



Databricks on AWS – An Architectural Perspective (part 1)



(//www.linkedin.com/i
n/jongaraialdegalle
o/)

Jon Garaialde (//www.linkedin.com/in/jongaraialdegalle/)
Cloud Data Solutions Engineer/Architect



(//www.linkedin.com/i
n/ruben-villa-munoz/)

Rubén Villa (//www.linkedin.com/in/ruben-villa-munoz/)
Big Data & Cloud Architect



([//www.linkedin.com/in/alfonsojerezizquierdo/](https://www.linkedin.com/in/alfonsojerezizquierdo/))

Alfonso Jerez ([//www.linkedin.com/in/alfonsojerezizquierdo/](https://www.linkedin.com/in/alfonsojerezizquierdo/))

Analytics Engineer | GCP | AWS | Python Dev | Azure | Databricks | Spark



([//www.linkedin.com/in/albertojaenrevuelta/](https://www.linkedin.com/in/albertojaenrevuelta/))

Alberto Jaén ([//www.linkedin.com/in/albertojaenrevuelta/](https://www.linkedin.com/in/albertojaenrevuelta/))

Cloud Engineer | 3x AWS Certified | 2x HashiCorp Certified | GitHub: ajaen4



Databricks has become a reference product in the field of unified analytics platforms for creating, deploying, sharing, and maintaining data solutions, providing an environment for engineering and analytical roles. Since not all organizations have the same types of workloads, Databricks has designed different plans that allow adaptation to various needs, and this has a direct impact on the architecture design of the platform.

With this series of articles, the goal is to address the integration of Databricks in AWS environments, analyzing the alternatives offered by the product in terms of architecture design. Additionally, the advantages of the Databricks platform itself will be discussed. Due to the extensive content, it has been considered convenient to divide them into three parts:

First installment:

1. Introduction.
2. Data Lakehouse & Delta.
3. Concepts.
4. Architecture.
5. Plans and types of workloads.

6. Networking.

Second installment:

7. Security.

8. Persistence.

9. Billing.

Introduction

Databricks is created with the idea of developing a unified environment where different profiles, such as Data Engineers, Data Scientists, and Data Analysts, can collaboratively work without the need for external service providers to offer the various functionalities each one needs in their daily tasks.

Databricks' workspace provides a unified interface and tools for a variety of data tasks, including:

- Programming and administration of data processing.
- Dashboard generation and visualizations.
- Management of security, governance, high availability, and disaster recovery.
- Data exploration, annotation, and discovery.
- Modeling, monitoring, and serving of Machine Learning (ML) models.
- Generative AI solutions.

The birth of Databricks is made possible through the collaboration of the founders of Spark, who released Delta Lake and MLFlow as Databricks products following the open-source philosophy.



Spark, Delta Lake and MLFlow partnership

This new collaborative environment had a significant impact upon its introduction due to the innovations it offered by integrating different technologies:

- **Spark:** A distributed programming framework known for its ability to perform queries on Delta Lakes at cost/time ratios superior to competitors, optimizing the analysis processes.
- **Delta Lake:** Positioned as Spark's storage support, Delta Lake combines the main advantages of Data Warehouses and Data Lakes by enabling the loading of both structured and unstructured information. It uses an enhanced version of Parquet that supports ACID transactions, ensuring the integrity of information in ETL processes carried out by Spark.
- **MLFlow:** A platform for managing the end-to-end lifecycle of Machine Learning, including experimentation, reusability, centralized model deployment, and logging.

Data Lakehouse & Delta

A Data Lakehouse is a data management system that combines the benefits of Data Lakes and Data Warehouses.

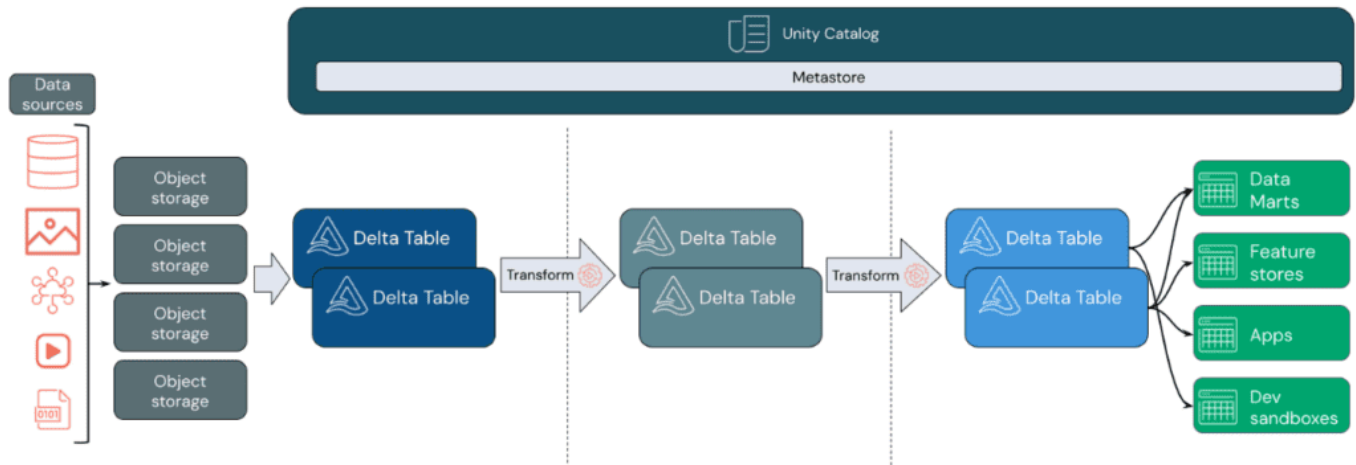


Diagrama de un Data Lakehouse (fuente: Databricks)

A Data Lakehouse provides scalable storage and processing capabilities for modern organizations aiming to avoid isolated systems for processing different workloads such as Machine Learning (ML) and Business Intelligence (BI). A Data Lakehouse can help establish a single source of truth, eliminate redundant costs, and ensure data freshness.

Data Lakehouses employ a data design pattern that gradually enhances and refines data as it moves through different layers. This pattern is often referred to as a medallion architecture.

Databricks relies on Apache Spark, a highly scalable engine that runs on compute resources decoupled from storage.

Databricks' Data Lakehouse utilizes two key additional technologies:

1. **Delta Lake:** An optimized storage layer that supports ACID transactions and schema enforcement.
2. **Unity Catalog:** A unified and detailed governance solution for data and artificial intelligence.

Data Design Pattern:

Data Ingestion: In the ingestion layer, data arrives from various sources in batches or streams, in a wide range of formats. This initial stage provides an entry point for raw data. By converting these files into Delta tables, Delta Lake's schema enforcement capabilities can be leveraged to identify and handle missing or unexpected data. Unity Catalog can be used to efficiently manage and log these tables based on data governance requirements and necessary security levels, allowing tracking of data lineage as it transforms and refines.

Processing, Cleaning, and Data Integration: After data verification, selection, and refinement take place. In this stage, data scientists and machine learning professionals often work with the data to combine, create new features, and complete cleaning. Once the data is fully cleaned, it can be integrated and reorganized into tables designed to meet specific business needs. The write-schema approach, along with Delta's schema evolution capabilities, allows changes in this layer without rewriting the underlying logic providing data to end users.

Data Serving: The final layer provides clean and enriched data to end users. The end tables should be designed to meet all usage needs. Thanks to a unified governance model, data lineage can be tracked back to its single source of truth. Optimized data designs for various tasks enable users to access data for machine learning applications, data engineering, business intelligence, and reporting.

Features:

- The Data Lakehouse concept leverages a Data Lake to store a wide variety of data in low-cost storage systems, such as Amazon S3 in this case.
- Catalogs and schemas are used to provide governance and auditing mechanisms, allowing Data Manipulation Language (DML) operations through various languages, and storing change histories and data snapshots. Role-based access controls are applied to ensure security.

- Performance and scalability optimization techniques are employed to ensure efficient system operation.
- It allows the use of unstructured and non-SQL data, facilitating information exchange between platforms using open-source formats like Parquet and ORC, and offering APIs for efficient data access.
- Provides end-to-end streaming support, eliminating the need for dedicated systems for real-time applications. This is complemented by parallel massive processing capabilities to handle diverse workloads and analyses efficiently.

Concepts: Account & Workspaces

In Databricks, a workspace is an implementation of Databricks in the cloud that serves as an environment for your team to access Databricks assets. You can choose to have multiple workspaces or just one, depending on your needs.

A Databricks account represents a single entity that can include multiple workspaces. Unity Catalog-enabled accounts can be used to centrally manage users and their data access across all workspaces in the account. Billing and support are also handled at the account level.

Billing: Databricks Units (DBUs)

Databricks invoices are based on Databricks Units (DBUs), processing capacity units per hour based on the type of VM instance.

Authentication & Authorization

Concepts related to Databricks identity management and access to Databricks assets.

- **User:** A unique individual with access to the system. User identities are represented by email addresses.
- **Service Principal:** Service identity for use with jobs, automated tools, and systems like scripts, applications, and CI/CD platforms. Service entities are represented by an application ID.

- **Group:** A collection of identities. Groups simplify identity management, making it easier to assign access to workspaces, data, and other objects. All Databricks identities can be assigned as group members.
- **Access control list (ACL):** A list of permissions associated with the workspace, cluster, job, table, or experiment. An ACL specifies which users or system processes are granted access to objects and what operations are allowed on the assets. Each entry in a typical ACL specifies a principal and an operation.
- **Personal access token:** A opaque string for authenticating with the REST API, SQL warehouses, etc.
- **UI (User Interface):** Databricks user interface, a graphical interface for interacting with features such as workspace folders and their contained objects, data objects, and computational resources.

Data Science & Engineering

Tools for data engineering and data science collaboration.

- **Workspace:** An environment to access all Databricks assets, organizing objects (Notebooks, libraries, dashboards, and experiments) into folders and providing access to data objects and computational resources.
- **Notebook:** A web-based interface for creating data science and machine learning workflows containing executable commands, visualizations, and narrative text.
- **Dashboard:** An interface providing organized access to visualizations.
- **Library:** A available code package that runs on the cluster. Databricks includes many libraries, and custom ones can be added.
- **Repo:** A folder whose contents are versioned together by synchronizing them with a remote Git repository. Databricks Repos integrates with Git to provide source code control and versioning for projects.

- **Experiment:** A collection of MLflow runs to train a machine learning model.

Databricks Interfaces

Describes the interfaces Databricks supports in addition to the user interface to access its assets: API and Command Line Interface (CLI).

- **REST API:** Databricks provides API documentation for the workspace and account.
- **CLI:** Open-source project hosted on GitHub. The CLI is based on Databricks REST API.

Data Management

Describes objects containing the data on which analysis is performed and feeds machine learning algorithms.

- **Databricks File System (DBFS):** Abstraction layer over a blob store. It contains directories, which can hold files (data files, libraries, and images) and other directories.
- **Database:** A collection of data objects such as tables or views and functions, organized for easy access, management, and updating.
- **Table:** Representation of structured data.
- **Delta table:** By default, all tables created in Databricks are Delta tables. Delta tables are based on the open-source Delta Lake project, a framework for high-performance ACID table storage in cloud object stores.
- **Metastore:** Component storing all the structure information of different tables and partitions in the data store, including column and column type information, serializers and deserializers needed to read and write data, and the corresponding files where data is stored.
- **Visualization:** Graphical representation of the result of executing a query.

Computation Management

Describes concepts for executing computations in Databricks.

- **Cluster:** A set of configurations and computing resources where Notebooks and jobs run.
There are two types of clusters: all-purpose and job.
 - An all-purpose cluster is created manually through the UI, CLI, or REST API and can be manually terminated and restarted.
 - A job cluster is created when running a job on a new job cluster and terminates when the job is completed. Job clusters cannot be restarted.
- **Pool:** A set of instances ready for use that reduces cluster start times and enables automatic scaling. When attached to a pool, a cluster assigns driver and worker nodes to the pool. If the pool doesn't have enough resources to handle the cluster's request, the pool expands by assigning new instances from the instance provider.
- **Databricks Runtime:** A set of core components running on clusters managed by Databricks. There are several runtimes available:
 - Databricks runtime includes Apache Spark and adds components and updates to improve usability, performance, and security.
 - Databricks runtime for Machine Learning is based on Databricks runtime and provides a pre-built machine learning infrastructure that integrates with all Databricks workspace capabilities.

Workflows

Frameworks for developing and running data processing pipelines:

- **Jobs:** Non-interactive mechanism to run a Notebook or library either immediately or on a schedule.
- **Delta Live Tables:** Framework for creating reliable, maintainable, and auditable data processing pipelines.

- **Workload:** Databricks identifies two types of workloads subject to different pricing schemes:
 - **Data Engineering (job):** An (automated) workload running on a job cluster that Databricks creates for each workload.
 - **Data Analysis (all-purpose):** An (interactive) workload running on an all-purpose cluster. Interactive workloads typically execute commands within Databricks Notebooks.
- **Execution context:** State of a Read-Eval-Print Loop (REPL) environment for each supported programming language. Supported languages are Python, R, Scala, and SQL.

Machine Learning

End-to-end integrated environment incorporating managed services for experiment tracking, model training, function development and management, and serving functions and models.

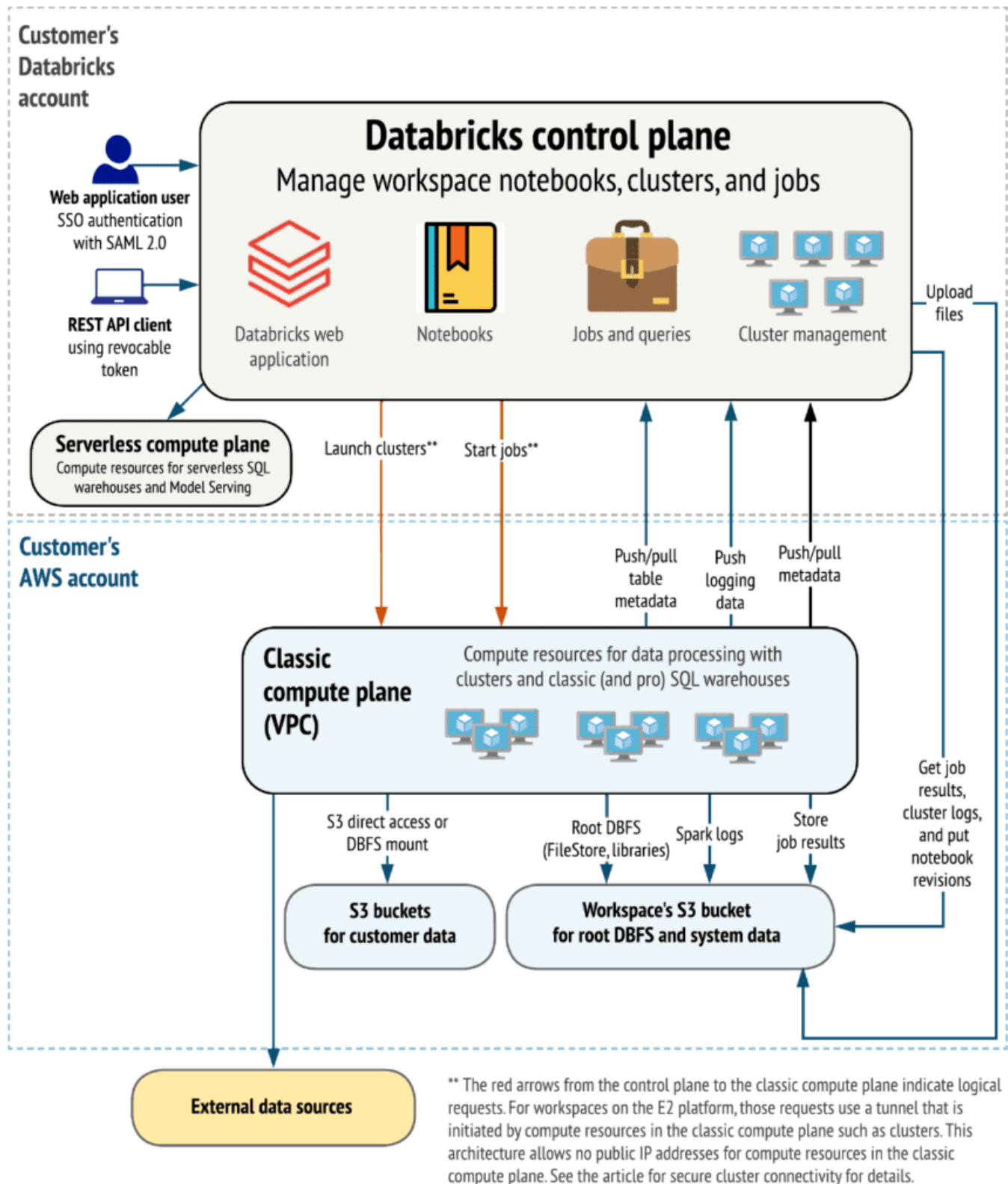
- **Experiments:** Primary unit of organization for tracking the development of machine learning models.
- **Feature Store:** Centralized repository of features enabling sharing and discovery of functions across the organization, ensuring the same function calculation code is used for both model training and inference.
- **Models & model registry:** Machine learning or deep learning model registered in the model registry.

SQL

- **SQL REST API:** Interface allowing automation of tasks on SQL objects.
- **Dashboard:** Representation of data visualizations and comments.
- **SQL queries:** SQL queries in Databricks.
 - **Query:** SQL query.
 - **SQL warehouse:** SQL storage.
 - **Query history:** History of queries.

Architecture: High-level architecture

Before we start analyzing the various alternatives that Databricks offers for infrastructure deployment, it is advisable to understand the main components of the product. Below is a high-level overview of the Databricks architecture, including its enterprise architecture, in conjunction with AWS.



High-level Architecture Diagram (source: Databricks)

Although architectures may vary based on custom configurations, the above diagram represents the structure and most common data flow for Databricks in AWS environments.

The diagram outlines the general architecture of the classic compute plane. Regarding the architecture for the serverless compute plane used for serverless SQL pools, the compute layer is hosted in a Databricks account instead of an AWS account.

Control plane and compute plane:

Databricks is structured to enable secure collaboration in multifunctional teams while maintaining a significant number of backend services managed by Databricks. This allows you to focus on data science, data analysis, and data engineering tasks.

- The control plane includes backend services that Databricks manages in its Databricks account. Notebooks and many other workspace configurations are stored in the control plane and encrypted at rest.
- The compute plane is where data is processed.

For most Databricks computations, computing resources are in your AWS account, referred to as the classic compute plane. This pertains to the network in your AWS account and its resources. Databricks uses the classic compute plane for its Notebooks, jobs, and classic and professional Databricks SQL pools.

As mentioned earlier, for serverless SQL pools, serverless computing resources run in a serverless compute plane in a Databricks account.

Databricks has numerous connectors to link clusters to external data sources outside the AWS account for data ingestion or storage. These connectors also facilitate ingesting data from external streaming sources such as event data, streaming data, IoT data, etc.

The Data Lake is stored at rest in the AWS account and in the data sources themselves to maintain control and ownership of the data.

E2 Architecture:

The E2 platform provides features like:

- Multi-workspace accounts.
- Customer-managed VPCs: Creating Databricks workspaces in your VPC instead of using the default architecture where clusters are created in a single AWS VPC that Databricks creates and configures in your AWS account.
- Secure cluster connectivity: Also known as “No Public IPs,” secure cluster connectivity allows launching clusters where all nodes have private IP addresses, providing enhanced security.
- Customer-managed keys: Provide KMS keys for data encryption.

Workload plans and types

The price indicated by Databricks is attributed in relation to the DBUs consumed by the clusters. This parameter is associated with the processing capacity consumed by the clusters and directly depends on the type of instances selected (when configuring the cluster, an approximate calculation of the DBUs it will consume per hour is provided).

The price charged per DBU depends on two main factors:

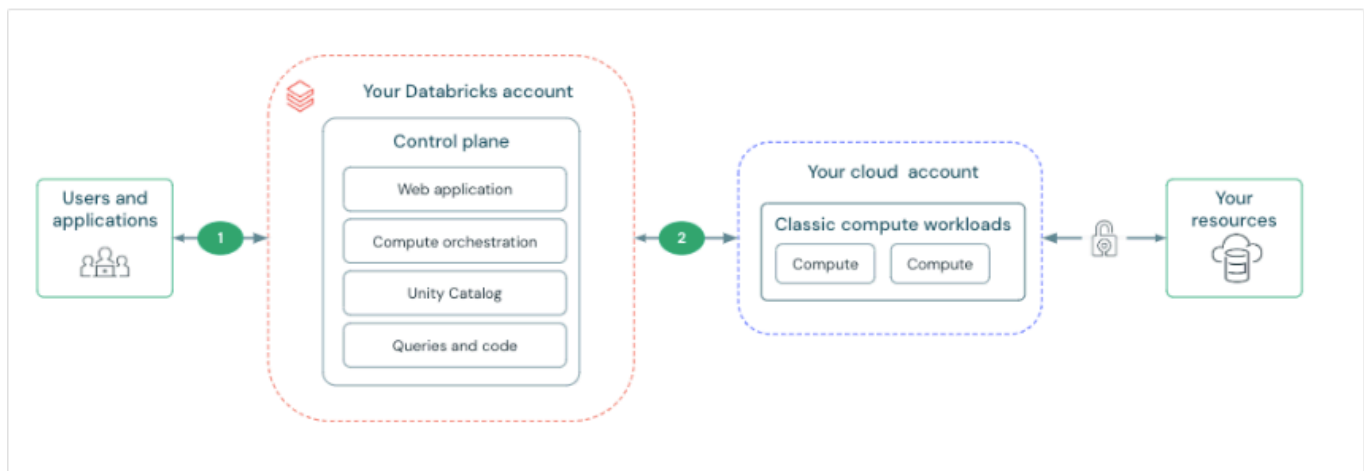
1. Computational factor: the definition of cluster characteristics (Cluster Mode, Runtime, On-Demand-Spot Instances, Autoscaling, etc.) that will result in the allocation of a specific package.
2. Architecture factor: customization of this (Customer Managed-VPC), in some aspects may require a Premium or even Enterprise subscription, causing the cost of each DBU to be higher as you obtain a subscription with greater privileges.

The combination of both computational and architectural factors will determine the final cost of each DBU per hour of operation.

All information regarding plans and types of work can be found at the following [link](https://www.databricks.com/product/aws-pricing) (<https://www.databricks.com/product/aws-pricing>).

Networking

Databricks has an architecture divided into control plane and compute plane. The control plane includes backend services managed by Databricks, while the compute plane processes the data. For classic computing and calculation, resources are in the AWS account in a classic compute plane. For serverless computing, resources run on a serverless compute plane in the Databricks account.



Thus, Databricks provides secure network connectivity by default, but additional features can be configured. Key points include:

- Connection between users and Databricks: This can be controlled and configured for private connectivity. Configurable features include:
 - Authentication and access control.
 - Private connection.
 - Access IP list.

- Firewall rules.
- Network connectivity features for the control plane and compute plane. Connectivity between the control plane and the serverless compute plane is always done through the cloud network, not over the public Internet. This approach focuses on establishing and securing the connection between the control plane and the classic compute plane. The concept of ‘secure cluster connectivity’ is worth noting, where, when enabled, the client’s virtual networks have no open ports, and Databricks cluster nodes do not have public IP addresses, simplifying network management. Additionally, there is the option to deploy a workspace within the Virtual Private Cloud (VPC) on AWS, providing greater control over the AWS account and limiting outbound connections. Other topics include the possibility of pairing the Databricks VPC with another AWS VPC for added security, and enabling private connectivity from the control plane to the classic compute plane through AWS PrivateLink.”

The following [link \(//docs.databricks.com/en/security/network/classic/index.html\)](https://docs.databricks.com/en/security/network/classic/index.html) is provided for more information on these specific features.

Connections through Private Network (Private Links)

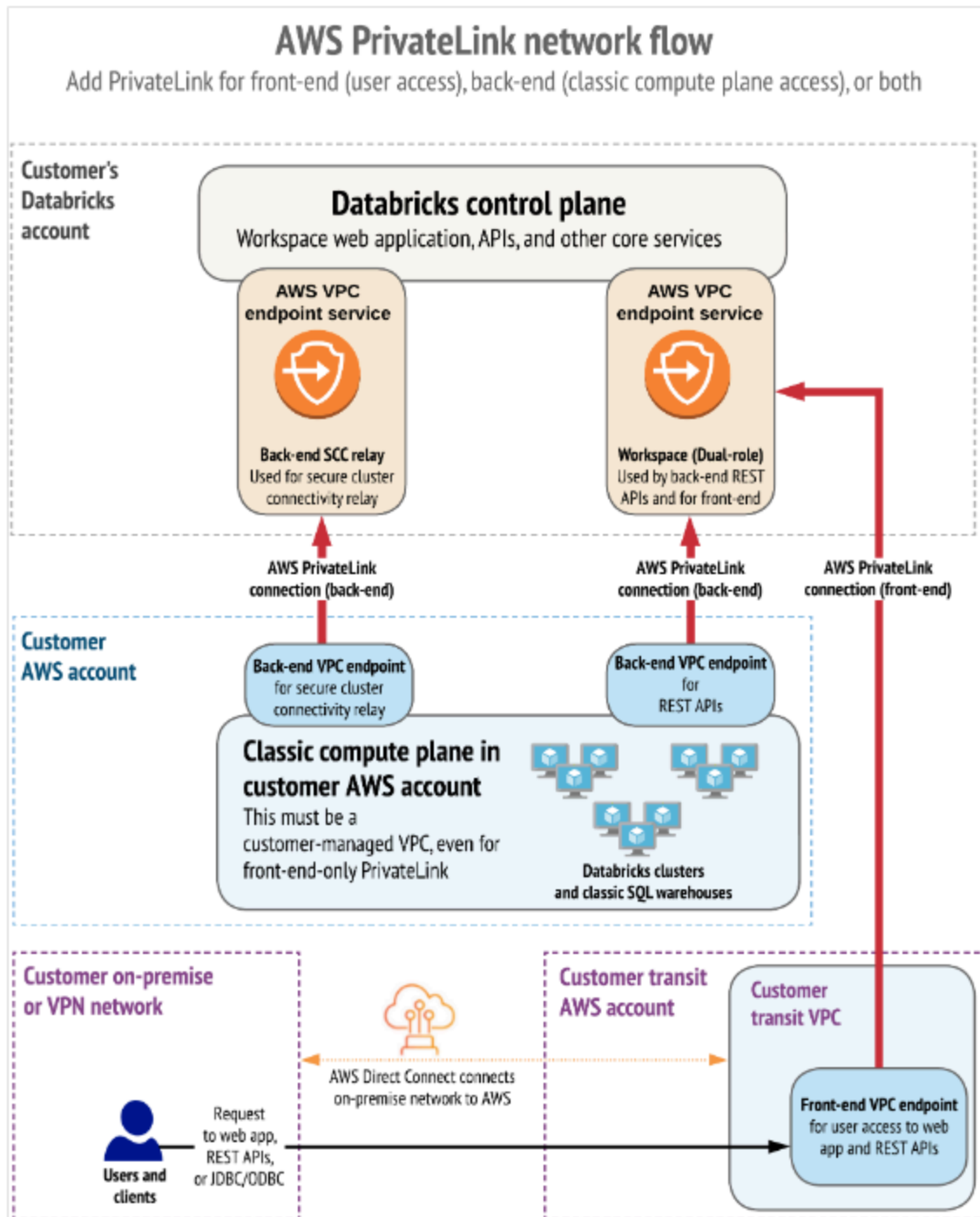
Finally, we want to highlight how AWS uses PrivateLinks to establish private connectivity between users and Databricks workspaces, as well as between clusters and the infrastructure of the workspaces.

AWS PrivateLink provides private connectivity from AWS VPCs and on-premises networks to AWS services without exposing the traffic to the public network. In Databricks, PrivateLink connections are supported for two types of connections: Front-end (users to workspaces) and back-end (control plane to control plane).

The front-end connection allows users to connect to the web application, REST API, and Databricks Connect through a VPC interface endpoint.

The back-end connection means that Databricks Runtime clusters in a customer-managed VPC connect to the central services of the workspace in the Databricks account to access the REST APIs.

Both PrivateLink connections or only one of them can be implemented.



Referencias

What is a data lakehouse? [[link \(//docs.databricks.com/en/lakehouse/index.html\)](https://docs.databricks.com/en/lakehouse/index.html)] (January 18, 2024)

Databricks concepts [[link \(//docs.databricks.com/en/getting-started/concepts.html\)](https://docs.databricks.com/en/getting-started/concepts.html)] (January 31, 2024)

Architecture [[link \(//docs.databricks.com/en/getting-started/overview.html\)](https://docs.databricks.com/en/getting-started/overview.html)] (December 18, 2023)

Users to Databricks networking [[link \(//docs.databricks.com/en/security/network/front-end/index.html\)](https://docs.databricks.com/en/security/network/front-end/index.html)] (February 7, 2024)

Secure cluster connectivity [[link \(//docs.databricks.com/en/security/network/classic/secure-cluster-connectivity.html\)](https://docs.databricks.com/en/security/network/classic/secure-cluster-connectivity.html)] (January 23, 2024)

Enable AWS PrivateLink [[link \(//docs.databricks.com/en/security/network/classic/privatelink.html\)](https://docs.databricks.com/en/security/network/classic/privatelink.html)] (February 06, 2024)

Do you want to know more about what we offer and to see other success stories?

**DISCOVER BLUETAB
(/es/)**



(//www.linkedin.com/in/jongaraialdegallgo/)

Jon Garaialde (//www.linkedin.com/in/jongaraialdegallgo/)

Cloud Data Solutions Engineer/Architect



(//www.linkedin.com/in/alfonsojerezizquierdo/)

Alfonso Jerez (//www.linkedin.com/in/alfonsojerezizquierdo/)

Analytics Engineer | GCP | AWS | Python Dev | Azure | Databricks | Spark



([//www.linkedin.com/in/ruben-villa-munoz/](https://www.linkedin.com/in/ruben-villa-munoz/))

Rubén Villa ([//www.linkedin.com/in/ruben-villa-munoz/](https://www.linkedin.com/in/ruben-villa-munoz/))

Big Data & Cloud Architect



([//www.linkedin.com/in/albertojaenrevuelta/](https://www.linkedin.com/in/albertojaenrevuelta/))

Alberto Jaén ([//www.linkedin.com/in/albertojaenrevuelta/](https://www.linkedin.com/in/albertojaenrevuelta/))

Cloud Engineer | 3x AWS Certified | 2x HashiCorp Certified | GitHub: ajaen4

SOLUTIONS, WE ARE EXPERTS

DATA STRATEGY

(/es/soluciones/data-strategy/)

DATA FABRIC

(/es/soluciones/data-fabric/)

AUGMENTED ANALYTICS

(/es/soluciones/augmented-analytics/)

You may be interested in

(../microsoft-fabric-una-nueva-solucion-de-analisis-de-datos-todo-en-uno/)

MICROSOFT FABRIC: Una nueva solución de análisis de datos, todo en uno (../microsoft-fabric-una-nueva-solucion-de-analisis-de-datos-todo-en-uno/)

October 16, 2023

LEER MÁS (../microsoft-fabric-una-nueva-solucion-de-analisis-de-datos-todo-en-uno/)

(../databricks-on-azure-an-architecture-perspective-part-1/)

Databricks on Azure – An Architecture Perspective (part 1) (../databricks-on-azure-an-architecture-perspective-part-1/)

February 15, 2022

LEER MÁS (../databricks-on-azure-an-architecture-perspective-part-1/)

(../how-much-is-your-customer-worth/)

How much is your customer worth? (../how-much-is-your-customer-worth/)

October 1, 2020

LEER MÁS (../how-much-is-your-customer-worth/)

[Legal \(../legal/\)](#)

[Privacy \(../privacy/\)](#)

[Cookies policy \(../cookies-policy/\)](#)

(/wp-

(/wp-

content/uploads/2025/02/DOC-170-Certificado-
de-Conformidad-
ENS_BLUETAB_MEDIA_2024_signed-1.pdf)

Patron

Sponsor

© 2025 Bluetab Solutions Group, SL. All rights reserved.