NameNode, there is only one JobTracker on the cluster that manages all MapReduce jobs submitted by clients. It schedules the Map and Reduce tasks using TaskTrackers on the DataNodes itself. Basically, the task is executed very intelligently using the locations of data blocks via rack-awareness. Rack-awareness is a configuration that means the data blocks should be processed on the same rack if possible. The block is also replicated on a different rack to handle rack failures. The JobTracker monitors the progress of TaskTrackers and reschedules failed TaskTrackers on a different replicated DataNode.
MapReduce V2 – YARN

MapReduce V1 dominated the big data landscape for many years, but there were a few limitations, such as:

- **Extreme scalability**: It could not accommodate extreme cluster sizes of more than 4,000 nodes or 40,000 concurrent tasks
- **Availability**: NameNode and JobTracker were the single point of failure
- **Jobs**: Only MapReduce jobs were supported

MapReduce V2 came up with architectural changes to fix these limitations. This new architecture is also called *Yet Another Resource Negotiator (YARN)*. However, it is not mandatory to run YARN on MapReduce V2.
The new architecture has DataNodes but has no TaskTrackers and a JobTracker. MapReduce V1 is supported on V2, as V1 is still widely used by organizations across the world.

YARN works on two fundamentals:

- Generic scheduling and resource management, not just MapReduce.
- Efficient scheduling and workload management. The resource manager is now aware of the capabilities of each node via communication with the NodeManager running on each node.

As shown in the following figure, when an application gets invoked by the client, an Application Master gets started on a NodeManager. The Application Master is then responsible for negotiating resources with the ResourceManager. These resources are assigned to containers on each slave node and the tasks are run in the containers.

The Hadoop architecture is now modified to support high availability of NameNode, which is a key requirement for any business-critical application. There are now two NameNodes, one active and one standby.

As shown in the following figure, there are JournalNodes. For a basic setup of one active and one standby NameNode, there are three JournalNodes. As expected, only one of the NameNodes is active at a time. The JournalNodes work together to decide which of the NameNodes is to be the active one. If, for some reason, the active NameNode has gone down, the backup NameNode will take over.
Hadoop has been improved further to provide extreme scalability. There are multiple NameNodes acting independently. Each NameNode has its own namespace and therefore has control over its own set of files. However, they share all of the DataNodes.
MapReduce V2, like V1, has awareness of the topology of the network. When rack-awareness is configured for your cluster, Hadoop will always try to run the task on the TaskTracker node with the highest bandwidth access to the data.

**The Hadoop jungle explained**

In this section, we will briefly explain the components on Hadoop and point you to related study materials. Many of these components will be referred to throughout the book.

**Big data tamed**

We know that big data is everywhere and that it needs to be processed and analyzed into something meaningful. But how do we process the data without breaking the bank?

Hadoop, the hero, can tame this big data monster because of the following features:

- It is enterprise grade but runs on commodity servers. Storing big data on traditional database storage is very expensive, generally in the order of $25,000 to $50,000 per terabyte per year. However, with Hadoop, the cost of storing data on commodity servers drops by 90 percent to the order of $2,500 to $5,000 per year.

- It scales horizontally; data can be stored for a longer duration if required, as there is no need to get rid of older data if you run out of disk space. We can simply add more machines and keep on processing older data for long-term trend analytics. As it scales linearly, there is no performance impact when data grows and there is never a need to migrate to bigger platforms.

- It can process semistructured/unstructured data such as images, log files, videos, and documents with much better performance and lower cost, which wasn't the case earlier.

- Processing has moved to data rather than large data moved to the computing engine. There is minimal data movement and processing logic can be written in a variety of programming languages such as Java, Scala, Python, Pig, and R.
Hadoop – the hero

The core of Hadoop is a combination of HDFS and MapReduce—HDFS for data storage and MapReduce for data computing.

But Hadoop is not just about HDFS and MapReduce. The little elephant has a lot of friends in the jungle who help in different functions of the end-to-end data life cycle. We will talk briefly on popular members of the jungle in subsequent sections.

HDFS – Hadoop Distributed Filesystem

HDFS is about unlimited reliable distributed storage. That means that, once you have established your storage and then your business grows, the additional storage can be handled by adding more inexpensive commodity servers or disks. Now, you may wonder whether it might fail occasionally if the servers are not high end. The answer is yes; in fact, on a large cluster of say 10,000 servers, at least one will fail every day. Still, HDFS is self-healing and extremely fault tolerant as every piece of data is replicated on different node servers (the default is three nodes). So, if one data node fails, either of the other two can be used. Also, if a DataNode fails or a block gets corrupted or missed, the failed DataNode's data or block is replicated to a live, light-weight data node.

MapReduce

MapReduce is about moving the computing onto data nodes in a distributed form. It can read data from HDFS as well as other data sources such as mounted filesystems and databases. It is extremely fault tolerant. If any of the computing jobs on the node fails, Hadoop reassigns the job to another copy of the data node. Hadoop knows how to clean up the failed job and combine the results from various MapReduce jobs.

The MapReduce framework processed data in two phases—Map and Reduce. Map is used to filter and sort the data and Reduce is used to aggregate the data. We will show you the famous word count example, which is the best way to understand the MapReduce framework:

- The input data is split into multiple blocks in HDFS.
- Each block is fed to Map for processing and the output is a key-value pair of word as key and count as the value.
- The key-value pairs are sorted by key and partitioned to be fed to Reduce for processing. The key-value pair with the same keys will be processed with the same Reducer.
- Reduce will sum the count of each word and output it as a key-count pair with word as key and count as the sum.
Another example of MapReduce at work is finding the customers with the highest number of transactions for a certain period, as portrayed in the following figure:

![MapReduce diagram](image)

**HBase**

HBase is a wide-column database, one of the NoSQL database types. The database rows are stored as key-value pairs and don’t store null values. So, it is very suitable for sparse data in terms of storage efficiency. It is linearly scalable and provides real-time random read-write access to data stored on HDFS.


**Hive**

Hive is a high-level declarative language for writing MapReduce programs for all the data-warehouse-loving people who like to write SQL queries on the existing HDFS. It is a very good fit for structured data, and if you like to do a bit more than standard SQL, write your own user-defined functions in Java or Python and use them.
Hive scripts are eventually translated to MapReduce jobs, and many times are almost as good as MapReduce jobs in terms of performance.

Please visit http://hive.apache.org/ for more details.

**Pig**
Pig is a high-level procedural language for writing MapReduce programs for data engineers coming from a procedural language background. It is also a good fit for structured data but can handle unstructured text data as well. If you need additional functions that are not available by default, you can write your own user-defined functions in Java or Python and use them.

As Hive, Pig scripts are also translated to MapReduce jobs, and many times are almost as good as MapReduce jobs in terms of performance.

Please visit http://pig.apache.org/ for more details.

**Zookeeper**
Zookeeper is a centralized service to maintain configuration information and names, provide distributed synchronization, and provide group services.

It runs on a cluster of servers and makes cluster coordination fast and scalable. It is also fault tolerant because of its replication design.

Please visit http://zookeeper.apache.org/ for more details.

**Oozie**
Oozie is a workflow scheduler system to manage a variety of Hadoop jobs such as Java MapReduce, MapReduce in other languages, Pig, Hive, and Sqoop. It can also manage system-specific jobs such as Java programs and shell scripts.

As for other components, it is also a scalable, reliable, and extensible system.

Please visit http://oozie.apache.org/ for more details.
Big Data Overview

Flume
Flume is a distributed, reliable, scalable, available service for efficiently collecting, aggregating, and moving large amounts of log data. As shown here, it has a simple and flexible architecture based on streaming data flows:

The incoming data can be log files, click streams, sensor devices, database logs, or social data streams, and outgoing data can be HDFS, S3, NoSQL, or Solr data.

The source will accept incoming data and write data to the channel(s). The channel provides flow order and persists the data, and the sink removes the data from the channel once the data is delivered to the outgoing data store.

Please visit http://flume.apache.org/ for more details.

Sqoop
Sqoop is a tool designed to efficiently transfer bulk data between Hadoop and relational databases. It will do the following:

- Import or export data from relational databases such as Oracle, SQL Server, Teradata, and MySQL to Hadoop
- Import or export data from relational database to HDFS, Hive, or HBase
- Parallelize data transfer for fast load

Please visit http://sqoop.apache.org/ for more details.
Hadoop distributions
There are many distributions in the market, generally classified into two segments—either on premise or in the cloud. It is quite important to know which route we should go for, preferably during the early phase of the project. It is not uncommon to choose both for different use cases or to go for a hybrid approach.

Here, we will look at the pros and cons of each approach:

- **Elasticity**: In the cloud, you can add or delete computing and temporary data resources depending on your usage. So, you pay for only what you use, which is a major pro of the cloud. On premise, you need to procure resources for your peak usage and plan in advance for data growth.

- **Horizontal scalability**: Whether on premise or in cloud, you should be able to add more servers whenever you like. However, it is much quicker to do so on the cloud.

- **Cost**: This is debatable, but going by plain calculation, you may save cost by opting for the cloud because you will make savings on staff, technical support, and data centers.

- **Performance**: This is again debatable, but if most of your data resides on premise, you will get better performance by going on premise due to data locality.

- **Ease of use**: Hadoop server administration on premise is hard, especially due to the lack of skilled people. It’s possibly easier to let our data servers be administered in cloud by administrators who are experts at it.

- **Security**: Although there is no good reason to believe that on premise is more secure than the cloud, because the data is moved out of local premises to the cloud, it is difficult to ascertain the security of sensitive data such as customers’ personal details.

- **Customizability**: There are certain limitations on configurations and the ability to code on the cloud as the software is generic across all of their clients, so it is much better to keep the installation on premise if there are complex requirements that need a high degree of customizations.
Forrester Research published its report in Q1 2014 on leading Hadoop distribution providers, as shown in the following figure. As of this writing, we are still waiting for the new report for 2015 to be released, but we don't expect to see any major changes.

The following figure shows all the leading Hadoop distributions in both segments—on premise and in cloud:

Source: Forrester Report on Big Data solutions Q1 2014
Distribution – on premise
A standard Hadoop distribution (open source Apache) includes:

- The Hadoop Distributed Filesystem (HDFS)
- The Hadoop MapReduce framework
- Hadoop common—shared libraries and utilities, including YARN

There are numerous Hadoop open source Apache components such as Hive, Pig, HBase, and Zookeeper each performing different functions.

If it’s a free, open source collection of software, then why isn’t everyone simply downloading, installing, and using it instead of different packaged distributions?

As Hadoop is an open source project, most of the vendors simply improve the existing code by adding new functionalities and combine it into their own distribution.

The majority of organizations choose to go with one of the vendor distributions, rather than the Apache open source distribution, because:

- They provide technical support and consulting, which makes it easier to use the platforms for mission-critical and enterprise-grade tasks and scale out for more business cases
- They supplement Apache tools with their own tools to address specific tasks
- They package all version-compatible Hadoop components into an easy installable
- They deliver fixes and patches as soon as they discover them, which makes their solutions more stable

In addition, vendors also contribute the updated code to the open source repository, fostering the growth of the overall Hadoop community.
As shown in the following figure, most of the distributions will have core Apache Hadoop components, complemented by their own tools, to provide additional functionalities:

There are at least a dozen vendors, but I will mainly focus on the top three vendors by market share:

- **Cloudera**: It is the oldest distribution with the highest market share. Its components are mostly free and open source, but gets complemented with a few of its own tools if you upgrade to the paid service.

- **MapR**: It has its own proprietary distributed file system (alternate to HDFS). Some of its components are open source, but you get a lot more if upgraded to the paid service. Its performance is slightly better than Cloudera and they have additional features such as mirroring and no single point of failure.

- **HortonWorks**: It is the only distribution that is completely open source. Despite being a relatively new distribution, it enjoys a significant market share.

There are other premium distributions from IBM, Pivotal Software, Oracle, and Microsoft that generally combine Apache projects with their own software/hardware or partner with other vendors to provide all-inclusive data platforms.
Distribution – cloud

It is also called Hadoop as a service. Today, many cloud providers offer Hadoop with a distribution of your choice. The main motivation to run Hadoop in the cloud is that you only pay for what you use.

Although the majority of Hadoop implementations are on premise, Hadoop in the cloud is also a good alternative for organizations who like to:

- Lower the cost of innovation
- Procure large-scale resources quickly
- Have variable resource requirements
- Run closer to the data already in the cloud
- Simplify Hadoop operations

The public cloud market reached $67 billion in 2014 and will grow to $113 billion by 2018, according to Technology Business Research. This only proves that in future more and more data will be generated and stored in the cloud, which will make it easier to process in the cloud.

The top vendors who provide Hadoop as a service, with sometimes a choice of distribution like Cloudera, MapR, or HortonWorks, are:

- Amazon Web Services
- Microsoft’s Azure
- Google Cloud
- Rackspace

You can literally run your own cluster, provided you have a credit or debit card with some balance left. The steps to get started in the cloud are quite simple:

1. Create an account with the cloud service provider.
2. Upload your data into the cloud.
3. Develop your data-processing application in a programming language of your choice.
4. Configure and launch your cluster.
5. Retrieve the output.

We will discuss this in more detail in a subsequent chapter.
Summary

In this chapter, we had an overview of big data and saw the need to process big data using Hadoop. We also had an overview of the Hadoop architecture and its key components and distributions. We also went through how Hadoop fits into the larger big data landscape and related career opportunities.

In the next chapter, we will look at big data use cases within the financial sector.
Where to buy this book
You can buy Hadoop for Finance Essentials from the Packt Publishing website.
Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.
Click here for ordering and shipping details.