Assessment of Databricks as a Data and Analytics Platform

Published 11 March 2022 - ID G00763629 - 54 min read

By Analyst(s): Prasad Pore

Initiatives: Data Management Solutions for Technical Professionals; Analytics and Artificial Intelligence for Technical Professionals; Evolve Technology and Process Capabilities to Support D&A

Databricks, a cloud-based data and analytics platform, has evolved from a managed Apache Spark platform to an optimized data lakehouse platform. This research helps data and analytics technical professionals evaluate capabilities of it and its suitability for adoption as an enterprise solution.

Overview

Key Findings

- Databricks is a commercial, integrated, end-to-end cloud-based data and analytics platform, delivered as a platform as a service (PaaS). It includes proprietary products and features (some are built on Apache Spark, MLflow and Delta Lake), and additional open-source software (e.g., Delta Sharing, Redash, TensorFlow, scikit-learn, etc.). It also integrates seamlessly with multiple products from the ecosystem of technology partners.
- Main value propositions of Databricks platform include Delta Lake, Delta Engine and Managed MLflow, which make it a strong competitor in the Cloud Data Management Platform and Data Science Machine Learning Platform marketplace.
- Databricks main components are modular, autonomous and orchestrated. Additionally, it has capabilities to catalog and manage data and analytics artifacts (i.e., databases, ML features, ML models). It makes Databricks first-class option available to design and implement composable data and analytics.
- Databricks has added several new products and features to its data and analytics platform in the past few years — some of which are not yet officially released (GA), but are in Private/Public Preview mode. It has also added new technology partners to its partner ecosystem, and contributed new projects to the open-source community, such as Delta Lake and Delta Sharing.

Recommendations

Technical professionals responsible for data management and analytics solutions should:

- Evaluate capabilities of Databricks proprietary features/products (e.g., Delta Engine, Managed MLflow, Unity Catalog, Delta Live Tables), other open-source products bundled with it (e.g., Delta Lake, Delta Sharing, TensorFlow, Redash, etc.) and required third-party products (e.g., Tableau, Power BI, Looker, etc.), and then align it with business requirements before implementing an end-to-end data and analytics architecture.

- Anticipate new processes required in implementing Delta Lake (lakehouse) and MLflow. Ensure that those processes are well-defined and can be easily integrated into your existing cloud infrastructure to deliver the necessary and required capabilities. It's also important to evaluate their applicability to the target use cases, scale and team size (e.g., customers with basic data science or BI use cases may not need end-to-end MLflow processes).

- Leverage Databricks capabilities, such as decoupled storage and compute, open data format, open data sharing, data and machine learning pipeline orchestration, data catalog, feature store, and centralized management to accelerate the initiatives of composable data and analytics (e.g., customers can use the Delta Lake with Microsoft Azure Machine Learning or Amazon SageMaker to reap benefits of lakehouse architecture). Also, users can use Amazon Redshift, Azure Synapse or Google Cloud BigQuery with Databricks Runtime ML and Managed MLflow to leverage parallel processing and end-to-end ML life cycle management.

- Differentiate released, in-public-preview and in-private-preview product features and timelines carefully, as there are many features and products in Databricks platform with different release types and timelines. Leverage public/private preview features of Databricks to evaluate its future capabilities and limitations to create either adoption strategy for those features, or an alternative strategy in case of its limitations.

Analysis

Databricks is a commercial, cloud-based data and analytics platform delivered as PaaS, built on open-source software (OSS) — Apache Spark, MLflow and Delta Lake. With the capabilities of Delta Lake to implement a lakehouse — an emerging concept of modern data management architecture — it is now marketed as 'The Databricks Lakehouse Platform.' It is a public cloud-based platform (e.g., Amazon Web Services [AWS], Azure Cloud Platform, Google Cloud Platform [GCP]) that supports Virtual Private Cloud (VPC) deployment, but doesn't have an on-premises or private cloud (e.g., IBM, Dell Technologies, SAP, Hewlett Packard Enterprise [HPE]) deployment support. At the core, it unifies multiple optimized OSS, the cloud platform and third-party softwares to offer a simplified managed data and analytics platform as a service. It is a leader in Gartner Magic Quadrant for Data Science and Machine Learning Platforms and Magic Quadrant for Cloud Database Management Systems.

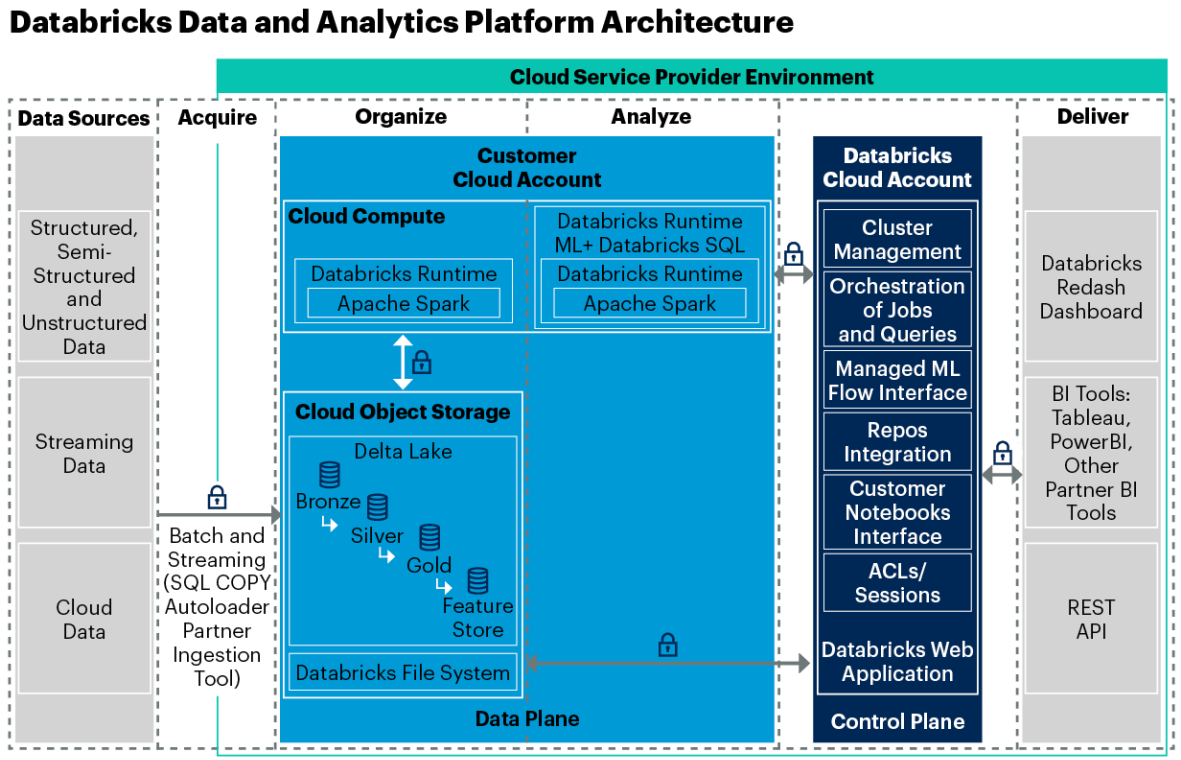
Databricks was founded with the initial goal of providing a fully managed, cloud-based Apache Spark Ecosystem to process and analyze data from data lake or relational database systems (RDBMS). Over the years, Databricks added many new proprietary and optimized features, some of which are built on four open-source projects created by Databricks: MLflow, Delta Lake, Koalas and Delta Sharing. Until the addition of Delta Lake to the platform, it was purely a managed, cloud-based data processing and analysis platform — without any control over data persistence, organization and management. With the addition of Delta Lake to the platform, it acquired the required capabilities of a database management system to implement the concept of a data lakehouse on the cloud storage, making it a cloud-based end-to-end data and analytics platform.

> **The idea of a lakehouse is to converge and consolidate data storage and processing into a single platform that supports different workloads, such as data engineering, data science, artificial intelligence/machine learning (AI/ML) engineering and business intelligence (BI).**
>
> *— Gartner, 2022*

For more information on the lakehouse architecture and how current data warehouse and data lake vendors are approaching to implement it, please refer to Exploring Lakehouse Architecture and Use Cases.

Databricks currently supports three public cloud service providers for hosting — AWS, Microsoft Azure and GCP. Customers can select any of them based on their requirements or existing cloud service provider. Once it is selected, Databricks service can be purchased from its marketplace (see Figure 1).

Figure 1: Databricks Data and Analytics Platform Architecture



As shown in Figure 1, Databricks platform has two planes: control plane and data plane. Control plane runs in a multitenant cloud account of Databricks, which contains Databricks web application, orchestration, cluster management and other tools required to provide managed services. Data plane runs in customers' cloud accounts and includes decoupled compute clusters, created by Databricks, and persistence storage where customer data is stored. While some data (e.g., configurations, logs, user information, etc.) resides in the control plane, it is encrypted at rest — as well as in transit. Communication between control plane and data plane always happens in a secure private network. Also, the customer has the option to configure its preferred location for such data. Databricks creates all the compute clusters in a single Virtual Private Cloud (VPC) by default. However, the customer can create their own customer-managed VPC to comply with specific cloud security and governance standards. All clusters created for the jobs are transient in nature and often terminated once the job is completed. Databricks offers its own container services, allowing customers to use docker images for full control over system libraries or standard environment use cases. Users can also create customized deep learning environments on GPU clusters. On GCP, Databricks uses Google Kubernetes Engine (GKE) for container orchestration. Though currently Databricks doesn't support private cloud vendors, containerization support is foundational for a platform to be portable and flexible. With decoupled architecture, support for containerization, use of open standards and integration with multiple third-party tools, Databricks can become an easily portable platform in the future, giving customers flexibility to migrate from one cloud vendor (public or private) to another.

Databricks platform offers three runtimes:

1. **Databricks Runtime:** It includes an optimized delta engine built on Apache Spark, its libraries and features to implement Delta Lake and other managed services. Compared to Apache Spark, it improves overall performance, usability and security of the platform. Additionally, it packages Ubuntu OS, GPU libraries and programming language libraries (e.g., Python, R, Scala and Java) to create nodes and clusters. Currently, only Databricks Runtime supports customized container services for cluster management.

2. **Databricks Runtime for ML:** A specialized version of Databricks Runtime for Machine Learning use cases. In addition to all features from Databricks Runtime, it also includes popular ML libraries (such as TensorFlow, Keras, PyTorch, MLflow, Horovod, GraphFrames, scikit-learn, XGboost, NumPy, MLeap and pandas) and fully managed ML life cycle management tool built on OSS MLflow.

3. **Databricks Light:** The open-source Apache Spark Runtime for use cases where optimized Databricks Runtime features are not required. It can be used for simple and non-critical jobs with a little lower pricing rate compared to Databricks Runtime.

Assessment

Databricks platform consists of multiple processes, proprietary products, open-source softwares and proprietary features built on it that make overall end-to-end assessment broad and generic. Hence, Gartner has developed a set of assessment criteria, which is divided into technical and nontechnical criteria (see Table 1).

Table 1: Gartner Assessment Criteria

1. Technical Criteria	2. Non-technical Criteria
1.1 Completeness of Solution	2.1 Skills Requirements/Development
1.2 Governance, Security and Compliance	2.2 Quality of Documentation
1.3 Deployment Support on Cloud	2.3 Future Technology Roadmap
1.4 Integration With Third-Party Products	2.4 Customer Support
1.5 Scalability, Performance, Disaster Recovery/High Availability	2.5 Licensing Cost
1.6 Ease and Speed of Development	
1.7 Ease of Operations and Management	
1.8 Commitment to Open Source and Standards	

Source: Gartner (March 2022)

1. Technical Criteria

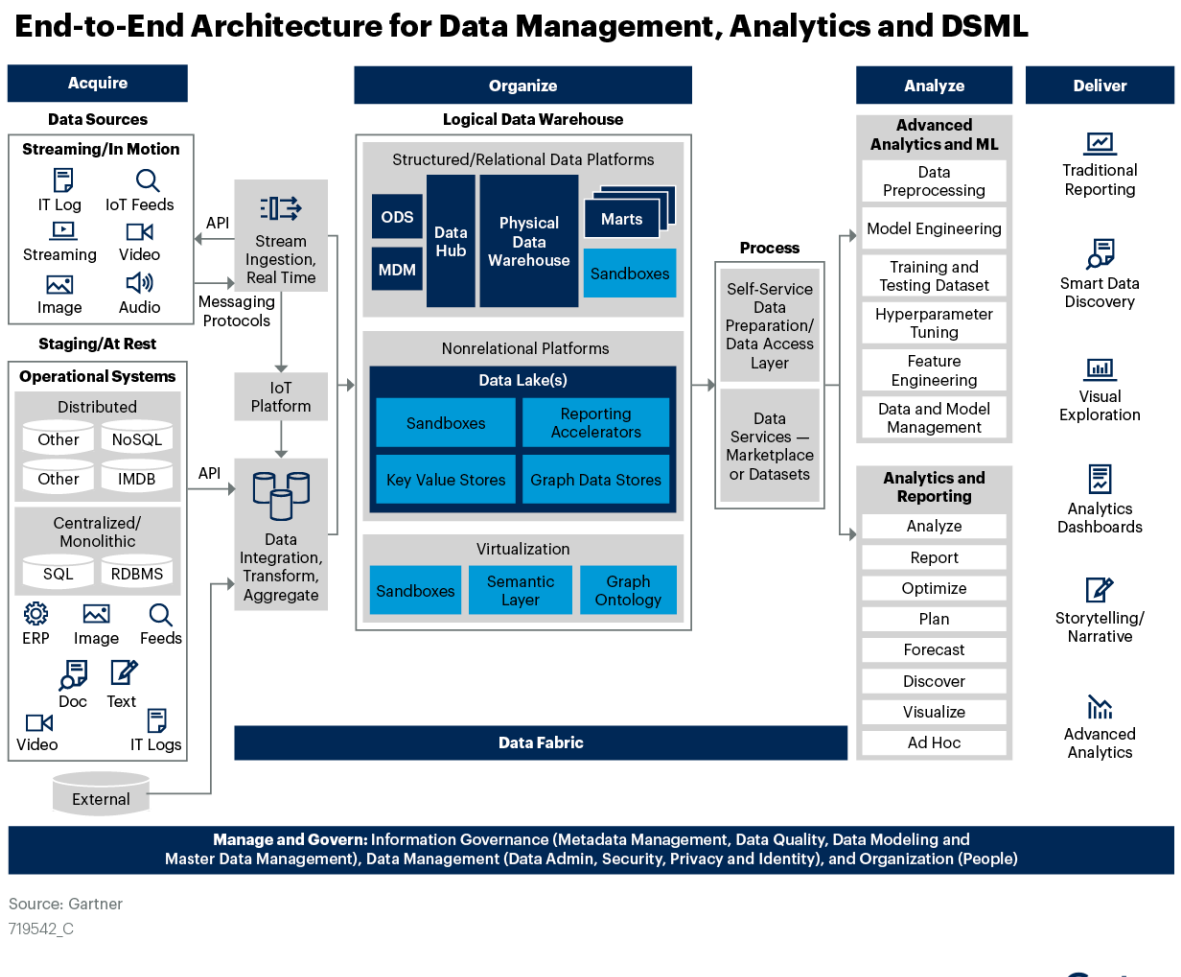
1.1 Completeness of Solution

For a data and analytics platform to be considered as an end-to-end data and analytics platform, it must have a set of products and/or features that supports different data sources and helps implement typical processes of a data and analytics architecture. Gartner recommends its client to use below reference architecture to design end-to-end data and analytics architecture. For more information about Gartner reference architecture, please refer to Solution Path for Building a Holistic Data Management and Analytics Architecture.

Below are the four main processes of Gartner data and analytics reference architecture (see Figure 2):

- **Acquire**
- **Organize**
- **Analyze**
- **Deliver**

Figure 2: End-to-End Architecture for Data Management, Analytics and DSML



Source: Gartner
719542_C

Gartner

Acquire

With the advent of big data and emerging analytics use cases, it is necessary for a data and analytics platform to support all types, states and locations of the data. Databricks supports structured, semistructured and unstructured types of data — and in motion (streaming) and at rest, states of the data. Databricks itself supports only host cloud data sources but, with the help of technology partners, it integrates on-premises and intercloud data sources. Please find the list of supported types of data sources in Table 2 below:

Table 2: Databricks Support for Data Sources and Types

Data Source	Supported Tables
Files	Apache Avro, Binary, CSV, Apache Hive Table, Image JSON, LZO compressed, MLflow experiment, Apache Parquet, XML, Zip
Data Storage Systems	AWS Redshift, Amazon Simple Storage Service (Amazon S3), Amazon S3 Select, Azure Blob Storage, Azure Data Lake Storage Gen1, Azure Data Lake Storage Gen2, Azure Cosmos DB, Azure Synapse Analytics, Apache Cassandra, Couchbase, Elasticsearch, Google BigQuery, Google Cloud Storage, MongoDB, Neo4j, Oracle, Redis, Riak TS, Snowflake, SQL databases using JDBC

Source: Gartner (March 2022)

Common use cases in the data acquisition process are batch and streaming (data in motion) data ingestion. Databricks leverages Apache Spark Structured Streaming API, which unifies batch and streaming data processing — removing the complexity, cost and efforts of lambda architecture. It has default microbatching mode (latency ~100 ms) and experimental continuous processing mode (latency ~1 ms). As structured streaming APIs are based on microbatch processing approach by default, it's a near real-time streaming engine, and not a true real-time engine, such as Apache Flink. Additionally, Databricks doesn't have streaming event ingestion capabilities as that of Apache Kafka, Amazon Kinesis, Azure Event Hub, etc. However, users can add these capabilities by integrating the event ingestion product/service of their choice with the Databricks platform.

Table 3 gives details of how Databricks supports data acquisition from different data source locations:

Table 3: Databricks Support for Data Acquisition

(Enlarged table in Appendix)

Data Acquisition From Host Cloud Storage	Data Acquisition From On-premises, Intercloud and Third-party Sources
■ Spark structured streaming API ingest data from host cloud storage using Auto Loader and SQL COPY command. ■ Auto Loader feature provides an optimized cloud file source, which incrementally and efficiently discovers a large number of new files landed on the host cloud storages (e.g., Amazon S3, ADLS Gen2, ADLS Gen1, Azure Blob Storage, Google BigQuery, Google Cloud Storage, HDFS). It provides the Trigger Once mode, which makes microbatching run periodically, rendering it a batch mode. Additionally, it also provides features, such as schema inference and support for handling schema evolution. ■ Users who prefer SQL can use SQL COPY INTO command to ingest the batch data with exactly once semantics. ■ SQL CREATE TABLE [USING] feature allows creation of External Tables to access data from external data systems.	■ Databricks doesn't have in-house products to ingest data from on-premises, legacy systems (e.g., Mainframes), other cloud service providers or third-party applications (e.g., Salesforce, Twitter, etc.). ■ To fulfill such an important use case, Databricks has created an ecosystem of technology partners, which are established fit for purpose products like Fivetran, Qlik, Infoworks, StreamSets and Syncsort. ■ Users can also use the products from cloud service providers, such as AWS Glue, Azure Data Factory and Google Cloud Data Fusion.

Source: Gartner (March 2022)

Organize

This process involves persistent data storage and data engineering tasks, such as storing/moving/deleting data and cleansing/transforming/validating the data. Users create data pipelines to implement this complete process flow. To support, manage and operationalize these data engineering activities, users additionally need to implement data operations, also called DataOps.

Data Storage

As shown in Figure 1, Databricks uses Delta Lake format to reliably persist data in the host cloud storage or existing data lake on the host cloud. Delta Lake is an open-source storage abstraction layer built on Parquet format — a columnar storage open format. Under the hood, it stores versioned Parquet files and transaction logs of all commits made to the table in the host cloud storage. It gives Delta Lake capabilities, such as low latency, high data compression, efficient encoding and low storage cost. It also allows external systems like Presto, Amazon Athena, AWS Redshift Spectrum, Apache Hive and Snowflake to read the data from the Delta Lake. Delta Lake is fully compatible with Apache Spark APIs and supports programming in Scala, Java, Python and SQL. In addition, it also supports connectivity to external systems with connectors including Rust API, Python API, DBT, Hive, Presto, etc.

Below are the capabilities of Delta Lake that help enable a data lakehouse architecture:

- ACID (Atomicity, Consistency, Isolation, Durability) Transactions: Delta Lake uses ACID transaction protocol to write the data, which ensures consistency in concurrently reading, writing or updating the data. By adhering to protocol, external systems can write the data to the Delta Lake.
- Scalable Metadata Handling: Delta Lake treats metadata the same as data by leveraging Spark's distributed processing capabilities, which makes Delta Lake capable of handling petabyte scale data.
- Time Travel (data versioning): Users can query the data in previous point in time to reproduce the same state of data (e.g., to rerun the experiment on the same version of data).
- Open Format: Delta Lake is an open format built by leveraging efficient compression and encoding scheme of Parquet format.
- Unified Batch and Streaming Source and Sink: Supports read/write of streaming data and can be used as source and sink for the streaming use cases.
- Schema Enforcement and Evolution: Users can specify and enforce schema to ensure good data quality. It also supports handling of schema changes in a simple way.
- Audit History: Transaction logs record all changes made to data, providing complete audit history.
- Updates and Deletes: Users can easily update and delete data in compliance with GDPR and CCPA.

- Delta Sharing: An open protocol for sharing data from Delta Lake.

With these useful features, there are some limitations in Delta Lake:

- No support for integrity constraints except NOT NULL and CHECK constraints
- No support for multitable transactions
- No support for some of the DDL and DML features, such as:
  - ANALYZE TABLE PARTITION
  - ALTER TABLE [ADD|DROP] PARTITION
  - ALTER TABLE RECOVER PARTITIONS
  - ALTER TABLE SET SERDEPROPERTIES
  - CREATE TABLE LIKE
  - INSERT OVERWRITE DIRECTORY
  - LOAD DATA
  - INSERT INTO [OVERWRITE] table with static partitions
  - INSERT OVERWRITE TABLE for table with dynamic partitions
  - Bucketing
  - Specifying a schema when reading from a table
  - Specifying target partitions using PARTITION (part_spec) in TRUNCATE TABLE
- Changing column type/name or dropping a column requires rewriting of the complete table.

A typical lakehouse storage architecture consists of multiple layers or zones, with each layer optimized for different styles of consumption. These all layers are accessible for consumption to different personas in the team, depending on the access permissions, such as data engineer, data scientist, data analyst and business analyst.

Each layer has its purpose for a particular use case:

- **Bronze Layer:** Data in this layer is the output of the acquisition process. It is a raw format data, stored in the same format as it is ingested from the sources. It could be in structured, semistructured or unstructured format. This data can be utilized for special use cases of data science or machine learning.
- **Silver Layer:** A refined version of Bronze layer data which is now filtered, cleansed, validated, transformed, enriched and even normalized. At this layer, tables can be joined and constraints can be added. As data in this layer is filtered, accurate and well-structured, it acts as Single Source of Truth (SSOT) for the organization and can be utilized for analytical use cases in wider organization, or further downstream applications and/or processes.
- **Gold Layer:** This data is summarized and aggregated version of Silver layer data, ready for dashboard and reporting in business use cases, as well as for consumption in Data Science and machine learning use cases. It is the highest quality data among all the storage layers.

Databricks also creates a distributed file system called Databricks File System (DBFS), an abstraction layer on top of host cloud storage to store data generated by workspace, interactive notebooks, logging and experimentation from different applications in the control plane. Data stored in DBFS is always encrypted and customers have the option (In Public Preview) to use customer managed keys for encryption. DBFS allows users to access cloud storage by mounting it without explicitly requiring credentials and to interact with it using directory and file semantics, instead of storage URLs.

Big data processing frameworks such as Hadoop and Spark implement fault-tolerance by dividing the processing job into small retriable tasks. Practically, when a job executes, some of the tasks may fail occasionally. To ensure that only the results of successfully completed tasks and jobs are available to end users/processes, these frameworks use commit protocol, which defines the rules about how the final results should be written when a job completes. Hadoop offers two versions of the commit protocol (Hadoop Commit V1 and Hadoop Commit V2) and open-source Apache Spark leverages the same protocols to achieve fault-tolerance. V1 protocol uses temporary staging to store individual task output files and move them from staging to final location after the job is complete, ensuring the transactionality (All or Nothing). V2 protocol, on the other hand, stores individual task output files directly to final location without staging, giving better performance than V1. In case of job failure, on the other hand, V2 causes partial or corrupted residual files, which can cause further issues in the data pipeline.

Databricks solved this trade-off by innovating a new transactional commit protocol called Databricks I/O (DBIO). When a job is submitted, DBIO will write individual task output files with a unique transaction identifier embedded in the filename directly to the final location. Once the job is completed, it will then mark the transaction as committed in the metadata service. While listing those files, DBIO will check for transaction ID embedded in the filename as well as corresponding status from the metadata. If the transaction ID is present and the status is committed, it will read the task output files, or otherwise, simply ignore those files. Thus, it ensures performance as well as reliability.

Data Processing and Querying

When data flows through above layers, it is processed by a proprietary, scalable and optimized data processing engine built on Apache Spark, with the same set of APIs, called Delta Engine (included in Databricks Runtime). It is optimized for lakehouse style data and specially built for modern hardware that can perform tasks like single instruction multiple data (SIMD) instruction set. Also, inside the Delta Engine, is a new addition of native vectorized query engine written in C++, called Photon (In Public Preview). As performance of native code is better than code written for JVM or similar technologies, Photon engine adds that extra boost in the performance. To use Photon, users need to select appropriate Databricks Runtime while creating clusters. All of these advanced features differentiates Databricks from open-source Apache Spark.

Delta engine is one of the core value propositions of the Databricks platform, which delivers high performance for large-scale data processing. It achieves that with many optimization techniques, some of which are listed below:

- Advanced file management
  - File compaction: It improves the performance of read queries by coalescing small files into larger files. Users can perform manually or using auto compaction with small performance cost while writing.
  - Data skipping technique collects min-max statistics while writing data, and utilizes it while querying.
  - Z-ordering is a multidimensional clustering technique to colocate related information in the set of files. Users must choose the right number of required columns for Z-ordering to achieve optimal performance gain.
- Caching: Databricks offers two types of cache — Spark cache and Delta cache, which can be used at the same time.

- Dynamic file pruning: Allows skipping unnecessary file scans for literal filters and join filters while executing queries.
- Bloom filter indexing: Uses a space-efficient probabilistic data structure for data skipping on selected columns.

Data Operations, aka DataOps

To help manage data pipelines, Databricks provides a Jobs feature for centralized orchestration (via workspace GUI or CLI). It helps create multiple tasks jobs and data pipelines, schedule the jobs, manage access and monitor activities. Visual representation of data pipelines helps monitor and trace the workflows. Users can also integrate and use third-party tools, such as Airflow, Azure Data Factory and AWS Glue.

Databricks has also launched a new framework called Delta Live Tables (In Public Preview) which helps building reliable, maintainable and testable data processing pipelines. It allows the creation of declarative pipelines using SQL or Python, focusing on business logic and letting Delta Live Tables handle underlying pipeline creation and management of table dependencies. Users can enable continuous, on-demand or trigger based updates of tables. Updates in one table will automatically trigger the updates on dependent downstream tables. Delta Live Tables allows users to set data quality expectations using business rules to validate the data during processing. Based on output of validation rules, it displays data quality results. Users can visually track, monitor and analyze end to end data pipelines, which helps them monitor pipeline health, as well as investigate performance issues using data flow graph dashboard.

As Delta Live Tables only supports SQL and Python for development of data pipelines, it's not suitable for teams who prefer interactive GUI based tools (e.g., AWS Glue, Azure Data Factory) to build data pipelines. Also, it is a very new feature/concept which will take some time to prove its applicability to real-world complex use cases.

For automated Continuous Integration/Continuous Delivery, Git repositories are widely used for storage, history and version control of source code, along with build automation tools such as Jenkins. Databricks doesn't have its own CI/CD tools or services, but integrates well with popular Git repositories and build automation tools like GitHub, Bitbucket, Jenkins, AWS suite of CI/CD Services, Azure DevOps Service. Users can copy the git code repository from the git server to Databricks workspace, edit and commit notebooks/source code files, and collaborate with other users.

Analyze

Once the data is organized and processed, it is ready for analytical use cases such as Business Intelligence and Machine Learning.

Business Intelligence

Ad hoc, interactive, high concurrency and low latency queries are common in BI use cases. Traditionally, they are implemented using OLAP and SQL queries. Databricks SQL offers SQL Endpoint, which is a computation resource that allows SQL commands to run on SQL objects (i.e., databases, tables, views, catalogs) stored in Delta Lake. Under the hood, it uses the Delta engine for executing SQL queries and is compliant with ANSI SQL standard. Users can create, configure, manage and monitor SQL endpoints with different cluster sizes for different use cases. Databricks has also built serverless SQL endpoints (In Public Preview), which uses compute resources managed by Databricks.

Databricks' SQL editor, with an inbuilt capability to visualize data, makes it very easy for data exploration. Users can create parameterized queries, scheduled queries/refresh, alerts and deliver results via REST APIs. It also supports SQL UDFs (User Defined Scalar Functions), which is user programmable code that executes on each row. These functions need to be registered before being utilized from other languages. Currently, all SQL endpoints use clusters from the data plane (customer cloud account), but Databricks also provides Serverless SQL Endpoints (In Public Preview), where serverless compute is used from the control plane (Databricks Cloud Account).

Machine Learning

Machine learning use cases involve data transformation, feature engineering, model training, experimentation, model deployment and monitoring. To enable and manage these end-to-end processes, Databricks provides Machine Learning Runtime, which includes Databricks Runtime and popular ML libraries like TensorFlow, Keras, PyTorch, MLflow, Horovod, GraphFrames, scikit-learn, XGboost, NumPy, MLeap and Pandas. When a user creates a cluster for machine learning use cases, all of these libraries are available by default in the environment. It removes the hassle of managing and keeping track of commonly used ML libraries. It also includes Python library — Hyperopt, which enables distributed hyperparameter tuning and model selection for single-node ML models, such as scikit-learn and TensorFlow, as well as distributed ML APIs like Apache Spark MLlib. Users can use Python, R, Java, Scala or SQL for data exploration, transformation and building machine learning models. As most of the included libraries are Python ML libraries, Python is the de facto language for Databricks Machine Learning.

Databricks also provides AutoML (In Public Preview) tool, which is a No Code/Low Code development environment. It automatically builds, tunes and evaluates baseline models by experimenting, and generates Python code. Users can review, reproduce and modify the code/experiments. Its limitation is that it only supports classification, regression and forecasting algorithms. Compared to leading market products, such as DataRobot, H2O.ai AutoML, it offers limited functionality.

Machine Learning Operations, aka MLOps

In addition to support for DataOps discussed earlier, Databricks helps implement MLOps with Managed MLflow. It is a proprietary and fully managed version of open-source MLflow to manage ML model life cycle. It tracks experimental runs, deploys models and maintains multiple versions of models in a centralized registry. With few clicks/code, users can move models from experimentation to production, securely and reliably. Compared to open-source MLflow, Managed MLflow has additional capabilities, such as managed tracking server, notebook integration, workspace integration, support for batch and streaming inference, ACL based stage transition (e.g., staging to production), role-based access to the resources.

Creating and maintaining features in machine learning projects is an iterative and time-consuming process. To solve this problem, Databricks has built a fully integrated capability called Feature Store (In Public Preview) to store reusable features in Delta Lake, create features registry and search the features. In addition, it manages access to features and tracks the lineage of the features from source dataset to all of its consumers — including models, endpoints, jobs and notebooks. It supports offline feature stores for all three cloud providers (AWS, Azure, GCP) and online feature stores for AWS and Azure only. Notable observed limitations of feature deletion are not supported.

Technical professionals with priority to implement feature stores or already using feature store products should integrate available products with Databricks (e.g., Amazon SageMaker Feature Store, Feast, Vertex AI Feature Store, Zipline, H2O.ai Feature Store, etc.).

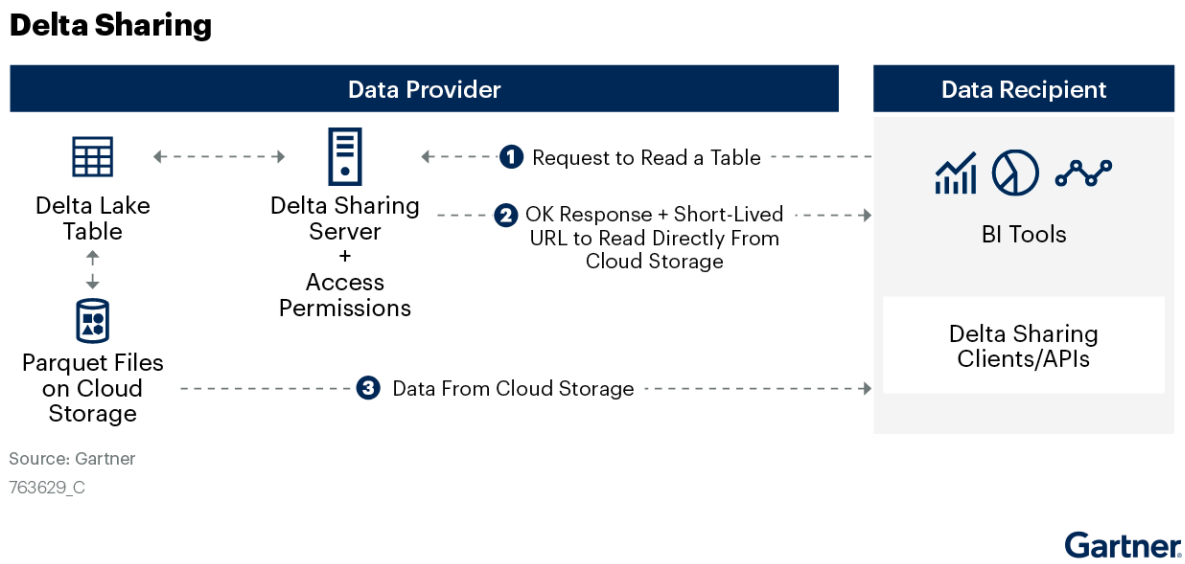
Deliver

Delivery of data, machine learning models and insights is a final and important process in the data and analytics architecture. It allows end users, downstream internal applications and third-party applications to consume delivery artifacts (i.e., data, models, dashboards, reports) for different business use cases.

Databricks uses Delta Sharing open protocol to securely share large datasets within and across the organizations. It is a simple REST protocol to share direct access to specific datasets from the host cloud storage. It also allows you to easily govern, track and audit access to your shared datasets. Delta Sharing Client APIs and Server are included in the Databricks platform, which enables easy sharing of Delta or Parquet format files. Currently, it only supports Delta or Parquet format files and does not support streams, views and arbitrary files, such as ML models.

External/internal applications can send data access requests to the server using Delta Sharing client APIs or third-party BI tools, which support Delta Sharing protocol. Then based on authentication and authorization of the user request, the server sends a short-lived pre-signed URL of the file in the cloud storage. Once receiving the URL, applications can directly access the file on cloud storage to transfer it parallely using cloud-native capabilities and bandwidth, bypassing the Delta Sharing Server. As Delta Lake data can be updated in real time with ACID transactions, receivers will always get the consistent and latest view of the data. Users can only share specific datasets or subsets of it by creating a SHARE object (a collection of tables to share) and using filters on it — currently, all Databricks BI and cloud partners support Delta Sharing protocol (i.e., Tableau, Microsoft Power BI, Looker, etc.) (see Figure 3).

Figure 3: Delta Sharing



Delivery of ML Models is an important process to make batch or stream inferences. Managed MLflow serves the models via Model Registry as a REST endpoint. As models are versioned, users can leverage them to use a particular version model for the inference. Model delivery via REST endpoint feature is currently In Public Preview, and available only for Python MLflow models.

Databricks recently acquired and released the official dashboarding tool Redash. It is an easy to use tool to connect to SQL endpoints, write SQL queries and generate different dashboards to deliver the insights. Data refresh intervals and triggers can be set to schedule dashboard refresh. Users can subscribe to a particular dashboard, as well as access different versions of the dashboard. Compared to market-leading BI products like Tableau, Power BI, Qlik, it has limited capabilities. Technical professionals, however, can integrate and continue using such leading or best-of-breed BI tools to deliver the insights.

Gartner Tips:

Regarding the completeness of Databricks platform, here are few things to consider:

- Databricks lacks data ingestion tool for on-premises/inter clouds use cases, as well as event ingestion tool. It is recommended to use required tools available from Databricks' cloud partner or technology partners, as they are natively integrated (e.g., in the case of Azure and Databricks combination, use Azure Event Hub and Azure Data Factory).
- Spark structured streaming is not a true real-time streaming yet. Use it for use cases where latency of ~100 ms or above is acceptable.
- Layered approach to Delta Lake is very simple and useful but, if implemented without planning and governance, may lead to data swamps. Particularly large organizations with many data sources and business units must focus on creating appropriate structure for these layers (e.g., single big Delta Lake or, business unit-wise, Delta Lakes). Additionally, it is highly recommended to capture all of the metadata and lineage from acquisition to delivery process, define data quality expectations and processes and roles to manage these activities.
- Databricks Delta Engine/Databricks Photon Engines are big data processing engines. Use them if you are migrating from the Hadoop or Apache Spark ecosystem. In case of migrating from a traditional Data Warehouse system or building a new system, it is highly recommended to evaluate all required features and test the performance of Databricks using queries with various complexity levels.

- Delta Live Tables and AutoML are In Public Preview and have limited capabilities compared to the market leading products with advanced capabilities, such as interactiveness or no code capability. Technical professionals looking for these advanced capabilities should leverage integration capabilities of the Databricks platform to integrate such best-of-breed tools.
- As Databricks doesn't have capabilities to design or build traditional reports, customers need to either integrate traditional reporting solutions to continue using it, or adopt and integrate advanced BI dashboard tools like Tableau, Power BI, Qlik or Looker.

1.2 Governance, Security and Compliance

Governance

Data and ML Governance is foundational for modern data and analytics architecture. Technical professionals planning to modernize their data and analytics architecture should equally focus on implementing governance as that of data storage, processing and analyzing. To implement governance, data and analytics platforms play an important role, enabling required capabilities like data quality management, data lineage, metadata management, model management and access management.

Databricks provides Delta Live Tables (In Public Preview) to define, enforce and monitor data quality via predefined error policies, data validity and integrity checks. In addition, it helps to track data lineage. On the ML side, MLflow helps model lineage tracking, model management and its access management. Databricks store all the table metadata in the central hive metastore. In addition, Databricks integrates with external hive metastore or other third-party metadata management tools, such as AWS Glue Data Catalog, Azure Data Catalog, Apache Atlas, Alation, Collibra and Informatica Enterprise Data Catalog.

Databricks has also launched its own proprietary data catalog called Unity Catalog/UC (In Public Preview), a multicloud centralized catalog for all Data and ML assets management and access management. Its main differentiating features are:

- **Row, column and view level access management:** Traditionally data lake storage systems (including S3, ADLS, GCS) provide only file/directory level access permissions, making it very difficult to share subset of data to specific users. UC enables permissions on tables (instead of files/folders) at row/column/view level, allowing easy granular access permissions to the end users.

- **Standardized open interface:** Many data catalogs in the market require either IAM access policy scripts or some other way to define access policies, which requires its understanding and skills to perform that task. UC provides ANSI SQL-based permission model, which also has a graphical user interface that makes it easy for database professionals, as well as data stewards, to create and maintain permission policies. Users can utilize this feature to manage access to various objects from Databricks platform, such as datasets, tables, view, ML models, ML Features and external data sources. It has also extended SQL model to support attribute based access control (e.g., users can tag multiple objects with common attributes, such as Personally Identifiable Information [PII] and apply the same policy to all those objects).

- **Centralized management for all data and ML assets:** UC can manage metadata, data catalog and access permissions across multiple workspaces, clouds and geographic locations, which gives users a central, seamless and unified view of all enterprise Data and ML assets.

- **Integrations with multiple data catalog/meta stores:** UC allows users to use existing Apache Hive metastore or other catalogs and governance tools like Privacera, Immuta, Alation and Collibra to integrate with it. It also allows other applications to securely access data or enforce security permissions via JDBC/ODBC or Delta Sharing protocol.

Security

- As Databricks is a cloud-based service, it provides an extra layer of security in addition to inherent host cloud security. Internally, it creates isolated environments (VPC/VNet); all of the customer data stays in the data plane with customer encryption keys, and all communication between data plane and control plane happens over a secure private network. Additionally, it also provides IP whitelisting and optional internode traffic encryption.

- The data that stays in the control plane (i.e., configurations, logs, user information, etc.) can be encrypted with customer managed keys (In Public Preview).

- Databricks supports column level, row level and data masking security by allowing users to create views and control the access. Though it's a simple form of security, it is hard to maintain once the number of views increases (e.g., a use case where data from one sales team should be restricted from others in big organizations with hundreds of sales territories).

- All accesses are granted through a system of least privileged access, based on individual user identity SSO (Single Sign-on) with SAML (Security Assertion Markup Language) or SCIM (System for Cross-domain Identity Management) via REST APIs. Users can be isolated at workspace level and cluster level. Additionally, it has integrations with third-party tools, such as Azure Active Directory or Okta.
- Databricks supports PrivateLink connections, which are a secure and reliable means to keep traffic entirely within a cloud provider's backbone network without exposing this traffic to the public internet.

Compliance

Databricks supports three security compliance standards: SOC 2 Type II, ISO/IEC 27018, ISO/IEC 27001 (Privacy focused). It also supports specific regulatory requirements, such as FedRAMP High, HITRUST, HIPAA and PCI — but customers should assess these capabilities based on their own security and governance policies.

Additionally, it helps comply with data protection laws such as General Data Protection Regulation (GDPR) and California Consumer Privacy Act (CCPA).

1.3 Deployment Support on Cloud

Databricks is a cloud-based data and analytics platform that can be deployed on single or multiple clouds. It doesn't support private cloud (e.g., IBM, Dell Technologies, SAP, HPE) or on-premises deployment. Currently, it only supports three major public cloud service providers: AWS, Azure and GCP, which are their official cloud partners. In these clouds, customers can create a VPC network to build a private environment. Databricks is built on foundational components/services of underlying cloud service providers (e.g., storage, compute, networking, administration and other support services). Table 4 lists how Databricks' offerings differ with the choice of underlying cloud:

Table 4: Databricks Deployment Support on Public Cloud

Functionality	AWS	Azure	GCP
Storage support	Simple Storage Service (S3)	ADLS Gen2, ADLS Gen1, Azure Blob Storage	Google Cloud Storage (GCS)
Compute cluster orchestration	Databricks Container Service	Databricks Container Service	Google Kubernetes Engine (GKE)
Integrations with the products from host cloud	AWS Glue, Amazon Redshift, Amazon SageMaker, Amazon Kinesis	Azure Data Factory, Azure Synapse Analytics, Azure Machine Learning, Power BI, Azure Event Hubs	BigQuery, Looker, Vertex AI, Pub/Sub

Source: Gartner (March 2022)

Technical professionals responsible for adopting Databricks must first define their single/hybrid/multicloud adoption strategy, and then Databricks adoption strategy. Especially with those who are migrating from on-premises, it's a critical step to define cloud strategy first and then Databricks adoption strategy (see How to Initiate the Selection of Strategic Cloud IaaS Providers for further information on selecting a cloud platform). Once a particular cloud is selected, select Databricks' offerings on that same cloud. Customers who want to migrate existing Databricks deployment from one cloud to another must evaluate Databricks offering from both of the clouds, compare equivalent individual components from them and create a strategy to mitigate the gaps or missing functionality. Also, it's important to note that the application performance, migration complexity and billing on Databricks on one cloud may differ from another cloud.

1.4 Integration With Third-Party Products

In a practical scenario, customers leverage third-party technologies from multiple vendors based on their requirements, talent available and cost. Integration and compatibility with such technologies, as well as the legacy systems, is one of the pivoting factors in selecting a platform like Databricks.

Databricks has focused on keeping its platform open to make it easy for customers to integrate the legacy products or the best-of-breed products. Databricks has inbuilt integration support with multiple technologies/products, as it is built on open-source technologies and open standards — which are integratable and compatible with many market products. In addition, Databricks has created an ecosystem of official cloud partners, technology partners and consulting partners. Such partnerships are very beneficial for all the parties involved — Customers, Databricks and its partners. Databricks has partnership with leading public cloud service providers (e.g., AWS, Azure cloud platform and GCP), 52 technology partners (e.g., Tableau, Power BI, Qlik, Alteryx, Looker, Matillion, Fivetran, MongoDB, TIBCO, Informatica, H2O.ai) and many consulting partners (e.g., Accenture, Capgemini and Cognizant).

Generally, when it comes to integrations with Data and Analytics platform, most of the partners are either integratable at the acquisition process or delivery process (e.g., data ingestion/ELT tools or BI/dashboard tools). Potentially, however, not many products are integratable in the organize or analyze process within the Data and Analytics platform. Databricks shines in this case because, in addition to being integratable at the acquisition and delivery process, it is integratable in the organize and analyze process as well. Delta Lake, Delta Engine, Databricks ML, Managed MLflow and Unity Catalog are modular and autonomous component, which can be used all together or in composable manner with Databricks partner products, such as Azure Data Lake, Azure ML, AWS SageMaker, H2O.ai, DataRobot, AWS Redshift and Google BigQuery. Customers can select best-of-breed or fit for purpose products and easily integrate with Databricks. To select the right products for a composable Data and Analytics architecture, it is highly recommended that technical professionals carefully evaluate capabilities of Databricks products and Databricks Partner products — with respect to their business and architectural requirements, available skills and the cost.

1.5 Scalability, Performance, Disaster Recovery/High Availability

Scalability

Databricks has implemented optimized autoscaling using periodic statistics on idle executors and location of intermediate files within the clusters. In the underutilization cluster scenario, it only removes idle workers (with no shuffle data) to scale down. This precise scale down approach leads to minimum wastage of compute resources and helps to maintain responsiveness of the clusters. It maintains an instance pool to achieve dynamic upscaling/downscaling.

Performance

Apache Spark has been a proven technology for high performance big data workloads. Databricks Delta Engine has added more optimizations for performance improvement. In addition, native vectorized Photon Engine (In Public Preview) improved the performance further, making it a competitive technology in the market for high concurrency, low latency BI workloads. Though many vendors in the market, including Databricks, provide their performance benchmarks, these benchmarks may not be applicable for your use cases (see Six Reasons to Ignore Vendor DBMS Benchmarks).

For comparing performance of different data and analytics products in the market, Gartner always recommends that organizations run a proof of concept (POC), using their own data and queries. Customers with traditional data warehouses/Hadoop/cloud data warehouses should run queries with various complexity levels and test the performance of Databricks against existing data systems and other alternative products during the POC. In addition to considering current performance requirements, customers should also consider their data growth rate and future performance requirements, which will help decide whether they should adopt it now, wait and adopt or not adopt at all. If there is substantial gain in terms of performance, cost and management efforts, then adopting Databricks is a valid and necessary decision to future-proof against the challenges of big data and analytics.

Disaster Recovery/High Availability (DR/HA)

Databricks recommends industry best practices for implementing DR/HA. As its hosted platform, DR/HA is a combined effort of host cloud and Databricks.

1.6 Ease and Speed of Development

Databricks provides easy, interactive and collaborative notebooks for writing data engineering and machine learning code in popular languages, such as Python, R, Scala and SQL. Under the hood, these languages use PySpark, SparkR, SparkSQL APIs to implement Spark applications. One of the good features of Databricks Notebook is that users can use multiple languages, shell commands, dbutil commands and markdown script in the same notebook by using the magic command %<language>. It makes it easy for development and swift collaboration with users having different language preferences. Notebooks also provide a feature to quickly visualize data inside the notebook. Git repository support makes collaboration and code versioning very easy.

Java developers can either load JARs of their code or use Databricks Connect for integrating popular Java IDEs to run their java code on Databricks clusters. Many companies have been implementing and maintaining Java-based Spark or Hadoop applications. For them, interactive notebooks may be useful only for data scientists for experimentation, but not for data and machine learning engineers who use Java to productionize Data/ML applications.

Additionally, it provides AutoML (In Public Preview), a Low Code/No Code to build machine learning models for citizen data scientists. For building data pipelines, it provides Delta Live Tables (In Public Preview), but for data transformations/processing, users need to write code in notebooks.

1.7 Ease of Operations and Management

Databricks is focused on making the platform simple and easy to manage. It provides a centralized environment called Workspace to access all features and the resources, such as delta storage, compute clusters, etc. It organizes Databricks objects, such as notebooks, libraries, dashboards and experiments into the folders and provides access to them. Customers can create separate workspaces for different use cases like development, staging and production. Users can easily create and manage clusters within workspace with specific runtime and worker/driver type. All notebooks/dashboards/experiments can be opened and managed from the workspace. With Databricks Repos, users can develop notebooks in workspace and use a remote Git repository for collaboration and version control. Managed MLflow makes end-to-end machine learning life cycle seamless and easy to follow, work and manage.

Collaborative and interactive notebooks, quick data visualizations in the notebooks, simple intuitive user interface, orchestration of data pipelines, scheduling of jobs and Managed MLflow to manage ML life cycle are all features that give a simplified and unified experience to the end users, which makes it easy to operationalize and manage.

1.8 Commitment to Open Source and Standards

Databricks supports, promotes and builds on open-source software and standards. It has been a major contributor in the big data open-source community and spurred innovation in the industry. Databricks was founded by innovators of open-source Apache Spark and originated as a managed open-source Apache Spark Platform. During these years, they built multiple open-source projects and standards, such as Delta Lake, MLflow and Delta Sharing. Databricks took the initiative to build the first open protocol for secure data sharing. It also acquired the commercial company behind an open-source project for visualization called Redash.

2. Non-technical Criteria

2.1 Skills Requirements/Development

As Databricks platform is based on Apache Spark technology, the main requirement for adopting it is understanding of Spark ecosystem, concepts, architecture and development skills in Spark compatible programming languages like Python, R, Scala, Java and SQL. Customers adopting Databricks lakehouse from data lake side have familiarity and expertise in architecture, concepts, development and programming languages. However, customers adopting from the traditional data warehouse side will need to build or acquire those skills. When a user selects a particular cloud service provider or third-party product, along with adoption of Databricks,it is very important and highly recommended to acquire required skills to use that cloud or third-party product. Databricks Labs initiative includes many sample projects and integration templates on GitHub to help customers expedite implementation of their use cases. The customers who are migrating from on-premises data and analytics systems to Databricks or planning to use hybrid cloud/multicloud approach with Databricks will need additional skill set of cloud platform management and hybrid cloud/multicloud management skills. In sum, migrating from on-premises traditional data and analytics systems to Databricks is a steep learning curve, but cloud based Databricks benefits are worth the efforts in growing big data use cases. This, however, is provided that Databricks satisfies customer-specific performance, cost and maintenance requirements via detailed evaluation during POCs.

Today, programming languages such as Python, Scala, Java, SQL are very popular among Data/ML engineers, whereas Python, R and SQL among data analysts and data scientists. Learning them is also becoming easier due to a vast array of resources on the internet, as well as free/paid training courses. Day by day, cloud adoption — especially public clouds (AWS, Azure, GCP) — is becoming very common and the first priority of businesses to reduce cost and improve flexibility, availability and reliability. There are many cloud-native developments, administration and management training courses/certifications available in the market. On the other hand, hiring candidates with expertise in the Databricks/Spark ecosystem (i.e., design, development or administration) is time-consuming and challenging. This challenge can be tackled to some extent by offering incentives to existing teams (with/without experience in Python, R, Java, Scala, SQL or Databricks/Spark ecosystem or cloud) for upskilling in required areas before adopting it.

There are many books in the market on related concepts published by different famous publishers (some of them sponsored by Databricks). It's always a best practice to learn from different resources to gain a diverse perspective on a topic. Additionally, Databricks academy offers paid instructor-led training for everyone, and free self-paced training for all of the customers. Users can also get Databricks certified once they pass paid certification exams from the Databricks academy.

2.2 Quality of Documentation

One big advantage of Databricks is that there is a large knowledge base and community support of the Apache Spark ecosystem available on the internet. This can be in the form of documentation, books, research papers, blogs and video recordings of Spark Summits, conferences, seminars and practical hands-on sessions.

Apart from that, Databricks also has its own official documentation, knowledge base and community to help users. Its documentation is very detailed, structured and available for each cloud service provider separately. Databricks' website has many blogs, articles, research papers, video recordings and books on many topics of Databricks implementation and customer use cases. As Databricks' platform has been rapidly evolving since its inception, some concepts or features have become stale and even obsolete. It is very difficult for the user to keep track of it, and even sometimes causes confusion. Users must be aware about the relevance of information (especially old recordings and PDFs on the website) with particular versions of the Databricks product.

Regarding the migration from Hadoop or data lake to Databricks, there is sufficient official information available on the website, whereas — for migration from data warehouses to Databricks — there is not much information available.

2.3. Future Technology Roadmap:

As per the news, updates and documentation from the official Databricks website, Databricks is working on below products to add and officially release them as part of Databricks Platform:

1. Photon (In Public Preview)
2. Serverless SQL (In Public Preview)
3. AutoML (In Public Preview)

4. Delta Live Tables (In Public Preview)

5. Unity Catalog (In Public Preview)

6. Feature Store (In Public Preview)

7. Delta Sharing for streams, SQL views and arbitrary files, such as machine learning models

Databricks is also improving their partner ecosystem via Partner Connect program to increase the number of technology product integrations with it, making it an open and flexible platform.

2.4 Customer Support

Customer support is an important contributing factor of a successful product implementation and maintenance. Databricks offer three tiers of support plan: Business, Enhanced and Production, in North America, Central Europe (CET), Singapore/China (SGT/CST) or Australia Eastern (AET) time zone. There are four levels of issue severities in the support process, and support cases can be opened via the help center portal. Databricks has differentiated single cloud and multicloud support service, where multicloud support is available for Enhanced and Production plans only.

Customers should note that this support is for Databricks managed services and products — and not for the underlying cloud service provider infrastructure/services/products used in the Databricks implementation.

Below are the rating from Gartner clients for customer support of Databricks:

- Databricks Data Science and Machine Learning Platform: 4.6/5.0 (Gartner Peer Insights)
- Databricks Data Management Platform: 4.7/5.0 (Gartner Peer Insights)

2.5 Licensing Cost

There are many pricing models that software vendors use to generate maximum revenue (e.g., yearly/monthly subscription, per user/group billing, etc.). All of these pricing models are different from cloud service provider pricing, which is based on compute resource usage per second/minute/hour, amount of storage per month or number of events. Databricks uses a similar approach to charge its customers, for only compute resources used per second.

As Databricks is a cloud-based platform, users should note here that Databricks pricing is only for Databricks services, and not for the underlying cloud service provider infrastructure and services. Before adopting Databricks, users need to buy cloud services and make a separate contract with the cloud service provider. Additionally, users need to pay separately to the cloud service provider for underlying cloud infrastructure/software and other services they use while using Databricks. If users don't use Databricks, they will still need to pay for cloud infrastructure usage, such as storage, network, etc.

Technical professionals already using cloud platforms are aware of the cloud pricing model, its challenges, need for performance optimization, high quality code development, standardized processes and administration. On the other hand, technical professionals using on-premises platforms are completely new to such challenges. They need to completely understand the pricing details, define the expense threshold, and observe and monitor resource consumption. In both the cases, it is highly recommended to focus on pricing factors while conducting POC for the Databricks. To see substantial gain in terms of cost, it normally takes at least six months with continuous monitoring, optimizing and fine tuning.

Databricks charge the customers USD per DBU (Databricks Unit, a unit of processing capability per hour, billed on a per-second usage). There is no upfront cost, and customers can pay as they go (provided that customers still need to pay to the underlying cloud service provider for infrastructure usage as per the contract). There is also a discounted usage commitment pricing option. Databricks has Standard and Premium Plans for all three cloud service providers, and additional Enterprise Plans for AWS only. Compared to the AWS Premium plan, the AWS Enterprise plan is designed for mission-critical workload, and has higher cost and few extra features, such as Customer Managed Keys, IP Access List and HIPAA Compliance. Core offerings are common to all clouds, but some offerings available only on particular clouds, (e.g., HIPAA Compliance support is available only on AWS and Azure).

Technical professionals planning to adopt Databricks should first finalize requirements, verify that required features are available on Databricks for preferred cloud, and then conduct comparative analysis of different pricing plans and corresponding Databricks offerings. Customers planning to use existing cloud infrastructure have to choose one of the Databricks plans available for that particular cloud only. On-premises customers planning to adopt Databricks need to first define cloud strategy, select cloud provider and then select one of the Databricks plans for that particular cloud.

Strengths

- **Data processing engine:** Core strength of Databricks platform is that it is built on high performance, scalable, flexible, robust and easy to use open-source data processing engine — Apache Spark. In addition, it further improved the overall performance of data engineering, SQL and ML workloads by implementing unique techniques as part of proprietary Delta Engine and Photon Engine (In Public Preview).

- **Delta lake:** With the addition of this fundamental component of data management, Databricks completes its end-to-end data analytics platform. It helps enable data lakehouse architecture, combining the best features of both worlds — data warehouse and data lake. Similar to data warehouse, it supports ACID transactions, schema enforcement, change-data-capture (CDC), slowly changing dimensions (SCD), auditing and time travel. And, similar to data lake, it supports petabyte scale processing of structured, semistructured and unstructured data, decoupled compute and storage, streaming workload and open standards. Unlike other MPP products (e.g., AWS Redshift + Athena), Databricks use the single paradigm to store and process structured, semistructured and unstructured data.

- **Fully managed ML life cycle:** Machine learning life cycle is inherently complex due to the iterative nature of the tasks like research, exploration, experimentation, model building, training, validation and deployment. Managing end-to-end ML life cycle is one of the pivoting factors of successful delivery of ML projects. Databricks provides all the features of open-source MLflow along with additional features, such as managed clusters for scaling, notebooks and workspace integration, CI/CD integration, feature store (In Public Preview), batch and streaming inference, and access management. This fully managed capability saves a lot of effort, time and cost of the projects.

- **Adherence to open standards and Integrations with leading data analytics products:** Databricks put a lot of effort into making it an open platform to make it easy for integration with a variety of products in the market, as well as legacy systems. It achieves that through creating open standards (i.e., Delta Sharing protocol or delta transaction log protocol), supporting existing open standards (i.e., JDBC/ODBC, OpenID, SAML or SCIM) and supporting open file formats, such as Parquet, Avro and LZO. Apart from this, Databricks has established an ecosystem of technology partners for making it easy for customers to onboard Databricks and complementary/supporting products from the partners. These integrations are foundational in the process of composing different components of end-to-end data and analytics architecture.

Weaknesses

- **Less appealing to citizen data scientists and business users**: Databricks is built on Apache Spark Ecosystem, which requires expertise in programming and technological concepts, making it less attractive to personas, such as business analyst, citizen data scientist and executive which prefer No Code/Low Code tools to perform data analysis.
- **Frequent updates/releases**: Databricks is continuously accommodating frequently released open-source software, introducing new features, products and partners — and changing platform taglines (e.g., Databricks Cloud, Databricks Unified Analytics Platform, Databricks Unified Data Platform, Databricks Unified Data Analytics Platform, The Databricks Lakehouse Platform), which seems a little overwhelming for new customers to keep track of its latest releases of this large ecosystem and new/changed platform taglines. Frequent releases also lead to substantial efforts to make required development, testing, operational or administration changes.
- **Increasing competition from cloud service providers**: With the new open concept of lakehouse, there is increasing competition with established cloud service providers having advantage of their own infrastructure, technological expertise, competitive and easy billing model. From the future customer's point of view, separate PaaS billing, security and maintenance of Databricks will be overhead if they can get similar performance and capabilities from their cloud service providers.
- **Traditional data warehouse to Databricks ecosystem is a steep route:** For customers with on-premises data warehouse infrastructure and expertise, adopting Databricks ecosystem and modern lakehouse architecture is a steep learning curve. And currently, most of the Databricks online contents/documentation is only focused on adoption of data lake house from data lake side.

Guidance

Adopting Databricks platform means making a fundamental shift in your traditional architecture and deployment location (in case of migration from on-premises). From an architectural perspective, you will either approach it from the data lake side or from the data warehouse side. An approach from data lake side will involve rearchitecting and adopting Delta Lake storage format and corresponding changes to leverage all of its features. An approach from the data warehouse side will either involve replacing traditional RDBMS data warehouse with Delta Lake, or building a hybrid architecture where Delta Lake feeds data warehouse. From a deployment perspective, you will need to define cloud strategy before adopting Databricks if you are currently using an on-premises deployment model. In any of these cases, it's a paradigm shift, and you will need to evaluate all capabilities of Databricks, understand lakhouse architecture, cloud infrastructure and then make a decision.

- In evaluating Databricks, it is important to clearly understand:
  - Which Databricks products are open-source, proprietary and third-party?
  - What are the capabilities of each Databricks product?
  - How are Databricks products integrated with each other?
  - Will current Databricks capabilities fulfill your future needs?
  - Which required capabilities are not available in Databricks or in preview release and need to buy them from other vendors?
  - How are third-party vendors integrated with Databricks?
  - What is the impact on cost when development workload is moved to Databricks (on cloud) from on-premises?
  - What are the skills and human resources required to fully implement and operationalize data and analytics applications on Databricks?

These are a few of many evaluation questions that need to be answered. Databricks has detailed documentation specific to all three leading public cloud service providers, and also good community support.

- Modern data and analytics architecture involves zoned/layered data organization. Make sure you define those zones as per the business requirements. Databricks documentation suggests at least three zones (bronze, silver, gold). Ensure underlying host cloud storage type is supported by Databricks. Delta Engine is a distributed processing engine which needs a new approach to develop data pipelines, as well as machine learning pipelines in case of migration from traditional data warehouses. As Lambda architecture is complex and difficult to manage, replace it with near real-time structured streaming provided by Databricks — unless there is a true real-time requirement. It will make the architecture simple and will save cost on maintenance, as well as future development of real-time use cases. Implementing standard processes around data management and machine learning is vital to reap the benefits of adopting Databricks platform (e.g., multilayer data processing, code/query optimization, sandbox creation and adoption, end-to-end pipeline monitoring, resource utilization and cost monitoring, MLflow standard processes, monitor and detect model drift or data drift, etc.). It is also highly recommended to create data governance policies and processes, and implement them from the beginning. Capture all the metadata and catalog all the data assets from the data acquisition stage itself. Define roles and responsibilities, and implement appropriate access management for data, code, projects, workspaces, features, models and insights.
- Databricks provides a distributed storage model (Delta Lake), distributed compute engine (Delta Engine), open data sharing (Delta Sharing), Data and ML pipeline orchestration (Jobs, Delta Live Tables and Managed MLflow), data catalog (Unity Catalog), feature storage (Feature Store) and centralized management. All of these features are integral to modern composable data and analytics architecture. Compose and integrate best-of-breed or fit for purpose products from Databricks partners or open-source softwares with Databricks, to gain maximum benefits. Below are some examples of compossibility:
  - Azure Data Factory + (Azure Blob Storage + Delta Lake) + Databricks Runtime +Azure ML + Databricks Managed MLflow + Power BI + Databricks Unity Catalog
  - AWS Glue + (AWS S3 + AWS Lake Formation) + Databricks Runtime ML + Databricks Managed MLflow + Tableau + AWS Glue Data Catalog

Recommended by the Author

Some documents may not be available as part of your current Gartner subscription.

Exploring Lakehouse Architecture and Use Cases

Solution Path for Building a Holistic Data Management and Analytics Architecture

A Guidance Framework for Deploying Data and Analytics in the Cloud

Demystifying XOps: DataOps, MLOps, ModelOps, AIOps and Platform Ops for AI

Building a Comprehensive Governance Framework for Data and Analytics

---

Table 1: Gartner Assessment Criteria

1. Technical Criteria	2. Non-technical Criteria
1.1 Completeness of Solution	2.1 Skills Requirements/Development
1.2 Governance, Security and Compliance	2.2 Quality of Documentation
1.3 Deployment Support on Cloud	2.3 Future Technology Roadmap
1.4 Integration With Third-Party Products	2.4 Customer Support
1.5 Scalability, Performance, Disaster Recovery/High Availability	2.5 Licensing Cost
1.6 Ease and Speed of Development	
1.7 Ease of Operations and Management	
1.8 Commitment to Open Source and Standards	

Source: Gartner (March 2022)

Table 2: Databricks Support for Data Sources and Types

Data Source	Supported Tables
Files	Apache Avro, Binary, CSV, Apache Hive Table, Image JSON, LZO compressed, MLflow experiment, Apache Parquet, XML, Zip
Data Storage Systems	AWS Redshift, Amazon Simple Storage Service (Amazon S3), Amazon S3 Select, Azure Blob Storage, Azure Data Lake Storage Gen1, Azure Data Lake Storage Gen2, Azure Cosmos DB, Azure Synapse Analytics, Apache Cassandra, Couchbase, Elasticsearch, Google BigQuery, Google Cloud Storage, MongoDB, Neo4j, Oracle, Redis, Riak TS, Snowflake, SQL databases using JDBC

Source: Gartner (March 2022)

Table 3: Databricks Support for Data Acquisition

Data Acquisition From Host Cloud Storage	Data Acquisition From On-premises, Intercloud and Third-party Sources
■ Spark structured streaming API ingest data from host cloud storage using Auto Loader and SQL COPY command. ■ Auto Loader feature provides an optimized cloud file source, which incrementally and efficiently discovers a large number of new files landed on the host cloud storages (e.g., Amazon S3, ADLS Gen2, ADLS Gen1, Azure Blob Storage, Google BigQuery, Google Cloud Storage, HDFS). It provides the Trigger Once mode, which makes microbatching run periodically, rendering it a batch mode. Additionally, it also provides features, such as schema inference and support for handling schema evolution. ■ Users who prefer SQL can use SQL COPY INTO command to ingest the batch data with exactly once semantics. ■ SQL CREATE TABLE [USING] feature allows creation of External Tables to access data from external data systems.	■ Databricks doesn't have in-house products to ingest data from on-premises, legacy systems (e.g., Mainframes), other cloud service providers or third-party applications (e.g., Salesforce, Twitter, etc.). ■ To fulfill such an important use case, Databricks has created an ecosystem of technology partners, which are established fit for purpose products like Fivetran, Qlik, Infoworks, StreamSets and Syncsort. ■ Users can also use the products from cloud service providers, such as AWS Glue, Azure Data Factory and Google Cloud Data Fusion.

Source: Gartner (March 2022)

Table 4: Databricks Deployment Support on Public Cloud

Functionality	AWS	Azure	GCP
Storage support	Simple Storage Service (S3)	ADLS Gen2, ADLS Gen1, Azure Blob Storage	Google Cloud Storage (GCS)
Compute cluster orchestration	Databricks Container Service	Databricks Container Service	Google Kubernetes Engine (GKE)
Integrations with the products from host cloud	AWS Glue, Amazon Redshift, Amazon SageMaker, Amazon Kinesis	Azure Data Factory, Azure Synapse Analytics, Azure Machine Learning, Power BI, Azure Event Hubs	BigQuery, Looker, Vertex AI, Pub/Sub

Source: Gartner (March 2022)