North American Journal of Engineering and Research

Est. 2020

Implementing Data Lakes with Databricks for Advanced Analytics

Ravi Shankar Koppula

Email: Ravikoppula100@gmail.com

Abstract

This paper provides a comprehensive guide for implementing data lakes using Databricks on Azure for advanced analytics. It begins with an introduction to the concept of data lakes, highlighting their capacity to store vast amounts of structured and unstructured data. The paper then discusses the Databricks platform, built on Apache Spark, which simplifies the creation and management of data lakes. Key features of Databricks, such as Delta Lake, are explored for their role in enhancing data storage and processing capabilities. The architectural considerations for choosing appropriate data storage solutions and differentiating between data lakes and data warehouses are examined. The paper also covers data ingestion strategies, both batch and real-time, and delves into the transformation and ETL processes within Databricks. Security and governance issues pertinent to data lakes are addressed, including best practices for maintaining data security. Finally, the paper illustrates the application of advanced analytics with Databricks, emphasizing predictive and prescriptive analytics and the integration of machine learning tools for comprehensive data analysis.

Keywords – Data Lake, Databricks, Azure, Apache Spark, Delta Lake, Big Data, Data Storage, ETL (Extract, Transform, Load), Data Ingestion, Advanced Analytics, Data Security, Data Governance, Machine Learning, Predictive Analytics.

Introduction to Data Lakes

This article will guide users to implement a data lake completely in the cloud using Azure Databricks. One can upload their data to blob storage, bring the data into a Databricks notebook, perform advanced analytics, and also generate a Power BI report using Databricks import function.

A data lake is a centralized repository that allows you to store all your structured and unstructured data at any scale. You can store your data as-is, without having to first structure the data, and run different types of analytics—from dashboards and visualizations to big data processing, real-time analytics, and machine learning to guide better decisions. Data lakes can store many types of data such as relational data from business line applications, object data, database data, files, Commercial Off-The-Shelf (COTS) data, data from on-premises data that has been uploaded, archived data, and operational or archived log files.[1]

Understanding the Concept of Data Lakes

A data lake is a scalable and centralized data storage repository that holds big data until it can be processed and analyzed using various big data technologies. It is capable of storing both unstructured and structured data in its original representations. It is exclusively about big data, serving as an advanced data repository where users can store every type of data in its native format without being forced to first structure the data and run it through the formal process of ingestion. The key characteristic of a data lake is that it has virtually no limits on data storage and can handle data at the petabyte scale with extremely high read and write performance.

The data lake can contain a lot of data, both raw and processed - operational log files, or data sets from Internet of Things (IoT) devices. Enterprises have multiple loads of data stored in different platforms. These platforms cannot store, process, and help analyze the data when needed by modern analytics. Enterprises need a robust architecture with scalable platforms for storing, managing, and retrieving the data when needed. Big data platforms handle this requirement well with data lake architecture. Organizations are moving their data from traditional relational data platforms to big data platforms by implementing data lakes using data storage tools. They extract, transform, keep, and analyze data using big data platforms. Databricks is a unified analytics platform for data engineering, data science, and machine learning, and integrates tightly with Databricks.

Overview of Databricks

Today's data architectures focus on gathering and processing data to create insights into business operations, customer behavior, and future trends. Built around open-source software, cloud storage, and cloud computing, these architectures create a rich ecosystem of big data tools critical to addressing an organization's data requirements. Databricks, which is built on Apache Spark, simplifies the creation and maintenance of big data projects and data lakes, enabling data science and data engineering teams to help their organizations gain insights into business trends more effectively.

With the increased focus on big data, its value has become extremely important to organizations. Many organizations already have data lakes in place. These data lakes can house structured, semi-structured, and unstructured data assets, and when properly utilized, they can be a gold mine of business insights. These data insights enable organizations to understand how their business operations are performing, the characteristics and behavior of their customers, and identify future trends. When the data stored in these data lakes can help businesses gain many important insights, organizations can use this information to drive sales and marketing campaigns, optimize business operations, and deliver value to their customers.[2]

Key Features and Capabilities

The Delta Lake is a fully compatible storage layer with the Parquet format, which is built on top of Apache Spark. In contrast with the original Parquet format, which is immutable and follows the write-once-read-many wall, the Delta Lake introduces a transaction log to enhance the ACID properties of Apache Spark. The transaction log for a Delta Lake is a write-ahead log for the Parquet files. Every time a user wants to commit some Parquet files into the Delta Lake, they will update the transaction log in the Delta Lake to record these transactions. The advantage of the introduction of a transaction log to the Parquet format is to blend the Apache Spark for both streaming and batch processing.

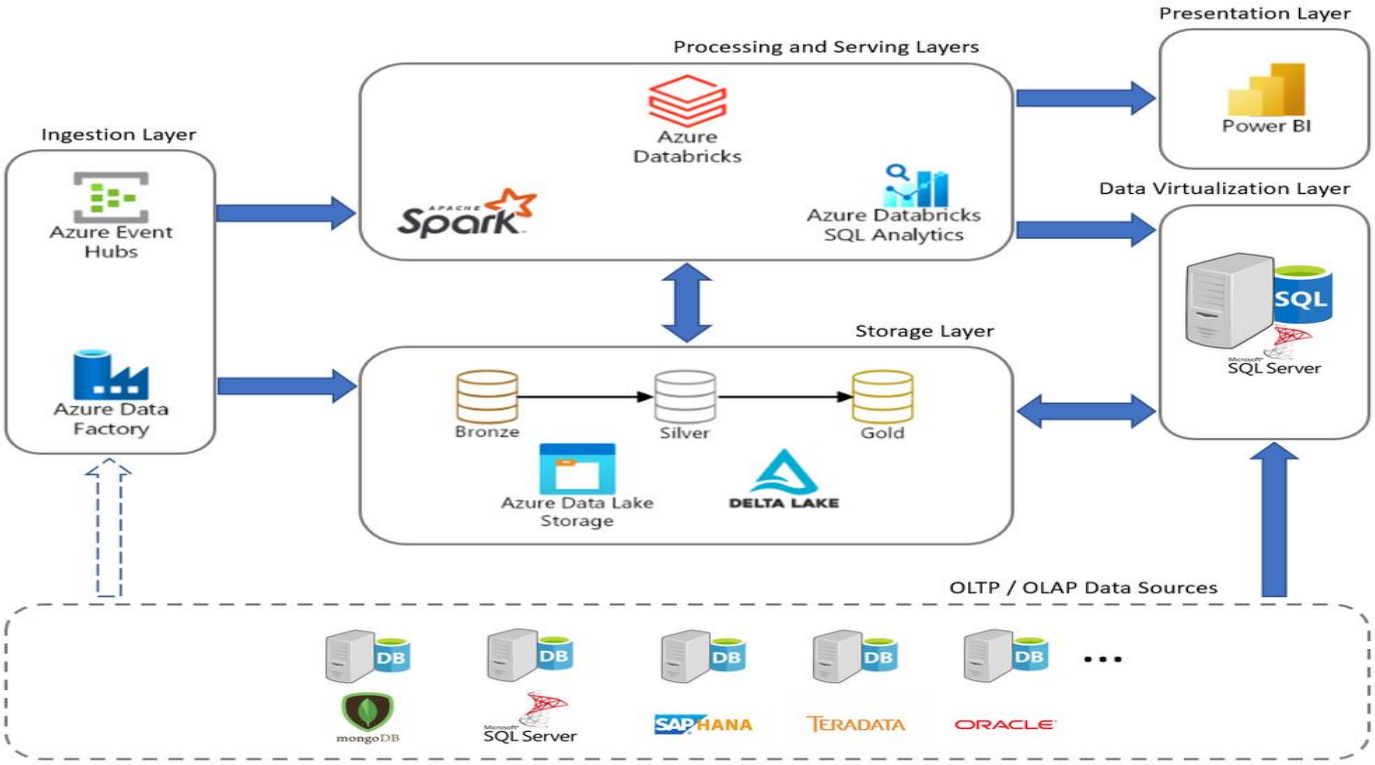

Fig. 1 [3]

The Apache Spark developers that are streaming data into the Delta Lake can work in mini-batches or micro-batches as offline bulk data loading, which is more scalable than the traditional method to commit the transactions in traditional SQL OLTP scenarios. The elaborate design of the Delta Lake has made the Delta Lake an essential tool to develop the data lake solutions and the Databricks data lake solutions. As mentioned in the previous section, the Databricks data lake solutions also include the Databricks Secrets, DBUtils, Data Lake Access Control, and the Databricks optimized Delta Lake caching mechanism.

Architectural Considerations

When to use the right data store: Storage is relatively cheap in terms of cost, but heavily affects analytic performance and usage cost. Deciding where and how to store data can significantly affect systems design, performance, and cost. The recent trend has been to follow the Data Lake architectural pattern: a single storage system used for serving storage needs, enabling big data analytics and data science, and providing data shared services to other clients across an enterprise. Even so, not all data is of similar value and requires common big data analytic engines and processing patterns.

Data Lake versus Data Warehouse: The Data Lake and Data Warehouse architectures serve different business needs and analytic use cases and are complimentary, not a replacement for each other. Both architectures have their place in Big Data and BI systems. As a result, many modern enterprises have both a data lake and a data warehouse. The data lake is used to store all raw data required for high-value analysis and the data warehouse is used to store the refined, clean, transformed, and aggregated data to answer common questions.

Data Lake versus Hadoop Data Lake: Hadoop and the ideas behind it changed how we think about scale and the ability to deal with the three Vs of Big Data (volume, variety, and velocity). The Hadoop File System, HDFS, enabled many new applications to efficiently store and process all kinds of unstructured, semi-structured, and structured data. Many people refer to any storage system with the Data Lake label as a Data Lake whether it runs on Hadoop or not. Spark is a general-purpose distributed data-processing engine that is suitable for use with many storage backends and for many use cases. HDFS became one supported backend storage plug-in for Spark. Spark users filesystem and object store access as it does HDFS access, by using the Hadoop FileSystem (the org.apache.hadoop.fs.FileSystem class). Databricks has thereby been able to provide a hybrid cloud solution for Spark.

Databricks Data Lake versus Traditional Data Lake: Databricks allows customers to treat a cloud storage system like a Hadoop Data Lake, and then Spark on Databricks connects to it just like Spark connects to HDFS. However, the cloud storage system offers cheaper per byte storage prices, fine granularity, and related services such as archival services and lower access costs. Besides, enabling a Hadoop Data Lake architecture on the cloud essentially requires two sets of hardware (the Data Lake file storage and the cloud storage solution), reliability (the needed setup of a NameNode, secondary NameNode, and DataNode is provided by the cloud storage solution), and performance properties (varied data locality settings are provided for cloud storage system write and for object store read operations). Spark on Databricks is merely leveraging the economical and improved performance to support petabyte scale data lakes. Note that when you use S3 and ADLS, Databricks offerings like Delta Lake allow users to do more with the Data Lake.[4]

Designing a Scalable Data Lake Architecture

This collaborative nature of the data lake compounds the need for a scalable design. Best practices recommend using cloud object stores for the data lake file system since they automatically take care of data backups and scaling up, but designing a top-level directory structure that can be navigated by every potential user is a black art. Fortunately, Databricks Autoloader allows us to aggregate data in the way that business users report and visualize the data, in a way that is only restricted by the natural hierarchy revealed by ETL.

By exposing this hierarchy to Databricks through placing at the same level of aggregation files, users experience the same benefits of scale as the ETL. Therefore, business users accessing the data lake directly do not have to import the whole sales data to look for evidence of that old problem with a particular point-of-sale. Instead, the Databricks backend engine ensures that only the file containing that day's sales is read. A Databricks Full Load ETL avoids the overhead of having to know which are the files that contain the sales to be loaded. However, large data files need to be managed, as they will be read and written every time the ETL process is run.

Data Ingestion Strategies

Data Lake Raw Data Ingestion: First, we define an external table to point to the raw data location and define the schema. This can be either a catalog-managed table, where we first define the metadata in the catalog and then rely on Spark to read the external source data and infer the schema, or a data source-managed table, where we define the exact schema and read the text file in the exact format the data is stored. This may result in performance improvements since external data sources are not used for data type extraction and schema inference. All data types that exist in Spark can be used to define the metadata, alongside with the format specification for data, such as CSV, text, Avro, and Parquet. Views are invited in Spark to abstract the consumption of data in which we do not own or maintain the schema, such as externally managed databases.

ETL Process with Databricks: Delta Once the data is considered in-sync, we partition it by date and write it back as Delta, which ushers the end of our data ingestion strategy process. If our primary goal is to make the raw data available for consumption, we can omit the last step and just persist it in an external table or create a view on top of the Parquet data.

However, in our scenario this makes sense, since we provide the data in Parquet format. The true power of data lakes comes from the ability to leverage the data in real-time for advanced analytics after the initial ETL processes and be augmented with curated and transformed data. The curated data is stored in Delta and can be leveraged to create views on top of the data. Users can then query these views directly with SQL in their notebooks or dashboards.[5]

Batch and Real-time Data Ingestion

The first step in creating a cohesive data lake is to create a single channel to obtain the data, whether that data is near-real-time streaming data or batch data that can be scheduled to be imported every hour, day, or week. Stream and batch

data can be brought into a common data lake across multiple protocols using sources like Databricks Connect, Databricks REST APIs, Storage Utilization, or the Databricks File System (DBFS) where the /dbfs/ is prefixed to every data source. Real-time data enables the refreshing of dashboards and reports as new analytics and machine learning models are triggered by the data being processed with Databricks notebooks.

The following four Databricks architecture diagrams focus on batch, real-time and hybrid ETL processing from both on-premise and cloud SQL databases using Databricks databases and tables or Delta Lake tables as a persistent data store for the transformed data. Batch data is traditionally processed using databases and ETL tools, whereas real-time data processing introduces storage, compute, and stream ETL (ST-ETL) services such as Change Data Capture (CDC) which delivers real-time change data to enable the efficient passing of only the data that has changed. Batch processing is likely to still be required to backfill any data that might be missing, but with real-time data processing, gone is the amenity to wait for the nightly batch jobs, implemented in full-stack development sprints, to complete.

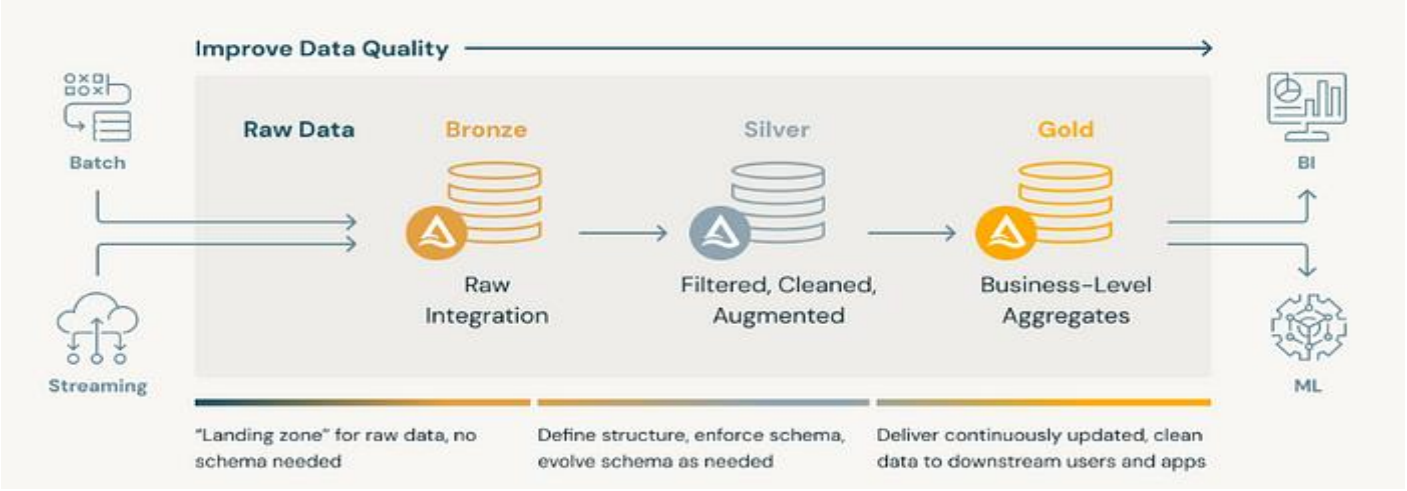

Fig. 2 [6]

Data Transformation and ETL Processes

Data transformation can be complex and costly. Often, the majority of data engineering environments are handed over to master ETL engineers, making it difficult to build anything ingenious because capital becomes a bottleneck for most businesses. Traditional data processing is based on an entire infrastructure in which data is first collected, stored in file systems and databases, and then ingested using ETL tools. Extract, Transform, and Load (ETL) has been a staple for moving and consolidating data sources. ETL and its compliant tools took over traditional message-oriented middleware, real-time data replication, distributed caching, and streaming and event processing tools because they handle big data workloads from a variety of sources and work well. The process, however, is complicated. ETLs perform complicated extractions, loads, and transformations, and as a result, have evolved to ELT (Extract, Load, and Transform).

Agile Data Prep tools disseminate data preparation and transformation pipelines to data engineers, data analysts, and citizen data scientists. Agile data preparation tools, on the other hand, still rely on traditional infrastructures such as layered databases and file systems as well as heavily partitioned data. Databricks, on the other hand, can read data directly from its original sources without configuration. It masks the system's transient properties and the file locations. Databricks, on the other hand, offers the ability to transform

data into required formats, such as Parquet and ORC, an extensive SQL language for transforming, aggregating, and filtering data, and a built-in function library for offline processing, real-time processing on Apache Spark streaming, and machine learning built on MLlib. The goal isn't to write Elasticsearch, Kafka, etc. in Spark and re-implement all of their standards, but to integrate Databricks with Azure and Azure Data Lake. With Databricks, you can access data stored in locations that are not exposed to Azure Data Lake, such as data built on a Linux host, and easily map and access that data as a Hive table. When required, Databricks reads the data directly from insecure locations.

Leveraging Databricks for Data Transformation

Data engineers know that the less we move data, the better. There is no common consensus on who coined the expression "Move the code, not the data," but it has been repeated many times with many variations, such as "Move the intelligence to the data" or "This is the mantra of modern computing: transport a little code to the data, one way or the other, run the code, get the results back." It means that if we can compute faster and cheaper something close to where the data sits, then we are better off. This is a founding principle of data lakes and data warehouses. The lake stores data, and we transport the code to the lake. When designing a data lake solution, we need to identify the best technologies that can be utilized to meet the performance and cost requirements.

Databricks is a cloud-native service that provides a no-limits workspace powered by Apache Spark with the Databricks Delta platform with support for data lake architectures. Databricks has been designed to combine the speed and flexibility of open-source Apache Spark with Databricks used workloads. It is a unified analytics platform for data engineering, machine learning, and analytics.

Data engineers that are building data lake solutions, platforms that can be leased by others, and data engineering specialized workloads can leverage Databricks to produce the E, for Extract-Transform-Load (ETL) workflows. Databricks is a great place to perform this part of the data preprocessing. Databricks combines active optimizations (coded data handling, caching, shuffle handling, etc.), whereas 70%-80% of the data processing typically involves data manipulations that are used in machine learning algorithms. Concentrating a significant portion of the ETL and training/evaluation of models on the same platform allows developers to include domain knowledge into the same deployment of trained models, instead of using intermediate formats that are less efficient in processing at scale. Organizations embedding models into transaction systems can reduce the computational redundancy. [7]

Data Lake Security and Governance

Given the increasing number of data lakes and the amount of data residing in them, data lake security and governance are becoming the most important aspects to focus on while implementing a data lake. Data governance is challenging enough, but adding security can often double the work. With the explosion in distributed systems for data such as Hadoop, Flink, Pandas, Apache Beam, and Spark, the complexity of security and data governance only becomes further daunting.

Effective security and data governance are essential for a data lake because apart from bringing together the data, the key value proposition is to offer a single platform for all your data processing and analytics by running various workloads like ETL, batch, interactive, and real-time stream processing. And, there is no value in it if you cannot apply the appropriate levels of security and governance. With the ability to run applications in mixed off/on in one platform, it is particularly important to ensure that one user's job doesn't inadvertently interfere with another. Keeping data secure and ensuring an efficient way to manage access and control for big data applications is critical.

In the context of big data, the more your use cases are tied to a data lake and the more with deep learning, ETL, batch, real-time use cases, the more your organization is going to have to consider proper handling of data lake security and governance aspects, or else you would be creating a data swamp and active data culture, which would again lead to data silos. Without proper data security policies in place, this data sharing can lead to compliance issues, data leaks, and large-scale data breaches just as the rise in data demands and digital transformation is taking place.

Best Practices for Securing Data Lakes

A common pattern for securing a data lake is by creating a zoned data lake architecture. In this architecture, the left zone has the most restrictive and sensitive data and users, while the right zone has restricted and more open data and users. The left zone can be secured with controls to prevent data exfiltration, while the right zone is architected with open APIs and controls. To use this architecture, data is structured by quality levels with security controls for each zone (left, core, right). Security controls in the core zone are oriented towards knowing who is doing what with the data, and how they are using it. In addition, you can strengthen the security of your data lake by performing DataOps, which involves using automation to bring transparency and application of controls throughout the whole data lifecycle to the data you have in your data lake.

Zoned Data Lake Security Model: You must evaluate data security in terms of operational security and performance tradeoffs. - Sensitive data is socialized with the most restrictive security controls, while open data services are socialized with service level controls. - It is recommended to maintain largely inert sensitive data in production environments that need high data freshness and durability. Workload isolation is an essential security best practice for data lakes where multiple modeling and analysis use cases execute queries in which sensitive data is a component. It is important to isolate workloads to specific VNet(s) and region(s). Protect your workload on Data Lake Storage Gen2 during the data lifecycle as it flows into and out of the data lake with Azure AD authentication.[8]

Advanced Analytics with Databricks

From diagnostic and diagnostic analytics, we move up the value chain to predictive and prescriptive analytics: these are the most advanced forms of analytics. At this level, we are making predictions about future events or making decisions impacting future events. With predictive analytics, we are using data to predict future events. With prescriptive analytics, the focus is on recommending the course of action based on the predictions that were made with predictive analytics. Fully exploiting predictive and prescriptive analytics means being able to adopt powerful and complex analytic methods to sophisticated and intensive data processing. These kinds of analytic methods are carried out in specialized tools and require specialized skill sets to manage and implement these analytics. Databricks handles big data processing of raw data, allows advanced analytics to be applied, and can manage the access and transfer of data.[9]

Databricks has a number of built-in libraries like SparkML, scikit-learn, Horovod, TensorFlow, Keras, etc. that provide support for advanced analytics like machine learning and deep learning on big data. SparkML is the fully supported machine learning library available as part of Databricks Runtime that allows Spark to become a very capable massively parallelized platform that can handle large-scale machine learning workloads because of its ability to leverage past data through patterns found from selected algorithms. Databricks has a very nice integration with XGBoost and its very powerful MLlib library to take full advantage of the very fast and flexible algorithm it provides. Databricks integrates very nicely with Scikit-learn, and that package allows the creation of many machine learning models with the use of spark-sklearn-interface to run the machine learning algorithm more efficiently. Furthermore, the deep learning community widely uses TensorFlow, and Databricks makes it possible to effectively execute deep learning computation with Databricks Runtime configured with the above-mentioned TensorFlow. With the Keras framework using high-level neural networks API, it can handle complex problems building powerful deep learning Keras models with large-scale data engines.

Machine Learning and AI Capabilities

Databricks Unified Analytics Platform offers an extensive suite of machine learning frameworks, such as Spark MLlib, TensorFlow, Horovod, Keras, and Scikit-Learn. Databricks has several features aimed at simplifying the end-to-end ML lifecycle, such as Interactive Notebooks, MLflow, and the ability to build scheduled data pipelines which can consume CI/CD code to support the data and model evolution over time.

Interactive Notebooks: Databricks ML demonstrates easy integration with machine learning libraries, specific libraries, and other analytics products such as Azure Machine Learning services, MLflow, and Horovod for distributed training and deep learning specific support. Interactive Notebooks greatly simplify training and model iteration before the final model implementation and support data exploration based on a very powerful platform running on a massive scale of data through a Unified Data Analytics Model.

The Unified Data Analytics Model allows for more democratization across an organization with data access and generalization in the application of algorithms. Democratizing the use of machine learning and having the feature available as a service accelerates model deployments at all levels – from simple models to solve business problems to more ambitious initiatives such as deep learning algorithms running on big data.[10]

Conclusion and Key Takeaways

The goal of this article is to provide recommendations on when and how to use Databricks for specific big data use cases. Databricks was conceptualized by the creators of Apache Spark to make big data and machine learning (ML) simple at scale. Databricks provides a platform that runs on top of Apache Spark, unifying data engineering with data science. With advances in big data technologies, including Delta Lake, it is now possible to implement data lakes for scalable BI on demand. Databricks provides one of the best workbenches to implement data lakes due to its excellent platform capabilities and integration with flagship technologies, including Azure Data Lake Storage and Azure Synapse Analytics. In this book, we have explained how to implement a big data platform with Databricks and Azure.

By doing big data at scale with Databricks, companies can now quickly gain significant competitive advantage in rapidly

changing scenarios. Solving large-scale analytical use cases in a matter of hours compared to days allows organizations to iterate through multiple qualitative life cycles before making informed decisions. It takes time for an organization to develop big data use cases and fully benefit from it. There is no silver bullet to big data, but Databricks is poised to take over as one of the best possible platforms to implement future state enterprise analytics systems. In this book, we are providing enterprise architects who want to implement advanced analytics at scale on a Databricks platform all the information they need to start their journey. With the real-world Databricks use case templates, scripts, and instructions we have provided, every data architect can benefit from the experience we have gained in producing modern data platforms. Start creating petabyte-scale data lakes on Azure Data Lake Storage and query the data with Azure Synapse Analytics at an ELT scale. Make advanced analytics a reality with scalable big data technologies like Databricks and its integration with Azure PaaS.

References

[1] C. L. Philip Chen and C. Zhang, 'Data-intensive applications, challenges, techniques and technologies: A survey on Big Data', 2014

[2] A. Labrinidis and H. V. Jagadish, "Challenges and opportunities with big data," *Proceedings of the VLDB Endowment*, vol. 5, no. 12, pp. 2032-2033, 2012.

[3] Data lake architecture," Databricks, [Online]. Available: https://databricks.com/solutions/data-lake

[4] J. Manyika et al., "Big data: The next frontier for innovation, competition, and productivity," *McKinsey Global Institute*, 2011.

[5] D. Agrawal, S. Das, and A. El Abbadi, "Big data and cloud computing: New wine or just new bottles?," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 1647-1648, 2011.

[6] "Databricks Unified Data Analytics Platform," Databricks, [Online]. Available: https://databricks.com/product/unified-data-analytics-platform.

[7] H. G. C. Gomes et al., "Databricks Delta: Simplifying Data Engineering and Management with a Unified Platform," *Proceedings of the 2019 International Conference on Management of Data (SIGMOD '19)*, pp. 2123-2136, 2019.

[8] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107-113, 2008.

[9] B. Spivey, "Hadoop Security: Protecting Your Big Data Platform," 2015.

[10] F. Provost and T. Fawcett, "Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking," 2013.