Microsoft

Synergy in Analytics: Unifying Azure Databricks and Microsoft Fabric

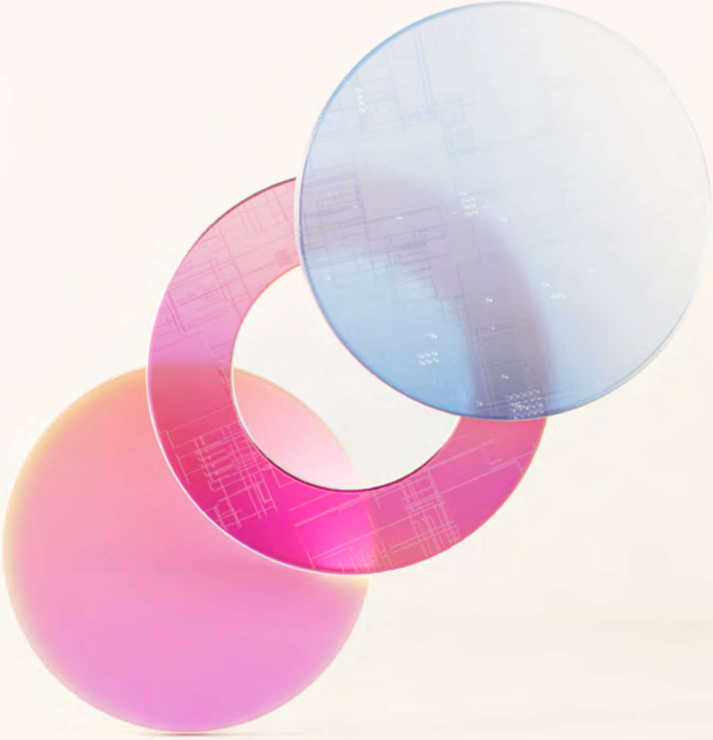
Synergy in Analytics: Unifying Azure Databricks and Microsoft Fabric

Empower modern data analytics with Azure Databricks and Microsoft Fabric

Today, organisations need to understand how to manage the ever-increasing flood of data in an efficient and insightful manner. A data lakehouse combines the vast storage of a data lake with the structured processing of various data services. It supports extensive data storage and complex analytics without compromise. It is more than a storage solution; it enhances data intelligence and supports advanced analytics, addressing the challenge of converting abundant data into actionable insights.

Cloud environments offer vast computational resources and scalability on demand; as an organisation's data grows, their infrastructure can grow alongside it seamlessly and cost-effectively. This synergy between cloud platforms and the data lakehouse architecture is pivotal, providing a resilient and adaptable foundation for any enterprise looking to thrive in the data-driven economy. Effective management and robust security measures are essential in the cloud-based data landscape to protect this invaluable asset.

Both Azure Databricks and Microsoft Fabric are comprehensive analytics solutions. Fabric has more business-user-friendly tools, and Azure Databricks has an integrated AI platform, but since they both rely on the same data layer, they can be used together as a more powerful whole. Azure Databricks, Fabric and OneLake allow organisations to streamline their data architecture, simplifying analytics workloads and enabling efficient data management and analysis across a unified platform.



Simplify analytics workloads with Azure Databricks and Microsoft Fabric

The modern data lakehouse architecture enables enterprises to utilise the synergies between Azure Databricks and Microsoft Fabric. Both Azure Databricks and Fabric offer a unified, comprehensive set of tools for a broad spectrum of advanced analytics scenarios and work together to provide a complete range of solutions for working with a data lakehouse. With elements from data engineering, data science, data warehousing and Power BI, they deliver wide-ranging analytics features, a cohesive experience for users and a single data repository accessible to various analytics tools. Azure Databricks also provides comprehensive governance and lineage tracking of both data and AI assets in a single unified experience.

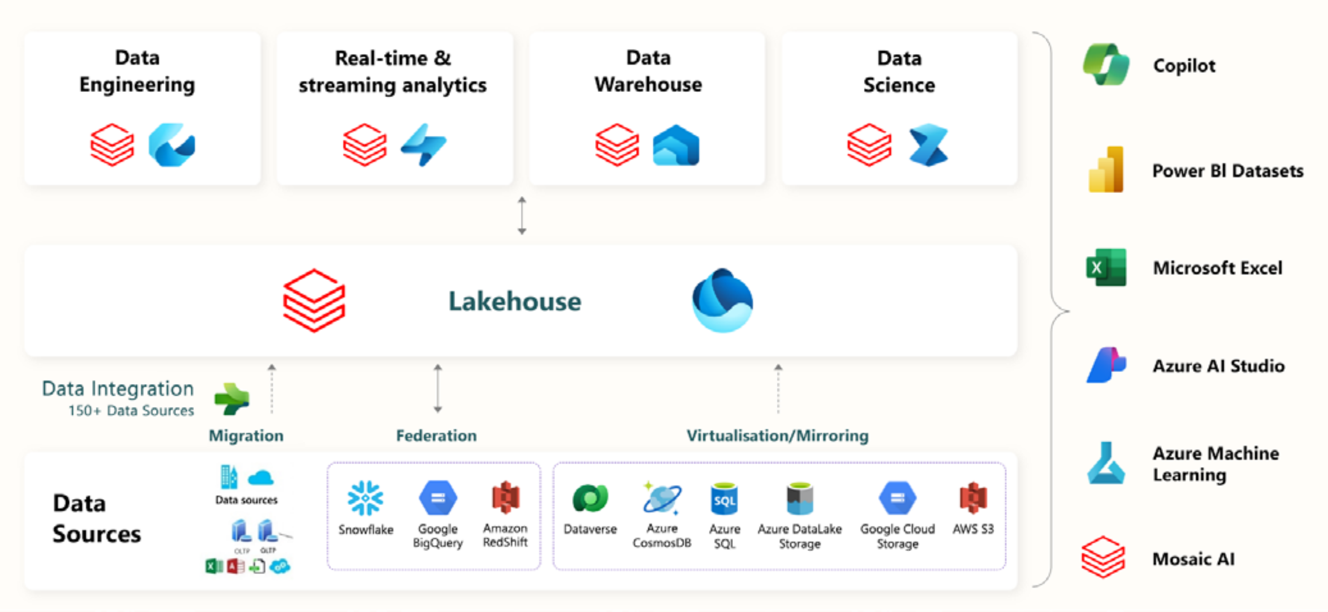


Figure 1: Azure Databricks and Microsoft Fabric integration in a lakehouse architecture

Maximise data potential with Azure Databricks and Microsoft Fabric

Azure Databricks and Fabric integration allows users to seamlessly switch between platforms, offering customers a cohesive and powerful solution for data management and analytics and facilitating AI and machine learning projects with ease and efficiency.

Data management

OneLake centralises data from diverse sources. The integration of Azure Databricks with Fabric not only revolutionises data management, scalability and data processing but also centralises data from a wide range of sources through OneLake. This comprehensive approach ensures that Azure Databricks can seamlessly connect with data stored in Azure Data Lake Storage (ADLS), various databases and OneLake itself. This simplifies the management of vast data volumes, enhances the ability to scale data projects and streamlines the data processing pipeline.

- **Centralised storage**: Using OneLake within Fabric allows centralised data management, which simplifies data access and governance, ensuring Azure Databricks can directly utilise the data for analytical processes.

- **Seamless integration**: The seamless integration between Azure Databricks and Data Factory in Fabric facilitates streamlined workflows from data ingestion and validation to transformation. This integration enables a cohesive data management strategy that supports data analytics, data science, as well as AI projects.

- **Enhanced security and accessibility**: Premium Azure Databricks workspaces support credential passthrough, strengthening the security and ease of access to OneLake resources. This feature ensures secure and straightforward access to the centralised data for further processing and analysis.

Scalability and reproducibility

Azure Databricks and Fabric support scalable data workflows, reproducible AI and analytics projects and dynamic data processing capabilities. This integration allows organisations to manage large volumes of data efficiently, ensure consistent results across their data environments and adapt processing power to meet diverse requirements, driving reliable and scalable data operations.

- **Scalable data workflows**: Azure Databricks activities within Data Factory are designed to support scalable data workflows. Organisations can efficiently handle vast amounts of data, scaling their data processing and analytics operations as needed without compromising performance or reliability.

- **Reproducible AI projects**: The integration ensures that AI and analytics projects are reproducible, benefiting from features such as data versioning and lineage tracking. These features, available natively on both platforms, enhance the reliability of AI projects and ensure consistency across data environments.

- **Dynamic data processing capabilities**: Azure Databricks offers dynamic data processing capabilities that adapt to varying data volumes and processing requirements. This flexibility is crucial for organisations to scale their data analytics operations efficiently.

Data processing

Combining the robust data transformation capabilities of Azure Databricks with the sophisticated orchestration of Data Factory pipelines, these technologies synergise to support a comprehensive range of data processing tasks.

- **Efficient data transformation**: Azure Databricks excels at transforming data stored in OneLake and other sources. Together, Azure Databricks and Fabric support an extensive range of data processing tasks, including data exploration, cleaning and preparation, crucial for preparing datasets for AI and machine learning.

- **Orchestration of complex workflows**: Data Factory pipelines that include Azure Databricks activities allow for the orchestration of complex data transformation workflows. These pipelines can validate data sources, copy data to designated storage and execute notebooks for data transformation, providing a comprehensive solution for data processing.

The synergy between Azure Databricks and Fabric, especially through Data Factory, enhances data management, ensures scalability and reproducibility and facilitates efficient data processing. This integration is vital for organisations using their data for insightful analytics and AI-driven decision making.

Advantages of lakehouse architecture

The lakehouse architecture is built on the open-source Delta Lake storage format. In addition to its technical capabilities, such as ACID transaction consistency, it enhances the overall effectiveness of the entire platform. Further, it enables use by multiple processing engines at the same time as it uses open formats and allows tools such as Azure Databricks and Fabric to work with the same copy of the data at the same time.

Enterprises don't have to rely on just one tool to process their data; instead, they can select the best tool for each project. Lakehouse architecture revolutionises how enterprises manage, scale and process their data. This innovative approach ensures data integrity and consistency through transactional support, fostering a more effective data management platform. Furthermore, the architecture's ability to amalgamate different storage formats under one roof simplifies the complex landscape of data estates. It facilitates the dynamic adjustment of computing resources in line with real-time demand, eliminating wasteful over-provisioning and enhancing cost efficiency and resource utilisation.

At the heart of this architecture is the integration of extensive data lakes and structured data warehouses, creating an optimal environment for fostering AI and machine learning innovation. This ensures access to both computational power and data, accelerating innovation and streamlining the management of diverse data systems.

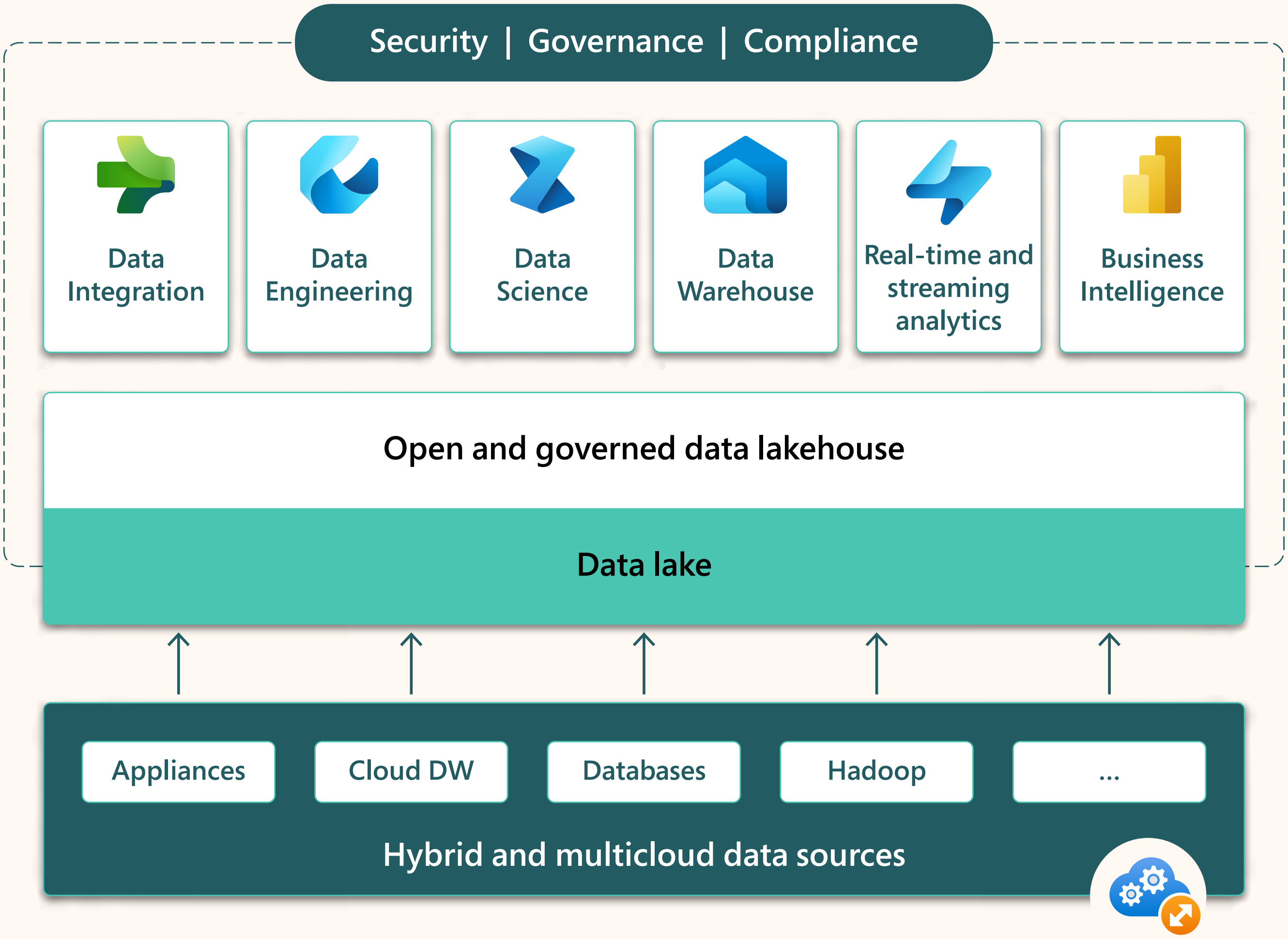

Security | Governance | Compliance
Data Integration
Data Engineering
Data Science
Data Warehouse
Real-time and streaming analytics
Business Intelligence
Open and governed data lakehouse
Data lake
Appliances
Cloud DW
Databases
Hadoop
...
Hybrid and multicloud data sources

Figure 2: Analytics in the lakehouse

With the unique combination of the vast storage capacity of a data lake and the structured, query-optimised environment of a data warehouse, the modern lakehouse emerges as the ideal platform for developing and deploying AI algorithms. This dual capability ensures that AI projects can utilise the necessary computational power and data accessibility, speeding up innovation and reducing overhead costs associated with managing separate data systems.

By simplifying data architecture and reducing infrastructure complexity, businesses can focus on creating value through AI rather than grappling with data management challenges in the following ways:

- The lakehouse architecture stores Delta Lake files in an ADLS account. This cloud storage service is extremely cost effective, and the Delta Lake format allows the storage of both structured and unstructured data.

- Building AI models requires massive amounts of compute power from both traditional CPUs and advanced GPUs. Since the data lakehouse architecture empowers the use of multiple compute engines, including both Azure Databricks and Fabric, enterprises can bring the right type of processing power to their data exploration and data modelling tasks.

Enterprises can use the advanced machine learning and AI capabilities of Azure Databricks and Fabric on their full data estate stored in a data lakehouse. These tools include end-to-end experiment management and automated machine learning toolkits that can super-charge AI projects.

Medallion architecture in Azure Databricks and Microsoft Fabric

The medallion architecture is a sophisticated approach within the broader concept of the lakehouse architecture, designed to streamline data workflows from ingestion to insights. At its core, it consists of three layers: bronze, silver and gold, each serving a distinct purpose in the data lifecycle.

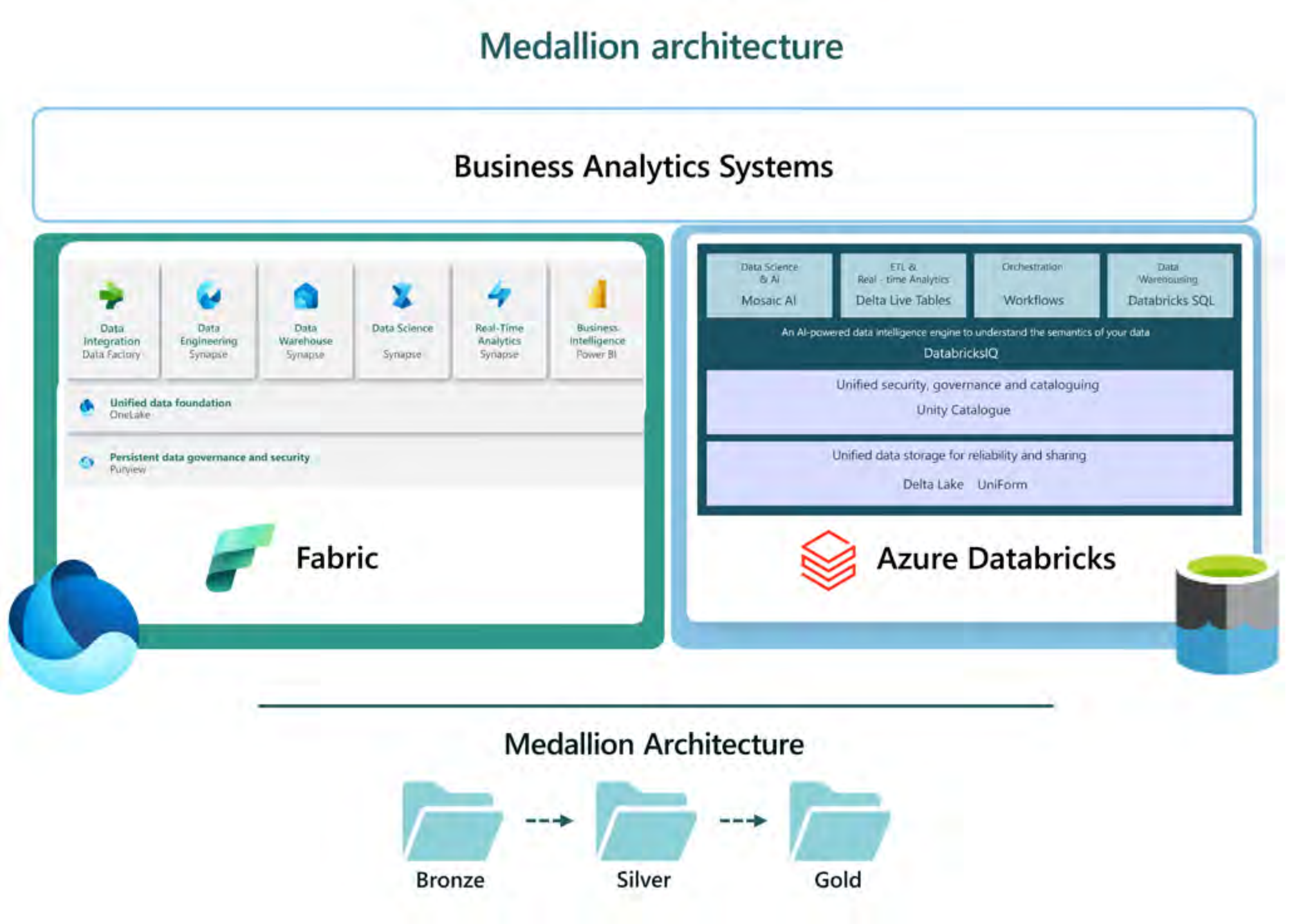


Figure 3: Medallion architecture

The three layers of the medallion architecture are:

1. **Bronze (raw)**: In this layer, raw data is initially ingested, retaining its original form. It acts as a staging area and is crucial for capturing the full granularity of data without any loss of fidelity.

2. **Silver (validated)**: In this layer, data from disparate sources is matched, merged and conformed, making it ready for more complex analytical tasks. The silver layer is designed to provide an enterprise view of key business entities and is critical for supporting self-service analytics and intermediate data storage needs.

3. **Gold (enriched)**: In this layer, data is further optimised for specific business needs and is often structured into de-normalised, read-optimised formats that are suitable for high-performance querying and reporting. The gold layer typically hosts data models that are directly used in business intelligence applications and decision support systems. Data becomes a true business asset in the gold layer, offering valuable and actionable insights.

Azure Databricks and Fabric utilise this architecture to enhance their data management and analytical offerings. Together, they create a robust environment in which data not only flows seamlessly through each stage of the medallion architecture, but is also enriched and made more accessible.

Lakehouse integration with Azure Databricks and Microsoft Fabric

The lakehouse-first pattern represents a transformative approach to data management and analytics. This approach is built on a tiered data storage system, organising data into bronze, silver and gold layers. This structured flow of data facilitates more efficient data processing, analytics and machine learning applications to transform raw data into data that’s optimised for businesses.

Azure Databricks excels in processing large volumes of data with its Spark-based analytics engine, effectively handling data transformations necessary for transitioning data from the bronze layer to the silver layer.

Fabric provides a cohesive analytics platform that integrates deeply with Azure Databricks. It offers sophisticated data management tools and helps connect and ingest data seamlessly from various sources through its extensive connector ecosystem.

This integration ensures that data moves freely through each layer of the medallion architecture, maintaining integrity and consistency while minimising complexity and overhead.

Foundation of open-source storage formats

Adopting Apache Parquet and Delta Lake enables OneLake and Azure Databricks to optimise Fabric engines and enhance interoperability across their platforms. This strategy ensures the robust handling of large datasets, facilitates seamless data access across the lakehouse architecture and reduces the complexities typically associated with managing large-scale data architectures:

- **Standardisation on Apache Parquet and Delta Lake**: OneLake adopts these formats for handling large datasets and support for transactional capabilities (ACID properties). This standardisation ensures that all data across the Fabric engines is optimised for both performance and compatibility, leading to more efficient data processing workflows.

- **Optimisation of Fabric engines for Apache Parquet and Delta Lake**: By redesigning data processing engines to be optimised for these formats, the system ensures high-performance data operations, which are crucial for processing large volumes of data efficiently.

- **Interoperability across the system**: The ability of Azure Databricks to read any Fabric artifact in OneLake highlights the interoperable nature of these technologies, ensuring that data can be seamlessly accessed and utilised across different parts of the lakehouse architecture.

The use of open-source storage formats within a lakehouse architecture maximises data utility, increases operational efficiency and reduces the complexity that's traditionally associated with managing large-scale data architectures. Apache Parquet and Delta Lake facilitate this by ensuring that data is stored in a robust and widely compatible format, making it easier for organisations to integrate and analyse data across diverse systems and platforms.

Integration of open-source formats with medallion architecture

The use of open-source formats such as Apache Parquet and Delta Lake standardises data storage and access within this integrated system, supporting advanced data management features such as ACID transactions and schema evolution. The powerful combination of processing capabilities of Azure Databricks and the management tools of Fabric within the medallion architecture enables businesses to approach data architecture in a scalable, efficient and highly conducive manner, facilitating the generation of transformative insights.

Azure Databricks excels in data processing and analytics, using Apache Spark to perform robust data transformations and analyses at scale. Its integration with ADLS Gen2 allows Azure Databricks to handle massive datasets efficiently, preparing data for further analytical processing. Fabric extends the capabilities of Azure Databricks by offering additional tools for data management, such as easy access to data sources through more than 200 native connectors and streamlined data ingestion mechanisms. This allows enterprises to implement a comprehensive data strategy that covers everything from ingestion to insightful analytics.

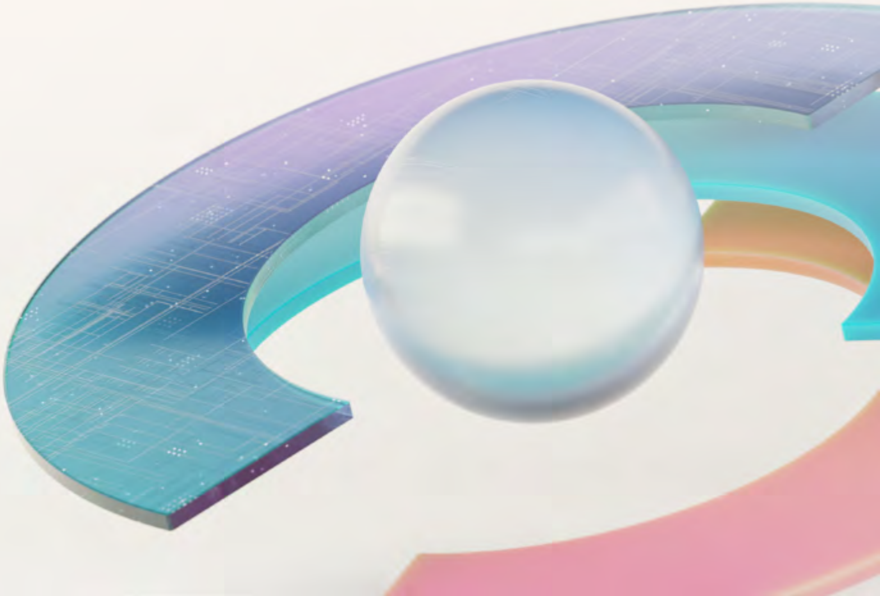
Within this architecture, OneLake helps centralise data management without necessitating physical data movement. Data stored in various locations can be accessed and analysed as if it were within a single repository. The federation capabilities of Azure Databricks further complement this by allowing queries across different data stores, thereby enhancing the flexibility and scope of data analytics.

The synergy between Azure Databricks and Fabric provides a robust foundation for building advanced lakehouse architectures. This combination simplifies data management across disparate data sources and enhances the analytical capabilities of organisations, enabling them to derive actionable insights more efficiently and with greater accuracy. This empowers enterprises with the ability to maximise the value of their data assets in a secure and scalable manner.

Lakehouse-first pattern scenarios

By adopting the medallion architecture with Azure Databricks and Fabric, integrating lakehouse data with Fabric workloads and utilising OneLake data with Lakehouse Federation, organisations can enable a sophisticated, tiered storage model within a lakehouse environment. This unified approach facilitates efficient data processing and management. It supports agile responses to data-driven insights and operational needs while enhancing the scalability and flexibility of data analytics across multiple storage systems:

- **Medallion architecture with Azure Databricks and Fabric**: This approach utilises a tiered storage model within a data lakehouse environment, facilitating efficient data processing and management. By using Azure Databricks in conjunction with Fabric, organisations can manage large-scale data analytics pipelines more effectively.
- **Integrating Azure Databricks lakehouse data with Fabric workloads**: In this scenario, data stored and managed in an Azure Databricks lakehouse can be directly used with the analytics tools in Fabric. This integration supports a more agile response to data-driven insights and operational needs.
- **Utilising OneLake data in Azure Databricks with Lakehouse Federation**: This set-up allows the utilisation of data across multiple storage systems within Azure Databricks. By federating data sources, users can query data across these sources as if they were a single entity, enhancing the flexibility and scalability of data analytics operations.



Use lakehouse data with Azure Databricks and Microsoft Fabric

The synergy between Azure Databricks and Microsoft Fabric offers organisations a powerful and efficient way to handle their data workloads. From ingestion and storage to analysis and reporting, enterprises benefit from a secure and governed framework. This flexibility empowers teams to select the platform that best fits their project needs, ensuring seamless integration within the broader enterprise ecosystem.

For example, a data science team that works primarily in notebook coding environments will appreciate the rich features of the Azure Databricks UI and the flexibility for managing advanced Spark libraries on the clusters, and AI engineers will appreciate the native ability to fine-tune models on their data. Business analysts may prefer the ease of use of the low-code dataflows in Fabric for quickly building pipelines that transform data and create new datasets in the gold layer of the lakehouse. Both teams can use their preferred tools to work on the same datasets without the need for either team to make copies of the data in their own environment.

Interacting with lakehouse data

Lakehouse data within the Azure ecosystem is typically stored in cloud locations, which can be categorised into two primary types:

1. **ADLS accounts**: ADLS is a cloud storage system optimised for analytics workloads. Enterprises can create and manage ADLS accounts to suit their data administration needs.

2. **OneLake**: OneLake is also an ADLS account, but unlike other accounts, Azure customers do not directly manage it. Instead, it is created as part of and administered by Fabric. It does not appear in the Azure portal, and although customers can interact with the data it contains, they do not have much control over the account itself.

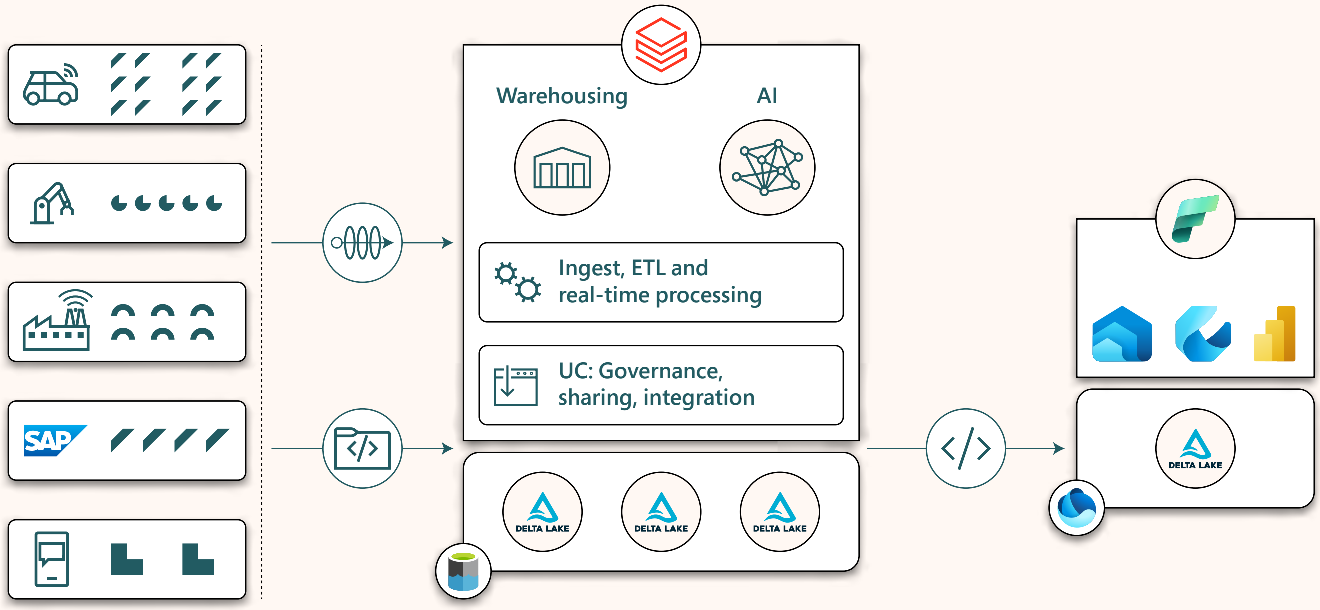

Warehousing
AI
Ingest, ETL and real-time processing
UC: Governance, sharing, integration
SAP
DELTA LAKE
DELTA LAKE
DELTA LAKE
DELTA LAKE

Figure 4: Linking ADLS accounts to OneLake with shortcuts

OneLake introduces shortcuts to help data professionals access data across OneLake and multiple ADLS accounts. Shortcuts allow data professionals to link data in their Unity Catalogue (with Azure Databricks shortcuts) or external ADLS accounts to OneLake, making them appear unified. Users can access data from these accounts seamlessly, without realising they're from different sources. Shortcuts help efficiently manage data by virtualising access to external data sources without unnecessary duplication, supporting scalable and efficient AI model training and deployment processes.

Integrate Azure Databricks with Power BI for enhanced data visualisation

For advanced visualisation and dashboarding scenarios with an Azure data lakehouse, most enterprises choose Microsoft Power BI as their tool of choice. This powerful visualisation and analytics tool is now offered as part of Fabric, allowing enterprises to fully integrate the administration and billing of Power BI with other Fabric resources.

Azure Databricks integrates seamlessly with Power BI. Databricks SQL warehouses and Unity Catalogue offer a flexible and scalable solution for Power BI in the lakehouse. Data that has been processed by Azure Databricks can be used in three ways:

1. **Azure Databricks Direct Publish for Power BI**: Databricks can now automatically sync tables, including relationships, to Power BI semantic models in a single click. This helps analysts build reports and dashboards faster than ever before.

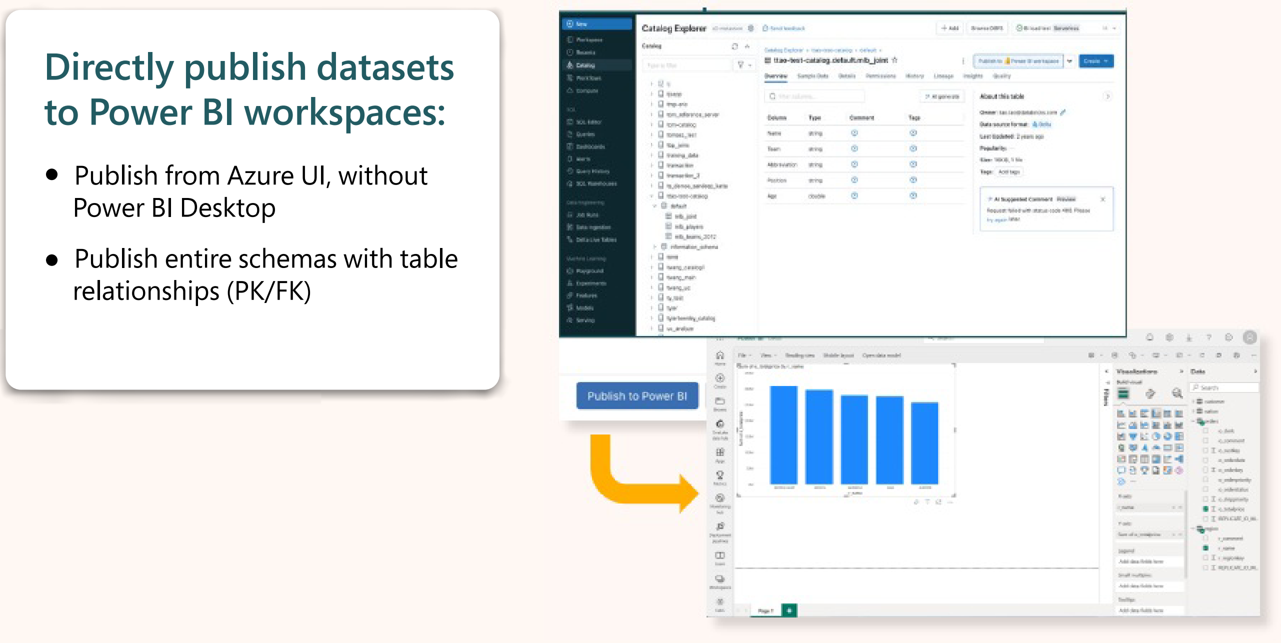
2. **Azure Databricks connector in Power BI Desktop**: Azure Databricks allows the Power BI client to connect to an Azure Databricks cluster, which can query and process the lakehouse data for you and send the results to Power BI for visualisation.

3. **Power BI Direct Lake mode**: Power BI can use its new Direct Lake mode to directly read Delta Lake data that has been written to an Azure storage location. This can be data that was written by Azure Databricks or by Fabric, and the storage location can be the OneLake account or any other ADLS account.

The previous section detailed how to use Azure Databricks to process raw data, prepare it for reporting and then write it to the lakehouse.

Direct Lake mode in Power BI to read and visualise Azure Databricks data

With OneLake storage, the files are stored in the efficient Delta Lake format. These Delta Lake tables have been optimised by the VertiPaq engine, making them highly efficient for consumption by Power BI. This enables Power BI to directly interact with the Delta Lake tables stored in OneLake without the need for an intermediary caching layer, such as Azure Analysis Services or Power BI datasets. This new mode of access, called Direct Lake mode, provides real-time data access without the need for refreshing models in Power BI.



Directly publish datasets to Power BI workspaces:

- Publish from Azure UI, without Power BI Desktop
- Publish entire schemas with table relationships (PK/FK)

Figure 5: Deep Power BI integration

The default dataset includes all tables from the lakehouse, allowing users to establish relationships and apply various modelling changes. These datasets from Unity Catalogue can be directly published to Power BI. Users can access and edit a published semantic model with the web modelling editor that's accessible through Power BI.

In the model view within the web modelling editor, you can see whether there's a Direct Lake connection by hovering the cursor over the table headers. Direct Lake also allows for the creation of new Power BI datasets directly through the web. This process ensures the use of Direct Lake for the connection. To learn more about using the web editor for semantic models, the following document will help you get started: Edit data models in the Power BI service (preview) – Power BI | Microsoft Learn.

Integrate Azure Databricks with Power BI to enhance data workflows

Integrating Azure Databricks with Power BI provides significant advantages for data management and visualisation, enhancing both security and performance in data analytics workflows:

1. First, the integration allows more secure and interactive data visualisation experiences directly from the data lake, avoiding the latency and costs associated with traditional data processing workflows. It uses Microsoft Entra ID for authentication, simplifying the user experience and increasing security, eliminating the need for personal access tokens. This integration ensures that security controls at the data lake level are enforced within Power BI, maintaining consistent security policies across platforms.[1]

2. Second, the semantic lakehouse architecture streamlines data ingestion and storage. It provides a unified storage layer that supports an extensive range of data formats and structures, significantly boosting the efficiency of data processing and transformation.

   This set-up not only simplifies the analytics stack, but also enhances data quality and accessibility for BI tools, enabling more sophisticated data modelling and analytics directly on large datasets.[2]

3. Last, the integration supports advanced analytics scenarios, simplifying the management and analysis of large data volumes. The DirectQuery option in Power BI plays a crucial role here, allowing users to perform real-time analysis without moving data out of the lakehouse. This capability is critical for maintaining up-to-the-minute accuracy in reports and dashboards, providing businesses with insights that are both deep and immediately actionable.[3]

These features collectively make Azure Databricks and Power BI a robust combination, offering businesses advanced tools to harness their data effectively and securely. To learn more about Power BI and Azure lakehouse integration, consult the following document: https://learn.microsoft.com/fabric/get-started/directlake-overview

[1] Power Up your BI with Microsoft Power BI and Lakehouse in Azure Databricks: Part 1 – Essentials – Microsoft Community Hub

[2] Power Up with Power BI and Lakehouse in Azure Databricks: Part 3 – Tuning Azure Databricks SQL – Microsoft Community Hub

[3] https://docs.databricks.com/partners/bi/power-bi.html

Use Data Activator to alert on changes in Azure Databricks data via Power BI

A common scenario for enterprises is that they want to receive alerts if particular metrics exceed certain thresholds. For example, they may want to know if there's a sudden, unexpected spike in sales for a particular item, or they may want to know if the volume of transactions has plummeted below the normal range, indicating a possible problem in the transaction pipeline.

These scenarios can be handled by a new feature in Fabric called Data Activator. This no-code tool monitors data in a Power BI report and automatically takes action if the data matches certain patterns or hits specified thresholds. When these events occur, Data Activator can take an action such as alerting a user or launching a Power Automate workflow.

In order to enable Data Activator, please follow the official documentation here: https://learn.microsoft.com/fabric/data-activator/

To create an alert with Data Activator when a freezer's temperature falls below 30° F in a Power BI report, follow these steps for monitoring freezer temperatures within a Fabric workspace:

1. Confirm that your Power BI report, which includes freezer temperature data, is published online to a Fabric workspace equipped with Premium capacity.

2. Choose the temperature visual:

    a. **Access the report**: Open the specific Power BI report that tracks freezer temperatures.

    b. **Select the relevant visual**: Find the visual that displays the freezer temperatures.

3. Click the ellipsis **(...)** in the top-right corner of the temperature visual and select **Set Alert** or use the **Set Alert** button found in the Power BI toolbar.

4. In the **Set Alert** pane, specify how you wish to receive alerts (email or Teams). If your visual includes multiple freezers (dimensions), use the **For each** dropdown to select the specific dimension (freezer) to monitor.

5. Define the alert condition, such as when the temperature drops below 30° F. Data Activator will monitor the temperature and notify you when this condition is met.

6. Decide where to save your Data Activator trigger in Power BI. You can add it to an existing reflex item or create a new one.

7. Click **Create alert** to finalise your Data Activator trigger. You can optionally deselect **Start my alert** if you prefer to edit the trigger in Data Activator before activating it.

By following these steps, you've successfully set up an alert in Data Activator to notify you when a monitored freezer's temperature falls below 30° F, allowing you to take immediate action if necessary. Once these data updates are complete, you should receive the alert from Data Activator that was configured.

Use Lakehouse Monitoring with Alerts to alert on changes in Azure Databricks

Enterprises often require alerts when data quality metrics exceed certain thresholds. For example, they may want to know if there's a sudden, unexpected spike in the number of missing values within a particular field, indicating a possible problem in the transaction pipeline, or if the quality of predictions from a machine learning model has declined, indicating a need to retrain the model on newer data.

These scenarios can be handled with an Azure Databricks feature called Lakehouse Monitoring with Alerts. This no-code tool monitors data quality in Unity Catalogue and automatically takes action if the data matches certain conditions or exceeds thresholds. When these events occur, Alerts will take a specified action, such as sending a notification via email, Slack or Teams. The alert can also call a webhook action, allowing users to build extensible, custom workflows based on changes in the data.

A monitor is a process that runs on a specified schedule to check the data quality of a particular table. When a user creates a monitor, it computes the data quality metrics for the table and stores the current values in a separate system table. Each time the monitor runs, it recomputes the quality metrics and compares them to the original values. If the quality has deteriorated, then an alert will be raised. For details on how a monitor can be created, consult the following document: https://docs.databricks.com/lakehouse-monitoring/create-monitor-ui.html

If a monitor detects that the quality of the data in the table has declined, it will raise the specified alert. This can be used to send a notification to the data engineering teams so they can investigate further. For details on how these alerts can be configured, check the following document: https://docs.databricks.com/lakehouse-monitoring/monitor-alerts.html

Better together: Azure Databricks, Unity Catalogue and Microsoft Fabric Purview

As the demand for analytics grows and data platforms evolve into more intricate systems, governing the platform – management of data availability, usability, integrity and security – becomes paramount. In a data lakehouse architecture, data governance helps ensure that data is properly catalogued, classified and managed. By implementing effective data governance, organisations can manage their data properly and use it to drive business value.

Effective data governance in a data lakehouse architecture requires the implementation of policies, procedures and standards for managing data. This includes defining data ownership and stewardship, establishing data quality standards and implementing data security and compliance measures. To provide these crucial data governance capabilities, both Azure Databricks and Microsoft Fabric offer powerful, modern features.

Unity Catalogue in Azure Databricks

Azure Databricks includes Unity Catalogue, which provides centralised fine-grained access control for an organisation's data storage locations, auditing of data access and lineage tracking from ingestion to all data workloads and Azure Databricks provides column-level and row-level access controls and data discovery tools. It now also includes system tables, which provide a straightforward way to query audit data, billing data and lineage. Additionally, Unity Catalogue is supported by AI capabilities to automatically document tables and columns, facilitate semantic search and help surface related data products.

Microsoft Purview to govern Microsoft Fabric

Fabric integrates with Microsoft Purview for data governance, information protection and data loss prevention. The information protection features enable enterprises to discover, classify and protect the data stored in the lakehouse and apply sensitivity labels to it. Data loss prevention uses policies to detect when sensitive data is uploaded to Power BI semantic models or other supported Fabric assets. It can also help detect common sensitive data. Fabric also includes tools for discovering data lineage so that data can be tracked through the analytical process as it moves from its original source, through the various transformations and into the various reporting models.

Microsoft Purview and Unity Catalogue to streamline data governance

Microsoft Purview and Unity Catalogue are two powerful tools designed to enhance data governance and management within cloud environments, particularly for the users of the extensive cloud services offered by Microsoft.

The broad governance capabilities of Microsoft Purview can extend into the Azure Databricks environment, where Unity Catalogue applies specific governance and security measures to Azure Databricks workspaces. This integration allows organisations to maintain a consistent governance strategy across all platforms, enhancing security and operational efficiency. Organisations can ensure that data policies are uniformly applied, data lineage is clear and auditable and all regulatory compliance requirements are met across their entire data estate.

Unity Catalogue offers a sophisticated and centralised governance solution for managing a variety of data assets within the Azure Databricks lakehouse platform. It integrates seamlessly with Azure to provide fine-grained governance capabilities, including access control, auditing and data lineage. Unity Catalogue simplifies data management across multiple Azure Databricks workspaces, enabling organisations to enforce consistent security and compliance policies across their data assets, whether they are files, tables or machine learning models.

Unity Catalogue provides a single point of control for data access policies, which apply uniformly across all workspaces. This ensures that data governance is not only centralised, but also deeply integrated into the Azure Databricks environment, enhancing security and governance. Additionally, Unity Catalogue supports comprehensive data discovery, making it easier for users to find and access the data they need while adhering to the defined access controls and policies. This unified approach helps streamline operations and reduces the complexity typically associated with managing large and diverse data environments.

Microsoft Purview allows enterprises to maintain control over their data through Fabric, enabling seamless integration and management of data from various sources down to detailed reports. Along with a suite of tools to protect sensitive data across different environments, Microsoft Purview provides capabilities such as sensitive data discovery, classification and protection using sensitivity labels. It also facilitates comprehensive auditing and data loss prevention strategies specifically tailored for complex environments such as Power BI semantic models.

Best practices

The integration of Azure Databricks with Microsoft Purview focuses on maximising data governance and security within Azure Databricks environments. Key best practices for this integration include:

- **Secure access to critical data**: Microsoft Purview can be used to automatically discover and classify data within Azure Databricks, visualise data lineage and manage access controls effectively. This ensures that only approved personnel can access sensitive or critical data and that all data policies are consistently applied across Azure services.

- **Use two separate connectors to manage metadata**: Microsoft Purview offers two separate connectors for Azure Databricks. Most enterprises will use the Azure Databricks Unity Catalogue connector because Unity Catalogue enables many of the modern features in Azure Databricks. However, for customers who have not yet migrated to Unity Catalogue and are still using Hive to manage their metadata, Microsoft Purview also has an Azure Databricks Hive Metastore connector that can be used.

- **Utilise custom rule sets**: Enterprises can use Microsoft Purview to scan catalogues, schemas, tables and views. As a best practice, enterprises should use custom rule sets in addition to the rule sets provided by Microsoft Purview. Creating a custom rule set for different regions of the world can speed up the scanning process by using only the classification rules required in a particular region.

- **Indicate data sensitivity with labelling tools**: Labelling tools in Microsoft Purview can be used on Unity Catalogue data to indicate the sensitivity of files and data columns. These labels travel with the data and can be used by other tools in the Microsoft data ecosystem, such as SharePoint and Power BI, to automatically apply data handling policies.

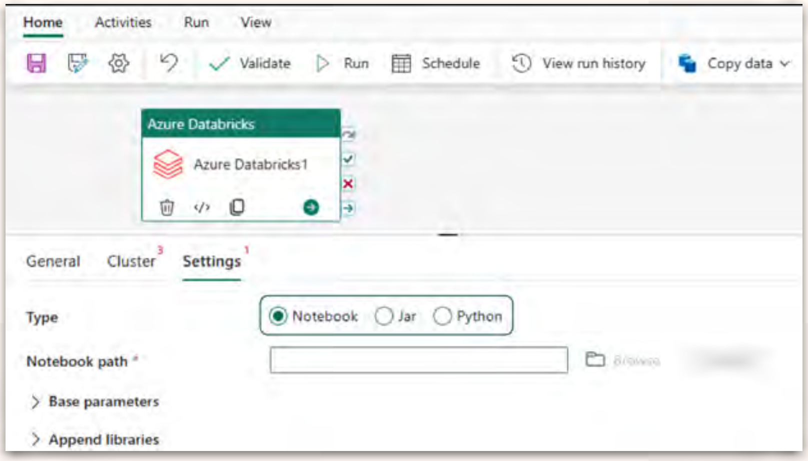
The combined capabilities of Microsoft Purview, Azure security in OneLake and Unity Catalogue support a resilient and agile data governance strategy, enabling businesses to use their data assets effectively in a digital landscape.

Data Factory and Azure Databricks activity in Microsoft Fabric

Azure Databricks activity in Microsoft Fabric represents a significant evolution in data processing within cloud environments, integrating the extensive capabilities of Azure Data Factory into a more unified and robust framework. With the new Azure Databricks activity, users can easily create and manage data pipelines in Fabric, incorporating sophisticated analytics and processing tasks directly into their workflows.

Users can configure Azure Databricks clusters that are used for data processing directly within Fabric, like the functionality offered in Azure Data Factory. This includes the ability to set up Azure spot instances for accessing unused Azure compute capacity at reduced costs and specifying cluster policies to ensure that cluster configurations meet organisational standards and requirements.

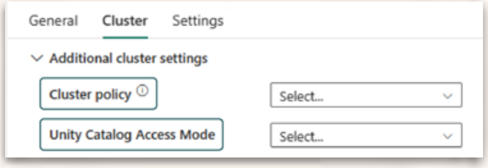
One activity that encompasses all three job types: **Notebook, Jar, Python**



Type ● Notebook ○ Jar ○ Python

Notebook path *

Unity Catalog support and **Policy ID** integration



Cluster policy Select...

Unity Catalog Access Mode Select...

Figure 6: Seamless integration between Azure Databricks and Data Factory with Microsoft Fabric

Moreover, the enhanced Azure Databricks integration introduces several new features. They are:

- **Configure Unity Catalogue access mode**: Users can configure the access mode of Unity Catalogue, which enhances governance and security by managing permissions more meticulously.
- **Run multiple tasks in a single Databricks activity**: Users can run various tasks, such as notebooks, JARs and Python scripts, within a single Databricks activity, streamlining the process and reducing the complexity previously associated with managing multiple types of data jobs.
- **Monitor notebook runs in real time**: During operation, users can initiate a data pipeline and immediately monitor its execution, with the system providing direct links to the outputs in the Azure Databricks instance. This real-time monitoring capability allows users to track the details of the notebook runs, including cluster performance and computational efficiency, directly from Fabric.

This integration not only brings existing Azure Databricks capabilities from Azure Data Factory into Fabric but also introduces new functionalities such as cluster policy and Unity Catalogue support, enhancing the overall data management and analytics experience.

Enhance organisational capabilities with generative AI

Advanced AI models have changed the technology landscape. Enterprises are trying to unlock the potential of their data and use AI to expand their business capabilities. This includes the use of generative AI to build their data lakehouse on Azure.

The next section will detail how to describe your desired architecture to ChatGPT and have it generate code that can be implemented in Azure.

Generate architecture and code with Azure OpenAI

Using prompt engineering, you can generate architecture plans and code using the same process that would be used to summarise a technical article.

Prompt Input

```
I want you to act as a data engineer. I will provide details about
a scenario where a customer needs to utilise data stored in ADLS Gen2.
It will be accessed by Azure Databricks and Microsoft Fabric to load
data for machine learning models. It will be your job to come up with
an architecture to address this need.
```

Answer from Azure OpenAI

```
To address the scenario where a customer needs to utilise data
stored in Azure Data Lake Storage Gen2 (ADLS Gen2), accessed by Azure
Databricks and Microsoft Fabric for loading data into machine learning
models, we'll design an architecture that leverages these services
efficiently. This architecture aims to streamline data processing,
enhance data governance and ensure scalable machine learning
operations.
```

```
Architecture Overview

  • Data storage layer: Azure Data Lake Storage Gen2: Acts as the
    central repository for storing raw data, processed data and
    machine learning artefacts. ADLS Gen2 is optimised for large-scale
    analytics scenarios and supports hierarchical namespace, which
    simplifies data management.

  • Data processing and analytics layer: Azure Databricks: Serves
    as the primary platform for data processing, analysis and
    machine learning model development. Azure Databricks provides a
    collaborative environment with support for multiple programming
    languages and integrates seamlessly with ADLS Gen2 for direct
    access to data.

  • Data management and governance layer: Microsoft Fabric: Utilised
    for managing and governing the data across the entire data
    landscape. Microsoft Fabric allows creating a unified data
    governance model that encompasses data in ADLS Gen2 and data
    processed or generated by Azure Databricks.

  • Machine learning operationalisation: Azure Machine Learning:
    Integrates with Azure Databricks for deploying and managing machine
    learning models. This service provides tools for model versioning,
    monitoring and serving, facilitating the operationalisation of
    machine learning models.


Detailed Workflow

1.Data ingestion: Data is ingested into ADLS Gen2 from various sources,
  including structured databases, IoT devices, log files, etc. This
  data is stored in raw format within a hierarchical file system
  structure.
```

> **Note**
>
> The rest of the content generated by Azure OpenAI can be found in the *Appendix*.
>
> Prompts and answers reflected here may not reflect your exact experience.

Generate and correct code with Databricks Assistant

Databricks Assistant is an AI-powered tool integrated within the Azure Databricks environment, designed to enhance coding productivity by assisting users with code generation, error resolution and documentation directly within Databricks notebooks. It uses the capabilities of Azure AI services, enhancing the way developers interact with data and code within the platform.

Databricks Assistant acts as a robust tool that supports developers by automating routine tasks, optimising code, explaining functionalities and troubleshooting, all within the Azure Databricks workspace. This not only speeds up the development process but also helps in maintaining a high standard of code quality and documentation, making it an asset in data engineering and analysis workflows.

Databricks Assistant enhances productivity in data science and engineering by utilising AI to help with code generation, error resolution and documentation within the Azure Databricks environment in the following ways:

- **Code generation**: Databricks Assistant simplifies coding by allowing users to input their requirements in natural language. It can generate executable SQL queries or transform code from one language to another, such as converting Python pandas code to PySpark. This feature speeds up development and reduces manual coding errors.

- **Error resolution**: Databricks Assistant can quickly identify and clarify coding errors, offering solutions by generating corrective code snippets. This is valuable for both novice and experienced programmers, as it provides immediate solutions to common syntax and runtime issues, thus minimising downtime.

- **Documentation**: It assists in documenting code by automatically generating comments that explain the functionality of code blocks, supporting the maintenance of clean and understandable codebases essential for long-term project sustainability and team collaboration.

- **Contextual help and learning**: Databricks Assistant aids in context by understanding the user's environment, including frequently used tables, schema and previous queries. It uses this context to provide precise answers and tailored code snippets, enhancing project-specific support.

- **Visualisations and dashboards**: In visualisation environments such as Lakeview, Databricks Assistant can generate data visualisations based on user prompts, allowing the quick creation and iteration of visual displays without needing deep technical expertise.

- **Data ingestion and ETL processes**: Databricks Assistant streamlines data ingestion and ETL tasks by automating code generation and optimisation, expediting the set-up and execution of data pipelines.

- **Security and compliance**: Azure Databricks ensures that interactions with Databricks Assistant are secure, adhering to user permissions and data governance policies, which makes it suitable for use in sensitive and regulated environments.

- **Integration and accessibility**: Databricks Assistant is accessible from notebooks, SQL editors and file editors, making it a versatile tool for various data tasks.

- **Feedback and iteration**: Users can provide feedback directly through the platform, which helps improve Databricks Assistant's accuracy and functionality. This ensures that the tool evolves in response to user needs and challenges.

Azure Databricks workspaces can be provisioned with a free 14-day trial that includes access to Assistant, allowing potential users to evaluate its capabilities and integrate it into their workflows without an initial investment.

Here are some examples of how Databricks Assistant can be used in different scenarios to enhance productivity and simplify tasks:

1. **Code generation**

   Scenario: A user needs to extract and summarise sales data by region from a DataFrame.

   User input:

```
Generate a SQL query to sum
all sales by region from the
sales_data table.
```

   Response:

```
sqlCopy code

SELECT region, SUM(sales) AS
total_sales FROM sales_data
GROUP BY region;
```

   Utility: This helps the user quickly get the desired code without having to manually write the SQL query.

2. **Fixing errors**

   Scenario: A user has written PySpark code that fails due to a syntax error.

   User input:

```
This code isn't working.
Can you help fix it?
```

   Diagnosis and fix: Databricks Assistant identifies a missing comma in a DataFrame operation and suggests the corrected code with a change highlighted.

   Utility: The user gets immediate feedback and correction suggestions, speeding up troubleshooting and reducing frustration.

3. **Code documentation**

   Scenario: A developer wants to add comments to a complex function to improve code readability.

   User input:

```
Can you document this
function explaining
each step?
```

   Response: Databricks Assistant adds comments before each significant line or block of code explaining what it does, such as initialising variables, error handling and logic flows.

   Utility: Ensures that the code is understandable for future reference or for other team members, enhancing maintainability.

These examples illustrate the practical benefits of Databricks Assistant in real-world development environments, streamlining the coding process, simplifying error resolution and ensuring thorough documentation.

> **Note**
> Prompts and answers reflected here may not reflect your exact experience.

Explore real-world use cases with hands-on examples

In earlier examples, you used Python code to read the data and aggregate it to answer some business questions. This section looks at an alternative to Python code to read data and how AI can be used to allow business users to query lakehouse data using English instead of a query language.

Use the English SDK for Spark to write queries in Azure Databricks and Fabric

To utilise the English SDK for Apache Spark, the following requirements should be met:

> **Note**
>
> Azure Databricks recommends using GPT-4.

1. **Install the English SDK package**: Begin by adding the SDK to your environment. Use the `%pip install pyspark-ai --upgrade` command in your notebook to ensure you have the latest version.

2. **Restart the Python kernel**: After installation, you need to restart the Python kernel to apply the updates. Execute `dbutils.library.restartPython()` in a new cell to reset the environment.

3. **Set the OpenAI API key**: Your API key from OpenAI is necessary for authentication. Implement it by setting an environment variable with the `os.environ['OPENAI_API_KEY'] = '<your-openai-api-key>'` Python code, replacing `<your-openai-api-key>` with your actual API key.

4. **Activate the SDK**: To use the SDK, activate it within your notebook. This involves initialising the SDK with your preferred language model (such as GPT-4) and then activating it to start interpreting English queries.

5. **Create a DataFrame**: Use SQL queries within the notebook to fetch data from your Azure Databricks workspace and save it as a DataFrame. This DataFrame will be the basis for your English queries.

6. **Query using English**: Finally, query the DataFrame by asking questions in plain English. The SDK interprets these questions and executes the corresponding SQL queries, returning the results directly to your notebook.

An example query using English with the English SDK for Apache Spark could be something such as:

```
What was the average trip
distance for each day during
the month of January 2016?
Print the averages to the
nearest tenth.
```

This query demonstrates how plain English can be utilised to conduct data analysis activities, such as calculating averages from a dataset, with the English SDK, allowing Apache Spark to interpret and execute English-language instructions.

Another example query using English for the English SDK for Apache Spark could be:

```
Show me the total revenue for
each product category in the
last quarter.
```

This type of query illustrates how users can request specific financial metrics, such as total revenue, broken down by categories, over a defined period, such as the last quarter, using natural language. This approach simplifies complex data analysis tasks into straightforward English questions.

Creating a notebook in Microsoft Fabric

Fabric notebooks are a key tool for crafting Apache Spark jobs and conducting machine learning experiments. With support for advanced visualisations and Markdown text integration, it offers a web-based interactive platform that's popular among data scientists and engineers for coding. Data scientists rely on these notebooks to develop and deploy machine learning models, including experimentation, model tracking and deployment phases. Fabric notebooks offer:

- Immediate usability with no set-up required
- An intuitive, low-code interface for data exploration and processing
- Enhanced data security through integrated enterprise-level features
- The ability to analyse data in various formats (including CSV, TXT, JSON, Parquet and Delta Lake) with the robust capabilities of Spark

Creating notebooks

When creating a notebook, users have two options: create a new one or import an existing one. Organisations can create a new notebook by following the familiar Fabric item creation workflow:

1. Initiate a new notebook directly from the Fabric **Data Engineering** or the **Data Science** home page, or throughout the workspace **New** option.
2. Choose **Import Notebook** in the same window to import an existing notebook, such as an Azure Databricks notebook file.
3. Once you have a notebook open, you can add code to it to write data to OneLake.

Working with data in OneLake is straightforward and does not involve a complex set-up to access the data.

Loading data into OneLake via a Microsoft Fabric data engineering notebook

```
from pyspark.sql import SparkSession

# Initialize Spark session (assuming it's not already initialized)

spark = SparkSession.builder.appName("ParkDataImport").getOrCreate()

# URL to the CSV file

data_url = "https://www.dropbox.com/s/268uogek0mcypn9/park-data.
csv?raw=1"

# Read the CSV data directly into a Spark DataFrame

df = spark.read.option("header", "true").csv(data_url)

# Assuming csv_table_name, parquet_table_name, and delta_table_name are
defined elsewhere in your code

csv_table_name = "park_data_csv"

parquet_table_name = "park_data_parquet"

delta_table_name = "park_data_delta"
```

```
# Save dataframe as CSV files to Files section of the default Lakehouse

df.write.mode("overwrite").format("csv").save("Files/" + csv_table_name)

# Save dataframe as Parquet files to Files section of the default
Lakehouse

df.write.mode("overwrite").format("parquet").save("Files/" + parquet_
table_name)

# Save dataframe as a delta lake, parquet table to Tables section of the
default Lakehouse

df.write.mode("overwrite").format("delta").saveAsTable(delta_table_name)

# Save the dataframe as a delta lake, appending the data to an existing
table

# Make sure the table exists and the schema matches to avoid errors

df.write.mode("append").format("delta").saveAsTable(delta_table_name)
```

Reading and data analysis

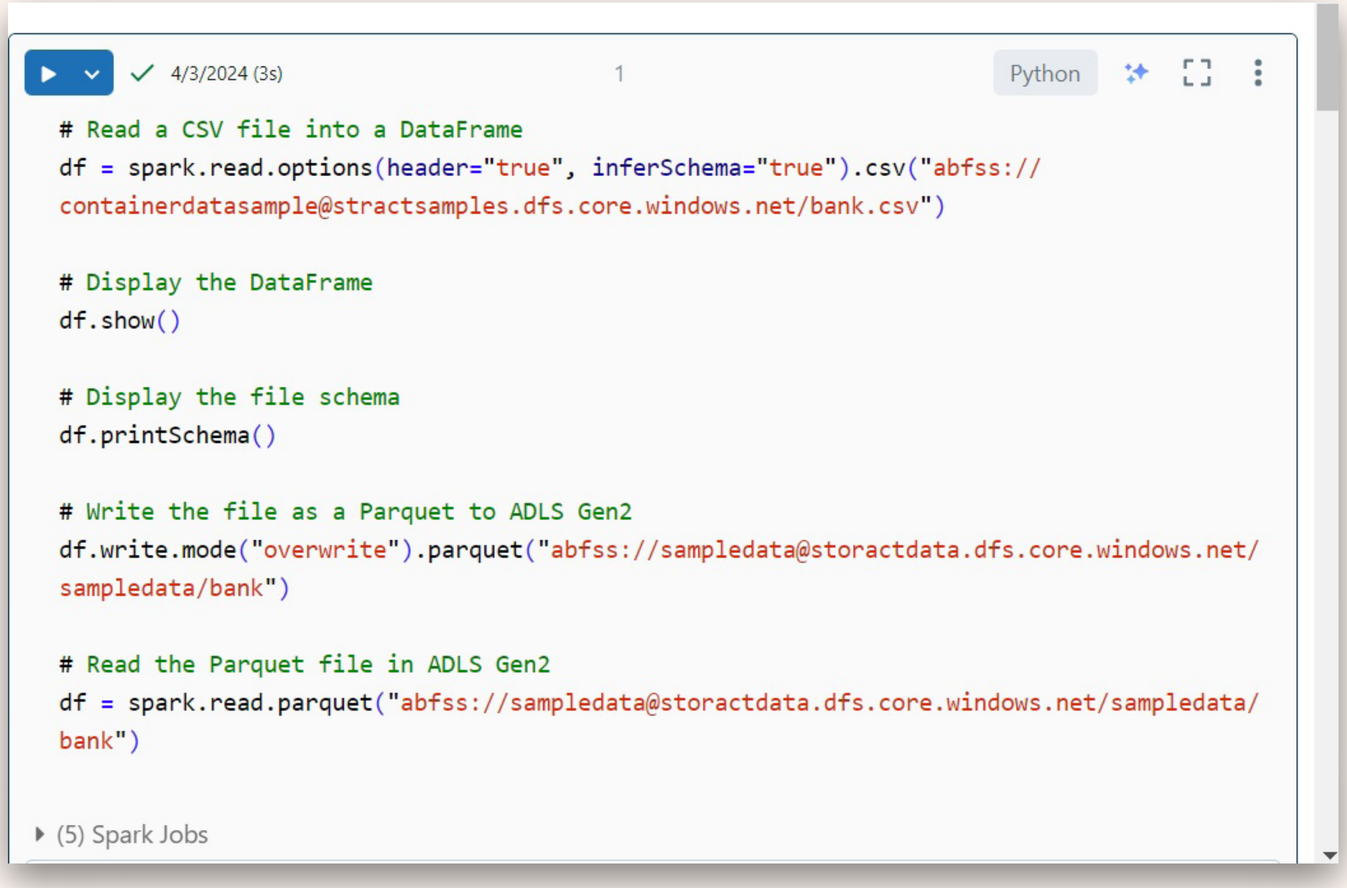
Once the data has been successfully uploaded, try reading and analysing the data:

```
# Basic Data Analysis

# Count of animal sightings by type (excluding squirrels)

animal_sightings = spark.sql("""

SELECT Animal_Type, COUNT(*) as Total_Sightings

FROM park_data_view

WHERE Animal_Type != 'Squirrel'

GROUP BY Animal_Type

ORDER BY Total_Sightings DESC
""")
animal_sightings.show()
# Average temperature and most common weather conditions
avg_temp = spark.sql("""
SELECT AVG(Temperature) as Average_Temperature
FROM park_data_view
""")
avg_temp.show()
common_weather = spark.sql("""
SELECT Weather, COUNT(*) as Frequency
FROM park_data_view
GROUP BY Weather
ORDER BY Frequency DESC
LIMIT 5
""")
common_weather.show()
# Total count of squirrel sightings
squirrel_sightings = spark.sql("""
SELECT COUNT(*) as Total_Squirrel_Sightings
FROM park_data_view
WHERE Animal_Type = 'Squirrel'
""")
squirrel_sightings.show()
```

Creating and modifying a Delta table from Parquet in Azure Databricks with changes reflected in Fabric

Azure Databricks and Fabric provide a data lakehouse environment that allows businesses to access and analyse their data simultaneously, using different tools. This supports a wide range of data processing activities on the same set of data, making it easier for organisations to manage and derive insights from their information efficiently.

1. Open your Azure Databricks workspace in a browser of your choice and launch a new Azure Databricks notebook.



```
# Read a CSV file into a DataFrame
df = spark.read.options(header="true", inferSchema="true").csv("abfss://
containerdatasample@stractsamples.dfs.core.windows.net/bank.csv")

# Display the DataFrame
df.show()

# Display the file schema
df.printSchema()

# Write the file as a Parquet to ADLS Gen2
df.write.mode("overwrite").parquet("abfss://sampledata@storactdata.dfs.core.windows.net/
sampledata/bank")

# Read the Parquet file in ADLS Gen2
df = spark.read.parquet("abfss://sampledata@storactdata.dfs.core.windows.net/sampledata/
bank")
```

Figure 7: Example notebook

2. Copy and paste the following script into your new notebook. Then, execute the following Python script in your notebook to create a Delta table within your ADLS Gen2 account. This script reads some sample Parquet data and then writes it as a Delta table into your ADLS account:

```
#python

# Adjust the file path to point to your sample parquet data using the
following format:

"abfss://<storage name>@<container name>.dfs.core.windows.
net/<filepath>"

# The line below reads Parquet files from your ADLS account

df = spark.read.format('Parquet').load("abfss://datasetsv1@olsdemo.dfs.
core.windows.net/demo/full/dimension_city/")

#This line writes the read data as Delta tables back into your ADLS
account

df.write.mode("overwrite").format("delta").save("abfss://datasetsv1@
olsdemo.dfs.core.windows.net/demo/adb_dim_city_delta/")
```

And, of course, Azure Databricks can also read the data in the ADLS account.

3. Azure Databricks can also modify the same sets of data that were originally created previously with Fabric. To see this in action, append some new rows to the Delta Lake tables you created in OneLake:

```
# Import the necessary libraries

from pyspark.sql import SparkSession

from pyspark.sql.functions import lit

# Initialize a Spark session

spark = SparkSession.builder.appName("AppendToDeltaTable").
getOrCreate()
```

```
# Define the path to your Delta Lake table in OneLake

# Replace '<your-delta-table-path>' with the actual path to your Delta
Lake table

delta_table_path = "abfss://<container-name>@<storage-account-name>.
dfs.core.windows.net/<your-delta-table-path>"

# Create a DataFrame with the new rows you want to append

# Replace the column names and values with those relevant to your table

new_rows = [

    ("NewValue1", 10),

    ("NewValue2", 20)

    # Add as many rows as needed

]

# Define the schema based on your Delta Lake table structure

# This is an example schema; adjust it to match your table's columns
and data types

schema = ["ColumnName1", "ColumnName2"]

# Create a DataFrame with the new data

new_data_df = spark.createDataFrame(new_rows, schema)

# Append the new data to the Delta Lake table

# Ensure the table format is set to 'delta' for Delta Lake
compatibility

new_data_df.write.format("delta").mode("append").save(delta_table_path)

# Verify by reading back the data from the Delta Lake table

df = spark.read.format("delta").load(delta_table_path)

df.show()
```

As the examples illustrate, a data lakehouse, built on any platform with the advantages of open platforms, enables enterprises to use a variety of engines to work on the same copy of the data at the same time.

Azure Databricks connector within Power BI

The Power BI connector for Azure Databricks provides seamless integration between Power BI and Azure Databricks, enabling organisations to connect, analyse and visualise data stored in Azure Databricks with ease. This integration supports Microsoft Entra ID authentication, removing the need for administrators to generate personal access tokens for connection. It is designed to enhance data connectivity and analysis experiences, allowing for efficient and secure data visualisation directly from the data lake.

1. Obtain your Azure Databricks server hostname and HTTP path for setting up the connection in Power BI.

2. Launch Power BI Desktop.

3. Choose **Get Data** from the home screen or navigate through **File > Get Data**.

4. Search for **Azure Databricks**.

5. Select **Azure Databricks connector** and then click **Connect**.

6. Input the server hostname and HTTP path you obtained earlier.

7. Decide between the **Import** and **DirectQuery** modes for your data connectivity. For more insights into these options, consider reading about the use of DirectQuery in Power BI Desktop.

8. Select your preferred authentication method:

   a. **Personal Access Token**: Enter your Azure Databricks personal access token.

   b. **Microsoft Entra ID**: Choose **Sign in** and follow the prompts.

   c. **Username/Password**: This option is typically not applicable.

9. After authentication, Power BI will present you with the **Navigator** window. Here, you can select the Azure Databricks data you wish to query. If your workspace has Unity Catalogue enabled, you'll first select a catalogue, followed by a schema and a table.

For workloads that need the processing power and flexibility offered by Azure Databricks, enterprises can use the advanced visualisation capabilities of Power BI along with Azure Databricks.

Achieve excellence with Azure Databricks and Microsoft Fabric

The integration of Azure Databricks and Microsoft Fabric represents a transformative approach to managing and analysing data within modern cloud environments. Azure Databricks provides a high-performance platform for data processing and AI-driven analytics, while Fabric enhances these capabilities with robust data management tools. This combination allows organisations to harness advanced analytics and AI solutions more effectively.

Together, Azure Databricks and Fabric streamline analytics workloads by providing seamless data access without the need for redundant data copies. This integration supports direct queries from Power BI, leading to improved performance and a simplified data architecture.

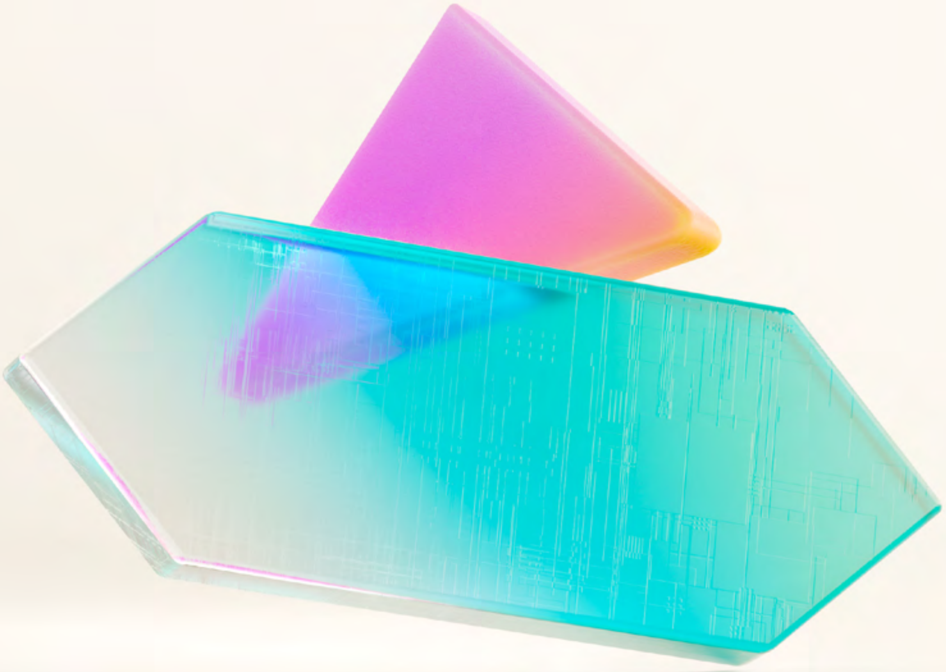
The synergy between Azure Databricks and Fabric unlocks the full potential of the lakehouse architecture, enhancing the ability to manage, process and analyse large datasets more effectively. This enables organisations to deploy AI solutions with ease and tackle complex analytical challenges.

Furthermore, the combined capabilities of Azure Databricks and Fabric address the entire lifecycle of AI and machine learning projects, from data ingestion and storage to analysis and deployment. This integration helps organisations maintain control over their data processes and ensure that AI initiatives are both scalable and compliant with industry regulations.

Organisations can enhance their data governance, boost operational efficiency and foster innovation in AI and analytics with the powerful integration of Azure Databricks and Fabric, making them well equipped in the face of a data-driven economy and enabling them to achieve their strategic goals.

Next steps

- Talk to a sales specialist.
- Get more value from your data with Microsoft Fabric.
- Unlock insights from your data and build AI solutions with Azure Databricks.
- Watch the webinar to learn more about how Azure Databricks and Microsoft Fabric are better together.



Appendix

Prompt input

I want you to act as a data engineer. I will provide details about a scenario where a customer needs to utilise data stored in ADLS Gen2. It will be accessed by Azure Databricks and Microsoft Fabric to load data for machine learning models. It will be your job to come up with an architecture to address this need.

Answer from Azure OpenAI

To address the scenario where a customer needs to utilise data stored in Azure Data Lake Storage Gen2 (ADLS Gen2), accessed by Azure Databricks and Microsoft Fabric for loading data into machine learning models, we’ll design an architecture that leverages these services efficiently. This architecture aims to streamline data processing, enhance data governance and ensure scalable machine learning operations.

Architecture Overview

- Data storage layer: Azure Data Lake Storage Gen2: Acts as the central repository for storing raw data, processed data and machine learning artefacts. ADLS Gen2 is optimised for large-scale analytics scenarios and supports hierarchical namespace, which simplifies data management.
- Data processing and analytics layer: Azure Databricks: Serves as the primary platform for data processing, analysis and machine learning model development. Azure Databricks provides a collaborative environment with support for multiple programming languages and integrates seamlessly with ADLS Gen2 for direct access to data.
- Data management and governance layer: Microsoft Fabric: Utilised for managing and governing the data across the entire data landscape. Microsoft Fabric allows creating a unified data governance model that encompasses data in ADLS Gen2 and data processed or generated by Azure Databricks.

- Machine learning operationalisation: Azure Machine Learning: Integrates with Azure Databricks for deploying and managing machine learning models. This service provides tools for model versioning, monitoring and serving, facilitating the operationalisation of machine learning models.

Detailed Workflow

1. Data ingestion: Data is ingested into ADLS Gen2 from various sources, including structured databases, IoT devices, log files, etc. This data is stored in raw format within a hierarchical file system structure.

2. Data processing and exploration:

   - Azure Databricks notebooks are used to access raw data stored in ADLS Gen2. Data engineers and scientists process and explore this data using Spark, performing tasks like data cleaning, transformation and feature engineering.

   - Processed data can be stored back into ADLS Gen2 in an optimised format (e.g., Delta Lake) for further analysis or machine learning.

3. Data governance and lineage:

   - Microsoft Fabric is configured to catalogue data assets in ADLS Gen2 and data products generated by Azure Databricks. It provides a unified view of the data, enforcing policies, data quality rules and security measures.

   - Microsoft Fabric enables setting up data lineage and tracking to maintain visibility into data transformations and usage throughout the data lifecycle.

4. Machine learning model training and deployment:

   - Data scientists use Azure Databricks for developing and training machine learning models, leveraging the integrated MLflow for experiment tracking and model management.

   - Trained models are registered in Azure Machine Learning, which is used to deploy these models to production as web services or containers, enabling real-time or batch scoring.

5.Monitoring and management:

- Azure Machine Learning provides model monitoring capabilities, tracking model performance and data drift over time. It facilitates retraining workflows when models become stale.

- Microsoft Fabric oversees the governance and compliance aspect, ensuring that data access and model deployments adhere to organisational and regulatory standards.

This architecture provides a comprehensive solution for leveraging ADLS Gen2 data in Azure Databricks and Microsoft Fabric, facilitating efficient machine learning model development, governance and operationalisation. It ensures scalability, security and compliance, catering to the dynamic needs of modern data-driven organisations.

> **Note**
>
> Prompts and answers reflected here may not reflect your exact experience.

Prompt input

Based on the above, can you provide detailed build instructions with code?

Answer from Azure OpenAI

Creating a detailed build guide with code for integrating Azure Data Lake Storage Gen2 (ADLS Gen2), Azure Databricks and Microsoft Fabric into a cohesive architecture for machine learning requires several steps. Below, we'll outline key phases in this process, including set-up, data processing and machine learning model lifecycle management. This guide assumes a basic understanding of Azure services and a subscription where you can deploy these services.

1. Set-up and Configuration

Azure Data Lake Storage Gen2

1. Create an ADLS Gen2 Storage Account:
   - Navigate to the Azure Portal.
   - Create a new storage account, selecting your subscription, resource group and the ‘StorageV2’ account kind.
   - Enable the hierarchical namespace to use ADLS Gen2 features.
2. Configure Storage Account Access:
   - Within the storage account, go to the ‘Access keys’ section and note down the connection string for later use.
   - Alternatively, set up Azure Active Directory (Microsoft Entra ID) authentication for more secure access.

Azure Databricks

1. Create an Azure Databricks Workspace:
   - In the Azure Portal, create a new Azure Databricks service.
   - Once the deployment is complete, navigate to the Azure Databricks workspace and launch the workspace.
2. Create a Cluster in Azure Databricks:
   - Inside the Azure Databricks workspace, go to ‘Compute’.
   - Create a new cluster, ensuring it’s compatible with the libraries you plan to use.
3. Mount ADLS Gen2 to Azure Databricks (for direct access to data):
   - In a Azure Databricks notebook, use the following code to mount your ADLS Gen2 storage for direct access:

Python

```
configs = { "fs.azure.account.auth.type": "OAuth", "fs.azure.
account.oauth.provider.type": "org.apache.hadoop.fs.azurebfs.
oauth2.ClientCredsTokenProvider", "fs.azure.account.oauth2.
client.id": "<Application-ID>", "fs.azure.account.oauth2.client.
secret": dbutils.secrets.get(scope="<scope-name>", key="<secret-
key>"), "fs.azure.account.oauth2.client.endpoint": "https://
login.microsoftonline.com/<Directory-ID>/oauth2/token" } dbutils.
fs.mount( source="abfss://<filesystem>@<storage-account-name>.
dfs.core.windows.net/", mount_point="/mnt/<mount-name>", extra_
configs=configs)
```

- Replace placeholders with your actual Microsoft Entra ID application registration details and ADLS Gen2 storage details.

Microsoft Fabric

1. Setting Up Microsoft Fabric:

   - Currently, Microsoft Fabric is a conceptual integration point and may not have direct customer-facing set-up instructions available publicly. Typically, set-up would involve configuring your environment to catalogue and manage data assets across Azure Databricks and ADLS Gen2.

   - For demonstration purposes, we’ll proceed to the next steps assuming Microsoft Fabric is used to govern and secure data access, ensuring compliance and data quality.

2. Data Processing with Azure Databricks

   1. Load Data from ADLS Gen2:

      - Use the mounted path to read data into a Azure Databricks DataFrame:

        pythonCopy code

        ```
        df = spark.read.csv("/mnt/<mount-name>/path/to/data.csv")
        ```

   2. Data Transformation:

      - Perform necessary data transformations using Spark DataFrame operations.

3. Save Processed Data:

    - Save the processed data back to ADLS Gen2 in an optimised format, such as Delta Lake:

      pythonCopy code

      ```
      df.write.format("delta").save("/mnt/<mount-name>/path/to/
      processed/data/")
      ```

3. Machine Learning Model Training and Deployment

    1. Model Training:

        - Use Azure Databricks to train a machine learning model using the processed data.

        - Example using Spark MLlib for logistic regression:

          pythonCopy code

          ```
          from pyspark.ml.classification import LogisticRegression #
          Assuming df is a DataFrame prepared for training # Train
          a model lr = LogisticRegression(featuresCol='features',
          labelCol='label') model = lr.fit(df) # Save the model model.
          write().overwrite().save("/mnt/<mount-name>/path/to/model")
          ```

    2. Model Deployment:

        - Integrate with Azure Machine Learning for model deployment, monitoring and management. This part is complex and typically involves registering the model, creating a scoring script and deploying the model as a web service.

> **Note**
>
> Prompts and answers reflected here may not reflect your exact experience.

Microsoft