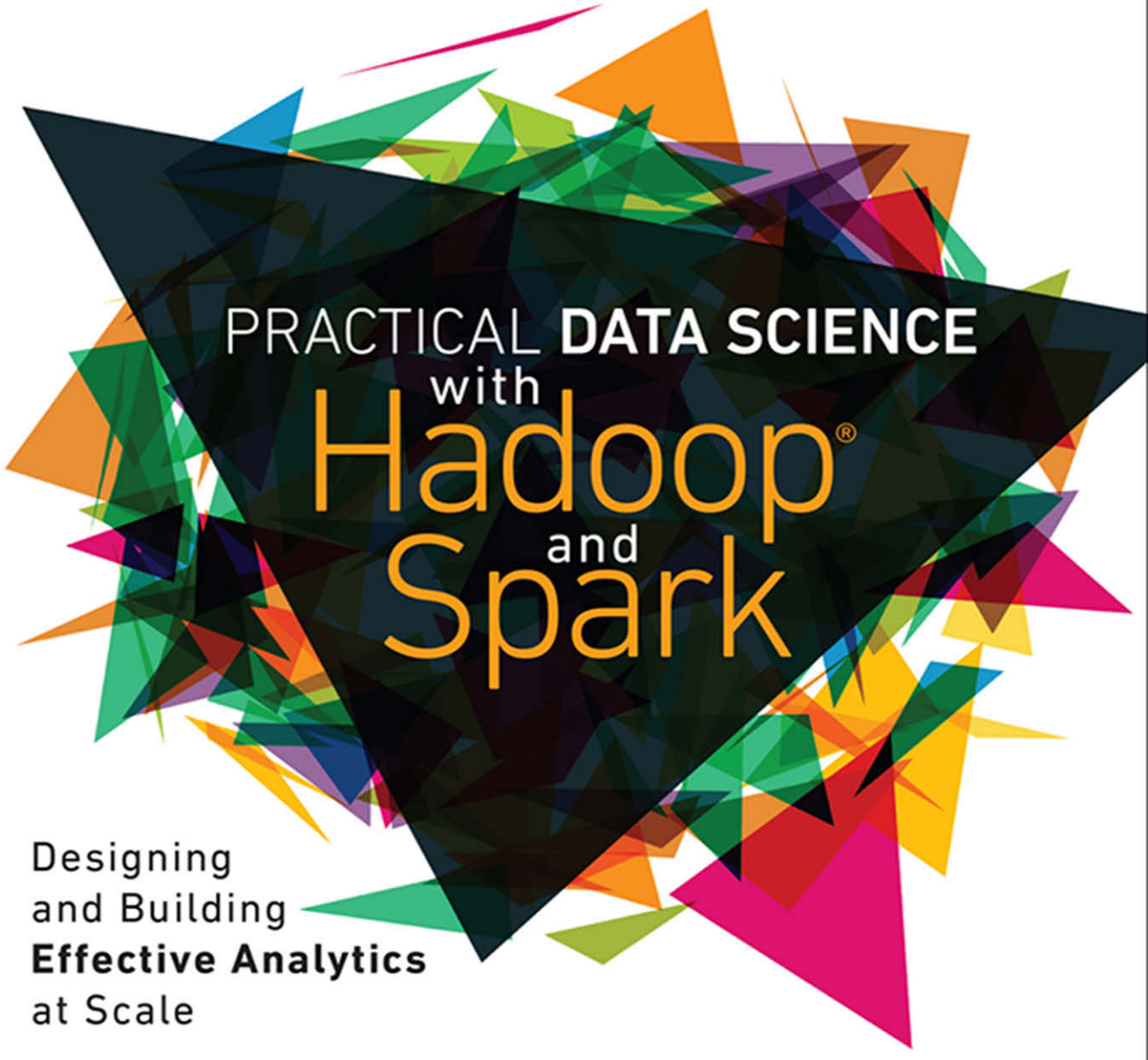
Addison Wesley Data & Analytics Series



PRACTICAL DATA SCIENCE with Hadoop® and Spark

Designing and Building **Effective Analytics** at Scale

OFER MENDELEVITCH
CASEY STELLA
DOUGLAS EADLINE

Practical Data Science with Hadoop® and Spark

This page intentionally left blank

II

Preparing and Visualizing Data with Hadoop

This page intentionally left blank

4

Getting Data into Hadoop

You can have data without information,
but you cannot have information without data.

Daniel Keys Moran

In This Chapter:

- The data lake concept is presented as a new data processing paradigm.
- Basic methods for importing CSV data into HDFS and Hive tables are presented.
- Additional methods for using Spark to import data into Hive tables or directly for a Spark job are presented.
- Apache Sqoop is introduced as a tool for exporting and importing relational data into and out of HDFS.
- Apache Flume is introduced as a tool for transporting and capturing streaming data (e.g., web logs) into HDFS.
- Apache Oozie is introduced as workflow manager for Hadoop ingestion jobs.
- The Apache Falcon project is described as a framework for data governance (organization) on Hadoop clusters.

No matter what kind of data needs processing, there is often a tool for importing such data from or exporting such data into the Hadoop Distributed File System (HDFS). Once stored in HDFS the data may be processed by any number of tools available in the Hadoop ecosystem.

This chapter begins with the concept of the Hadoop data lake and then follows with a general overview of each of the main tools for data ingestion into Hadoop —Spark, Sqoop, and Flume—along with some specific usage examples. Workflow tools such as Oozie and Falcon are presented as tools that aid in managing the ingestion process.

Hadoop as a Data Lake

Data is ubiquitous, but that does not always mean that it's easy to store and access. In fact, many existing pre-Hadoop data architectures tend to be rather strict and therefore difficult to work with and make changes to. The data lake concept changes all that.

So what is a data lake?

With the more traditional database or data warehouse approach, adding data to the database requires data to be transformed into a *pre-determined* schema before it can be loaded into the database. This step is often called "extract, transform, and load" (ETL) and often consumes a lot of time, effort, and expense before the data can be used for downstream applications. More importantly, decisions about how the data will be used must be made during the ETL step, and later changes are costly. In addition, data are often discarded in the ETL step because they do not fit into the data schema or are deemed un-needed or not valuable for downstream applications.

One of the basic features of Hadoop is a central storage space for all data in the Hadoop Distributed File Systems (HDFS), which make possible inexpensive and redundant storage of large datasets at a much lower cost than traditional systems.

This enables the Hadoop data lake approach, wherein all data are often stored in raw format, and what looks like the ETL step is performed when the data are processed by Hadoop applications. This approach, also known as schema on read, enables programmers and users to enforce a structure to suit their needs when they access data. The traditional data warehouse approach, also known as schema on write, requires more upfront design and assumptions about how the data will eventually be used.

For data science purposes, the capability to keep all the data in raw format is extremely beneficial since often it is not clear up front which data items may be valuable to a given data science goal.

With respect to big data, the data lake offers three advantages over a more traditional approach:

- All data are **available**. There is no need to make any assumptions about future data use.
- All data are **sharable**. Multiple business units or researchers can use all available data[1], some of which may not have been previously available due to data compartmentalization on disparate systems.
- All **access methods** are available. Any processing engine (MapReduce, Tez, Spark) or application (Hive, Spark-SQL, Pig) can be used to examine the data and process it as needed.

1. The capability to use all available data is, of course, governed, as you might expect, by the appropriate security policy with Hadoop tools such as Apache Ranger. The point here is that there is no technical hurdle to data sharing, as is often the case with traditional data architectures.

To be clear, data warehouses are valuable business tools, and Hadoop is designed to complement them, not replace them. Nonetheless, the traditional data warehouse technology was developed before the data lake began to fill with such large quantities of data. The growth of new data from disparate sources including social media, click streams, sensor data, and others is such that we are starting to quickly fill the data lake. Traditional ETL stages may not be able to keep up with the rate at which data are entering the lake. There will be overlap, and each tool will address the need for which it was designed.

The difference between a traditional data warehouse and Hadoop is depicted in Figure 4.1.

Different data sources (A, B, C) can be seen entering either an ETL process or a data lake. The ETL process places the data in a schema as it stores (writes) the data to the relational database. The data lake stores the data in raw form. When a Hadoop application

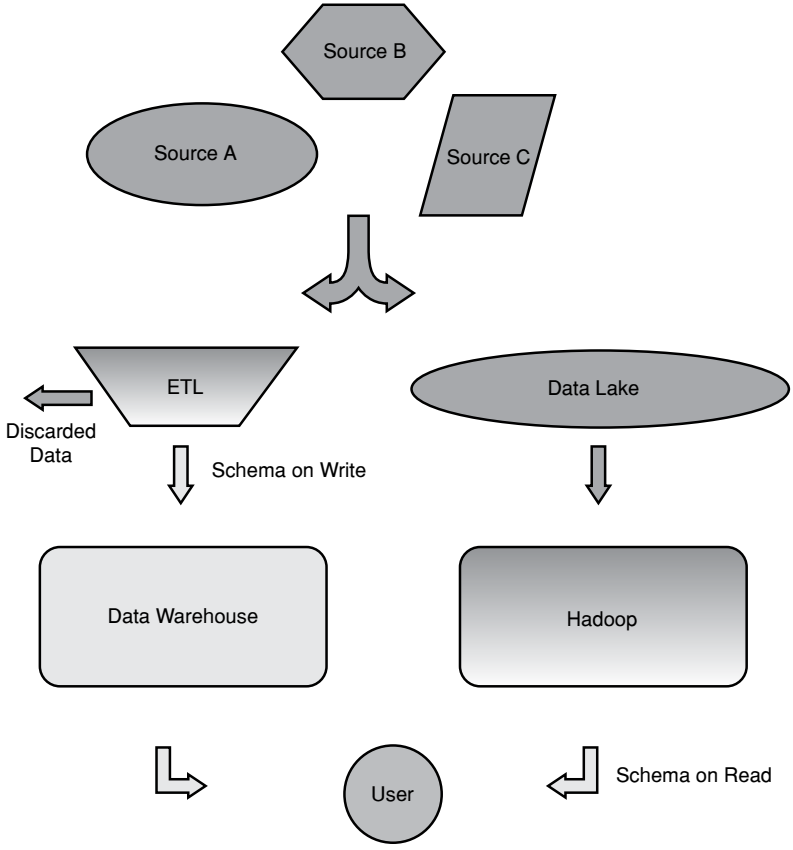


Figure 4.1 The data warehouse versus the Hadoop data lake.