Contents

1 Dataflows Gen2 in Microsoft Fabric

1.1 Introduction

Microsoft Fabric offers a unified solution for data engineering, integration, and analytics. A crucial step in end-to-end analytics is data ingestion. Dataflows Gen2 are used to ingest and transform data from multiple sources, and then land the cleansed data to another destination. They can be incorporated into data pipelines for more complex activity orchestration and also used as a data source in Power BI.

1.2 Use Case

Imagine you work for a retail company with stores across the globe. As a data engineer, you need to prepare and transform data from various sources into a format that is suitable for data analysis and reporting. The business requests a semantic model that consolidates disparate data sources from the different stores. Dataflows Gen2 allow you to prepare the data to ensure consistency, and then stage the data in the preferred destination. They also enable reuse and make it easy to update the data. Without a dataflow, you would have to manually extract and transform the data from every source, which is time-consuming and prone to errors.

1.3 Overview of Dataflows Gen2

Dataflows are a type of cloud-based ETL (Extract, Transform, Load) tool for building and executing scalable data transformation processes.

Dataflows Gen2 allow you to extract data from various sources, transform it using a wide range of transformation operations, and load it into a destination. Using Power Query Online also allows for a visual interface to perform these tasks.

1.4 How to use Dataflows Gen2

Traditionally, data engineers spend significant time extracting, transforming, and loading data into a consumable format for downstream analytics. The goal of Dataflows Gen2 is to provide an easy, reusable way to perform ETL tasks using Power Query Online.

Adding a data destination to your dataflow is optional, and the dataflow preserves all transformation steps. To perform other tasks or load data to a different destination after transformation, create a data pipeline and add the Dataflow Gen2 activity to your orchestration.

1.5 Benefits and Limitations

Benefits:

- Extend data with consistent data, such as a standard date dimension table.
- Allow self-service users access to a subset of data warehouse separately.
- Optimize performance with dataflows, which enable extracting data once for reuse, reducing data refresh time for slower sources.
- Simplify data source complexity by only exposing dataflows to larger analyst groups.
- Ensure consistency and quality of data by enabling users to clean and transform data before loading it to a destination.
- Simplify data integration by providing a low-code interface that ingests data from various sources.

Limitations:

- Dataflows are not a replacement for a data warehouse.
- Row-level security is not supported.
- Fabric capacity workspace is required.

1.6 Power Query Online Interface

- **Power Query ribbon:** Dataflows Gen2 support a wide variety of data source connectors, such as relational databases, Excel files, SharePoint, and Fabric lakehouses.
- **Queries pane:** Displays different data sources as queries, which can be duplicated or referenced.
- **Diagram view:** Visually represents the data sources and applied transformations.
- **Data Preview pane:** Shows a subset of data to preview transformations.
- **Query Settings pane:** Lists applied transformation steps.

1.7 Integration with Pipelines

Dataflows Gen2 provide an excellent option for data transformations in Microsoft Fabric. The combination of dataflows and pipelines is useful when you need to perform additional operations on the transformed data.

1.8 Common Pipeline Activities

- Copy data
- Incorporate Dataflow
- Add Notebook
- Get metadata
- Execute a script or stored procedure

Pipelines provide a visual way to complete activities in a specific order. You can use a dataflow for data ingestion, transformation, and landing into a Fabric data store. Then incorporate the dataflow into a pipeline to orchestrate extra activities, like executing scripts or stored procedures after the dataflow has completed.

1.9 Automation with Pipelines

Pipelines can also be scheduled or activated by a trigger to run your dataflow. By using a pipeline to run your dataflow, you can have the data refreshed when you need it instead of having to manually run the dataflow. When dealing with enterprise or frequently changing data, automation allows you to focus on other responsibilities.

2 Data Pipelines in Microsoft Fabric

2.1 Introduction

Data pipelines define a sequence of activities that orchestrate an overall process, usually by extracting data from one or more sources and loading it into a destination; often transforming it along the way. Pipelines are commonly used to automate extract, transform, and load (ETL) processes that ingest transactional data from operational data stores into an analytical data store, such as a lakehouse, data warehouse, or SQL database.

If you're already familiar with Azure Data Factory, then data pipelines in Microsoft Fabric will be immediately familiar. They use the same architecture of connected activities to define a process that can include multiple kinds of data processing tasks and control flow logic. You can run pipelines interactively in the Microsoft Fabric user interface or schedule them to run automatically.

2.2 Core Pipeline Concepts

Pipelines in Microsoft Fabric encapsulate a sequence of activities that perform data movement and processing tasks. You can use a pipeline to define data transfer and transformation activities, and orchestrate these activities through

control flow activities that manage branching, looping, and other typical processing logic. The graphical pipeline canvas in the Fabric user interface enables you to build complex pipelines with minimal or no coding required.

2.3 Activities

Activities are the executable tasks in a pipeline. You can define a flow of activities by connecting them in a sequence. The outcome of a particular activity (success, failure, or completion) can be used to direct the flow to the next activity in the sequence.

There are two broad categories of activity in a pipeline:

- **Data transformation activities** - activities that encapsulate data transfer operations, including simple Copy Data activities that extract data from a source and load it to a destination, and more complex Data Flow activities that encapsulate dataflows (Gen2) that apply transformations to the data as it is transferred.
- **Control flow activities** - activities that you can use to implement loops, conditional branching, or manage variable and parameter values.

2.4 Parameters

Pipelines can be parameterized, enabling you to provide specific values to be used each time a pipeline is run. For example, you might want to use a pipeline to save ingested data in a folder, but have the flexibility to specify a folder name each time the pipeline is run.

Using parameters increases the reusability of your pipelines, enabling you to create flexible data ingestion and transformation processes.

2.5 Pipeline Runs

Each time a pipeline is executed, a data pipeline run is initiated. Runs can be initiated on-demand in the Fabric user interface or scheduled to start at a specific frequency. Use the unique run ID to review run details to confirm they completed successfully and investigate the specific settings used for each execution.

2.6 Using the Copy Data Activity

The Copy Data activity is one of the most common uses of a data pipeline. Many pipelines consist of a single Copy Data activity that is used to ingest data from an external source into a lakehouse file or table.

You can also combine the Copy Data activity with other activities to create a repeatable data ingestion process - for example by using a Delete Data activity to remove existing data, a Copy Data activity to replace the deleted data with

a file containing data from an external source, and a Notebook activity to run Spark code that transforms the data in the file and loads it into a table.

2.7 Pipeline Templates

You can define pipelines from any combination of activities you choose, enabling you to create custom data ingestion and transformation processes to meet your specific needs. However, there are many common pipeline scenarios for which Microsoft Fabric includes predefined pipeline templates that you can use and customize as required.

To create a pipeline based on a template, select the **Choose a task to start** tile in a new pipeline. Selecting this option displays a selection of pipeline templates that can be customized to fit your specific use case.

2.8 Running and Monitoring Pipelines

When you have completed a pipeline, you can use the **Validate** option to check that the configuration is valid, and then either run it interactively or specify a schedule.

You can view the run history for a pipeline to see details of each run, either from the pipeline canvas or from the pipeline item listed in the page for the workspace.

When you view a pipeline run history from the workspace page, you can select the **Run start** value to see the details of an individual run; including the option to view the individual execution time for each activity as a Gantt chart.

3 Apache Spark in Microsoft Fabric

3.1 Introduction

Apache Spark is an open source parallel processing framework for large-scale data processing and analytics. Spark has become popular in "big data" processing scenarios, and is available in multiple platform implementations; including Azure HDInsight, Azure Synapse Analytics, and Microsoft Fabric.

This section explores how you can use Spark in Microsoft Fabric to ingest, process, and analyze data in a lakehouse. While the core techniques and code described here are common to all Spark implementations, the integrated tools and ability to work with Spark in the same environment as other data services in Microsoft Fabric makes it easier to incorporate Spark-based data processing into your overall data analytics solution.

3.2 Prepare to Use Apache Spark

Apache Spark is a distributed data processing framework that enables large-scale data analytics by coordinating work across multiple processing nodes in a cluster, known in Microsoft Fabric as a Spark pool. Spark uses a "divide and

conquer" approach to processing large volumes of data quickly by distributing the work across multiple computers.

3.3 Spark Pools

A Spark pool consists of compute nodes that distribute data processing tasks:

- A **head node** coordinates distributed processes through a driver program.
- Multiple **worker nodes** execute actual data processing tasks.

Spark pools use this distributed compute architecture to access and process data in a compatible data store, such as a data lakehouse based in OneLake.

3.4 Spark Pools in Microsoft Fabric

Microsoft Fabric provides a starter pool in each workspace, enabling Spark jobs to be started and run quickly with minimal setup and configuration. Users can configure or create custom Spark pools with specific node configurations to support particular data processing needs.

3.5 Runtimes and Environments

Microsoft Fabric supports multiple Spark runtimes and allows users to configure runtime settings, including installed libraries. Organizations may define multiple environments to support a diverse range of data processing tasks, selecting environments based on specific runtime versions and required libraries.

3.6 Additional Spark Configuration Options

Native Execution Engine: A vectorized processing engine in Microsoft Fabric that can significantly improve query performance when working with large datasets in Parquet or Delta file formats.

High Concurrency Mode: Allows multiple users to share Spark sessions across notebooks while maintaining isolation.

Automatic MLFlow Logging: MLFlow is used in data science workloads to track machine learning training and model deployment. Microsoft Fabric supports implicit logging of ML experiments.

3.7 Running Spark Code

Notebooks: Spark in Microsoft Fabric supports interactive data exploration and analysis using notebooks, which contain executable code written in languages such as PySpark and Spark SQL.

Spark Job Definition: Spark jobs can be scheduled to execute scripts automatically as part of an ingestion or transformation workflow.

3.8 Working with Data in Spark DataFrames

Spark primarily operates on **DataFrames**, which are optimized for distributed data processing. Users can:

- Load data from structured sources such as CSV or Delta Lake.
- Transform data using Spark SQL and DataFrame operations.
- Apply explicit schema definitions for better performance.
- Filter, group, and aggregate data using DataFrame methods.
- Save transformed data back to storage using efficient formats like Parquet.

3.9 Working with Data Using Spark SQL

Spark SQL allows querying structured data using standard SQL expressions. Users can create temporary views or persistent tables in the Spark catalog, making data accessible for SQL-based analytics.

3.10 Visualizing Data in a Spark Notebook

Notebooks in Microsoft Fabric provide built-in charting capabilities for visualizing query results. Additionally, users can leverage external libraries such as Matplotlib or Seaborn to create customized visualizations.

This section provides a foundational understanding of how to use Apache Spark within Microsoft Fabric for efficient data processing and analytics.

4 Real-Time Intelligence in Microsoft Fabric

4.1 Introduction

Most modern data analytics solutions enable two common patterns for data analysis:

- **Batch data analytics**, in which data is loaded into an analytical data store at periodic intervals as a batch operation; enabling historical analysis of data from past events.
- **Real-time data analytics**, in which data from events is ingested in real-time (or near real-time) as events occur in a stream of data that can be analyzed, visualized, and used to trigger automated responses.

Microsoft Fabric provides capabilities for both batch and real-time analytics. This section focuses on the Real-Time Intelligence features of Microsoft Fabric to explore how you can build real-time data analytics solutions with minimal coding that scale to huge volumes of data from a diverse range of sources.

4.2 Understanding Real-Time Data Analytics

Real-time data analytics is based on the ingestion and processing of a data stream consisting of a continuous series of data related to point-in-time events. Examples include:

- Messages submitted to a social media platform.
- Environmental measurements recorded by IoT sensors.
- Financial transaction logs analyzed in real-time.

Key characteristics of real-time analytics solutions include:

- Unbounded data streams.
- Temporal event data.
- Aggregation over defined time windows.
- Use of processed results for real-time automation or visualization.

4.3 Fabric's Real-Time Intelligence Capabilities

Microsoft Fabric's Real-Time Intelligence workload provides:

- Eventstreams to capture and process real-time data.
- Eventhouses to store real-time event data.
- KQL databases for querying and analyzing event data.
- Power BI and dashboards for real-time visualization.
- Activator for triggering automated actions based on real-time data insights.

4.4 Eventstreams: Ingesting and Transforming Real-Time Data

Eventstreams in Microsoft Fabric enable users to capture, transform, and load real-time data from various sources, such as:

- Azure Event Hubs and IoT Hubs.
- Apache Kafka streams.
- Change Data Capture (CDC) feeds.
- Fabric workspace events.

Data transformation options include:

- Filtering data records.
- Aggregating metrics over time windows.
- Joining data from multiple streams.
- Expanding nested data structures.

4.5 Storing and Querying Real-Time Data

Real-time data ingested through eventstreams can be stored in an eventhouse, which contains:

- **KQL databases**, optimized for querying large volumes of event data using Kusto Query Language (KQL).
- **KQL querysets**, collections of queries used to extract insights from eventhouse data.

Users can write KQL queries to analyze streaming data, aggregate trends, and create materialized views for further insights.

4.6 Visualizing Real-Time Data

Real-time dashboards provide interactive visualization capabilities for monitoring live data. Users can:

- Pin data visualizations from KQL queries to dashboards.
- Interact with data in Power BI reports.
- Configure charts and filters for better analysis.

4.7 Automating Actions with Activator

Activator is a rule-based automation tool in Microsoft Fabric that enables:

- Real-time alerts for anomalies or threshold breaches.
- Automated workflow execution based on incoming events.
- Notifications and external API triggers based on real-time conditions.

4.8 Conclusion

Microsoft Fabric's Real-Time Intelligence capabilities enable organizations to build scalable, low-code real-time analytics solutions. With eventstreams, eventhouses, KQL-based queries, dashboards, and automated actions, businesses can leverage real-time insights to drive operational efficiency and intelligent decision-making.

5 Real-Time Evenstreams in Microsoft Fabric

5.1 Introduction

Most modern data analytics solutions enable two common patterns for data analysis:

- **Batch data analytics**, in which data is loaded into an analytical data store at periodic intervals as a batch operation; enabling historical analysis of data from past events.
- **Real-time data analytics**, in which data from events is ingested in real-time (or near real-time) as events occur in a stream of data that can be analyzed, visualized, and used to trigger automated responses.

Microsoft Fabric provides capabilities for both batch and real-time analytics. This section focuses on the Real-Time Intelligence features of Microsoft Fabric to explore how you can build real-time data analytics solutions with minimal coding that scale to huge volumes of data from a diverse range of sources.

5.2 Eventstreams: Capturing and Processing Real-Time Data

Eventstreams enable you to capture and process real-time events without needing to write any code. You can set up event sources, destinations, and transformations in the event stream, allowing you to collect, filter, aggregate, group, and combine data from real-time sources before loading it into a destination.

Benefits of Microsoft Fabric Eventstreams:

- No-code experience for capturing, transforming, and routing real-time events.
- Multiple source connectors including Azure Event Hubs, Azure IoT Hub, Apache Kafka, and others.
- Ability to send data to various destinations such as eventhouses, lakehouses, and custom endpoints.
- Scalable infrastructure for streaming ETL operations.

5.3 Components of Eventstreams

Microsoft Fabric eventstreams work by creating a pipeline of events from multiple internal and external sources to different destinations. The main components are:

- **Sources**: Where event data originates, including Azure Event Hubs, IoT Hubs, and various CDC connectors.

- **Transformations**: Data operations such as filtering, joining, aggregating, and grouping to refine the event data.
- **Destinations**: Where transformed data is stored, including eventhouses, lakehouses, and custom applications.

5.4 Eventstream Sources and Destinations

Event Sources:

- Azure Event Hub, Azure IoT Hub, and various CDC connectors.
- External sources such as Google Cloud Pub/Sub, Amazon Kinesis, and Confluent Cloud Kafka.
- Fabric workspace events, Azure Blob Storage events, and custom endpoints.

Event Destinations:

- **Eventhouse**: Real-time event data is stored in a KQL database for analysis.
- **Lakehouse**: Preprocessed real-time events are stored in Delta Lake format.
- **Custom Endpoint**: Real-time data is sent to an external system or application.
- **Fabric Activator**: Enables automated responses to streaming data.

5.5 Eventstream Transformations

The drag-and-drop eventstream interface allows you to construct complex event data processing workflows. Key transformations include:

- **Filter**: Retain events based on specific conditions.
- **Manage Fields**: Modify, add, or remove fields.
- **Aggregate**: Compute sum, min, max, or average over time windows.
- **Group By**: Aggregate events across defined time windows.
- **Union**: Merge multiple event streams.
- **Expand**: Convert arrays into separate event rows.
- **Join**: Combine data from two streams based on a matching condition.

5.6 Windowing Functions in Eventstreams

Windowing functions allow analysis of streaming events within specific time frames:

- **Tumbling Windows**: Fixed, non-overlapping time intervals.
- **Sliding Windows**: Overlapping time intervals.
- **Session Windows**: Variable-length intervals based on activity gaps.
- **Hopping Windows**: Overlapping windows with a fixed refresh interval.
- **Snapshot Windows**: Grouping of events that occur at the same timestamp.

5.7 Storing and Querying Real-Time Data

Real-time data ingested through eventstreams can be stored in an eventhouse, which contains:

- **KQL databases**: Optimized for querying large volumes of event data using Kusto Query Language (KQL).
- **KQL querysets**: Collections of queries used to extract insights from eventhouse data.

Users can write KQL queries to analyze streaming data, aggregate trends, and create materialized views for further insights.

5.8 Visualizing Real-Time Data

Real-time dashboards provide interactive visualization capabilities for monitoring live data. Users can:

- Pin data visualizations from KQL queries to dashboards.
- Interact with data in Power BI reports.
- Configure charts and filters for better analysis.

5.9 Automating Actions with Activator

Activator is a rule-based automation tool in Microsoft Fabric that enables:

- Real-time alerts for anomalies or threshold breaches.
- Automated workflow execution based on incoming events.
- Notifications and external API triggers based on real-time conditions.

5.10 Conclusion

Microsoft Fabric's Real-Time Intelligence capabilities enable organizations to build scalable, low-code real-time analytics solutions. With eventstreams, eventhouses, KQL-based queries, dashboards, and automated actions, businesses can leverage real-time insights to drive operational efficiency and intelligent decision-making.

6 Work with real time-data

6.1 Introduction

Most modern data analytics solutions enable two common patterns for data analysis:

- **Batch data analytics**, in which data is loaded into an analytical data store at periodic intervals as a batch operation; enabling historical analysis of data from past events.
- **Real-time data analytics**, in which data from events is ingested in real-time (or near real-time) as events occur in a stream of data that can be analyzed, visualized, and used to trigger automated responses.

Microsoft Fabric provides capabilities for both batch and real-time analytics. This section focuses on the Real-Time Intelligence features of Microsoft Fabric to explore how you can build real-time data analytics solutions with minimal coding that scale to huge volumes of data from a diverse range of sources.

6.2 Eventstreams: Capturing and Processing Real-Time Data

Eventstreams enable you to capture and process real-time events without needing to write any code. You can set up event sources, destinations, and transformations in the event stream, allowing you to collect, filter, aggregate, group, and combine data from real-time sources before loading it into a destination.

Benefits of Microsoft Fabric Eventstreams:

- No-code experience for capturing, transforming, and routing real-time events.
- Multiple source connectors including Azure Event Hubs, Azure IoT Hub, Apache Kafka, and others.
- Ability to send data to various destinations such as eventhouses, lakehouses, and custom endpoints.
- Scalable infrastructure for streaming ETL operations.

6.3 Components of Eventstreams

Microsoft Fabric eventstreams work by creating a pipeline of events from multiple internal and external sources to different destinations. The main components are:

- **Sources**: Where event data originates, including Azure Event Hubs, IoT Hubs, and various CDC connectors.
- **Transformations**: Data operations such as filtering, joining, aggregating, and grouping to refine the event data.
- **Destinations**: Where transformed data is stored, including eventhouses, lakehouses, and custom applications.

6.4 Eventhouses: Storing and Querying Real-Time Data

An eventhouse in Microsoft Fabric provides a data store for large volumes of data optimized for time-based events, making it ideal for real-time analytics. Common uses include:

- Querying data using Kusto Query Language (KQL) or a subset of SQL in a KQL Queryset.
- Visualizing data in Real-Time Dashboards.
- Automating actions using Fabric Activator.

An eventhouse typically includes one or more KQL databases, where you can create tables, stored procedures, materialized views, and other analytical structures. Data is often loaded into an eventhouse from static locations (such as OneLake, Azure Storage, or local files) or real-time sources (such as eventstreams or Azure Event Hubs).

6.5 Querying Data in KQL Databases

To analyze data stored in an eventhouse, KQL queries can be used. Examples include:

Retrieving all data from a table:

```
Automotive
```

Equivalent SQL query:

```
SELECT * FROM Automotive;
```

Filtering rows:

```
Automotive
| where fare_amount > 20
| project trip_id, pickup_datetime, fare_amount
```

Equivalent SQL query:

```
SELECT trip_id, pickup_datetime, fare_amount
FROM Automotive
WHERE fare_amount > 20;
```

Grouping and Aggregation:

```
Automotive
| summarize trip_count = count() by vendor_id
| project vendor_id, trip_count
```

Equivalent SQL query:

```
SELECT vendor_id, COUNT(*) AS trip_count
FROM Automotive
GROUP BY vendor_id;
```

KQL is optimized for handling large amounts of data efficiently, with advantages in simplicity, performance, and integration within the Microsoft ecosystem.

6.6 Optimizing KQL Queries

To improve performance in querying large real-time datasets, consider:

- **Filtering columns**: Use the `project` keyword to retrieve only necessary columns.
- **Filtering rows**: Use the `where` keyword to limit results based on specific conditions.
- **Summarizing data**: Aggregate data at meaningful levels instead of querying individual events.
- **Using time-based functions**: Functions like `ago()` and `ingestion_time()` help retrieve recent data efficiently.

6.7 Materialized Views and Stored Functions

Materialized Views: A materialized view is a precomputed summary of a dataset that is automatically updated as new data is ingested. Example:

```
.create materialized-view TripsByVendor on table Automotive
{
    Automotive
    | summarize trips = count() by vendor_id, pickup_date = format_datetime(pickup_datetime,
}
```

This enables faster queries by maintaining aggregated results in a ready-to-use format.

Stored Functions: KQL functions encapsulate query logic for reuse. Example:

```
.create-or-alter function trips_by_min_passenger_count(num_passengers:long)
{
    Automotive
    | where passenger_count >= num_passengers
    | project trip_id, pickup_datetime
}
```

Calling the function:

```
trips_by_min_passenger_count(3) | take 10
```

6.8 Visualizing Real-Time Data

Real-time dashboards provide interactive visualization capabilities for monitoring live data. Users can:

- Pin data visualizations from KQL queries to dashboards.
- Interact with data in Power BI reports.
- Configure charts and filters for better analysis.

6.9 Automating Actions with Activator

Activator is a rule-based automation tool in Microsoft Fabric that enables:

- Real-time alerts for anomalies or threshold breaches.
- Automated workflow execution based on incoming events.
- Notifications and external API triggers based on real-time conditions.

6.10 Conclusion

Microsoft Fabric's Real-Time Intelligence capabilities enable organizations to build scalable, low-code real-time analytics solutions. With eventstreams, eventhouses, KQL-based queries, dashboards, and automated actions, businesses can leverage real-time insights to drive operational efficiency and intelligent decision-making.

7 Introduction to end-to-end analytics using Microsoft Fabric

7.1 Introduction

Microsoft Fabric is an end-to-end analytics platform that provides a single, integrated environment for data professionals and the business to collaborate on data projects. Fabric provides a set of integrated services that enable you to ingest, store, process, and analyze data in a single environment.

Microsoft Fabric provides tools for both citizen and professional data practitioners, and integrates with tools the business needs to make decisions. Fabric includes the following services:

- Data engineering
- Data integration
- Data warehousing
- Real-time intelligence
- Data science
- Business intelligence

This module introduces the Fabric platform, discusses who Fabric is for, and explores Fabric services.

7.2 End-to-End Analytics with Microsoft Fabric

Microsoft Fabric provides a unified analytics environment that removes complexity and fragmentation. With Fabric, organizations do not need to integrate multiple third-party services but can rely on a single product that is easy to understand, set up, and manage. Fabric offers persona-optimized experiences and tools within an integrated user interface.

Fabric is built as a Software-as-a-Service (SaaS) platform, ensuring scalability, cost-effectiveness, and accessibility from anywhere with an internet connection. Updates and maintenance are managed by Microsoft.

7.3 OneLake: The Core of Microsoft Fabric

OneLake is Fabric's lake-centric architecture, providing a unified storage layer accessible by all analytics engines in Fabric. It eliminates the need for data duplication and facilitates collaboration across teams. OneCopy, a feature of OneLake, enables reading data from a single copy without requiring movement or replication.

- OneLake is built on Azure Data Lake Storage (ADLS) and supports formats like Delta, Parquet, CSV, and JSON.

- Fabric's compute engines store data in OneLake automatically, enabling seamless access across different workloads.
- Shortcuts allow referencing other storage locations without duplicating data.

7.4 Fabric's Analytics Experiences

Fabric includes various analytics experiences, each tailored for specific workloads:

- **Synapse Data Engineering**: Spark-based data transformation at scale.
- **Synapse Data Warehouse**: High-performance SQL-based data warehousing.
- **Synapse Data Science**: Machine learning and model training within Fabric.
- **Synapse Real-Time Intelligence**: Real-time querying and large-scale data analysis.
- **Data Factory**: Data integration using Power Query and Azure Data Factory capabilities.
- **Power BI**: Business intelligence and data visualization.

7.5 Workspaces in Microsoft Fabric

Workspaces in Fabric act as logical containers for managing data assets, reports, and analytics items. They facilitate:

- Collaboration across teams.
- Secure access control and permission management.
- Integration with Git for version control.
- Compute resource allocation and management.

7.6 Security and Governance in Fabric

Fabric's OneLake provides centralized security and governance. Administrators manage data access, security policies, and compliance settings through the Fabric Admin Center. Fabric integrates natively with Microsoft Purview for data classification and protection.

7.7 Collaboration and Role-Based Workflows

Fabric streamlines collaboration between various data professionals:

- **Data engineers** manage data ingestion and transformation.
- **Data analysts** create reports and dashboards with Power BI.
- **Data scientists** integrate machine learning techniques.
- **Analytics engineers** curate datasets and enable self-service analytics.
- **Business users** leverage curated data without requiring technical expertise.

7.8 Enabling Microsoft Fabric

To use Microsoft Fabric, an organization must enable it via the Fabric Admin Center. This requires permissions as a Fabric admin, Power Platform admin, or Microsoft 365 admin. Organizations can enable Fabric for all users or restrict access to specific groups.

7.9 Exploring OneLake Data Hub

OneLake Data Hub allows users to:

- Discover and connect to organizational data sources.
- Narrow results by domains and filter by workspaces.
- Explore endorsed datasets for business use.

7.10 Creating Items in Fabric

Fabric users can create items such as lakehouses, warehouses, reports, and pipelines. The platform supports various workloads, allowing users to switch between experiences seamlessly.

7.11 Conclusion

Microsoft Fabric is a fully integrated analytics platform designed to streamline data workflows, enhance collaboration, and provide a scalable SaaS-based solution for end-to-end analytics. By centralizing data storage, governance, and analytics, Fabric empowers organizations to derive insights efficiently without the complexity of multiple disparate services.

8 Microsoft Fabric Lakehouse

8.1 Introduction

The foundation of Microsoft Fabric is a lakehouse, which is built on top of the OneLake scalable storage layer and uses Apache Spark and SQL compute engines for big data processing. A lakehouse is a unified platform that combines:

- The flexible and scalable storage of a data lake.
- The ability to query and analyze data of a data warehouse.

Microsoft Fabric provides a scalable and flexible data store for files and tables that can be queried using SQL.

8.2 Exploring the Microsoft Fabric Lakehouse

A lakehouse presents as a database and is built on top of a data lake using Delta format tables. Lakehouses combine the SQL-based analytical capabilities of a relational data warehouse and the flexibility and scalability of a data lake.

Some benefits of a lakehouse include:

- Use of Spark and SQL engines to process large-scale data.
- Schema-on-read format, defining schema as needed rather than being predefined.
- Support for ACID (Atomisk, konsistent, isolert, durabel) transactions through Delta Lake formatted tables.
- Single access point for data engineers, data scientists, and analysts.

8.3 Loading Data into a Lakehouse

Fabric lakehouses are a central element for an analytics solution. The ETL (Extract, Transform, Load) process is followed to ingest and transform data before loading it into the lakehouse.

Methods for ingesting data:

- Upload local files.
- Use Dataflows Gen2 to import and transform data.
- Use Apache Spark notebooks for ingestion and transformation.
- Utilize Data Factory pipelines for orchestrating ETL activities.

8.4 Security and Governance

Lakehouse access is managed through workspace or item-level sharing:

- Workspace roles grant access to all items within the workspace.
- Item-level sharing is best for read-only access.
- Microsoft Purview integration provides advanced data governance features.

8.5 Working with a Microsoft Fabric Lakehouse

When creating a lakehouse, three data items are automatically generated:

- The lakehouse containing shortcuts, folders, files, and tables.
- A default Semantic model for Power BI integration.
- A SQL analytics endpoint for read-only SQL querying.

Data can be accessed through:

- The Lakehouse explorer to manage tables, files, and folders.
- The SQL analytics endpoint to query data using SQL.

8.6 Accessing Data with Shortcuts

Shortcuts enable integration of external data sources without duplication. Shortcuts can point to storage accounts, warehouses, and KQL databases, ensuring seamless data access across various platforms.

8.7 Transforming and Analyzing Data

Data transformation can be performed using:

- Notebooks with PySpark, SQL, or Scala.
- Dataflows Gen2 with PowerQuery for visual transformations.
- Pipelines for complex ETL orchestration.

After data is processed, it can be analyzed using:

- Power BI for visualization and reporting.
- Notebooks for machine learning models.
- The SQL analytics endpoint for structured queries.

8.8 Conclusion

By integrating data storage, processing, and analytics, Microsoft Fabric Lakehouses provide a powerful platform for scalable, real-time, and governed data solutions. The combination of SQL and Spark capabilities makes them suitable for various analytics workloads.

9 Microsoft Fabric Lakehouse v2

9.1 Introduction

Tables in a Microsoft Fabric lakehouse are based on the Linux Foundation Delta Lake table format, commonly used in Apache Spark. Delta Lake is an open-source storage layer for Spark that enables relational database capabilities for batch and streaming data. By using Delta Lake, you can implement a lakehouse architecture to support SQL-based data manipulation semantics in Spark with support for transactions and schema enforcement. This results in an analytical data store that offers many of the advantages of a relational database system with the flexibility of data file storage in a data lake.

While you don't need to work directly with Delta Lake APIs to use tables in a Fabric lakehouse, an understanding of the Delta Lake metastore architecture and familiarity with some of the more specialized Delta table operations can greatly expand your ability to build advanced analytics solutions on Microsoft Fabric.

9.2 Understanding Delta Lake

Delta Lake is an open-source storage layer that adds relational database semantics to Spark-based data lake processing. Tables in Microsoft Fabric lakehouses are Delta tables, signified by the triangular Delta (Δ) icon on tables in the lakehouse user interface.

Delta tables are schema abstractions over data files stored in Delta format. Each table contains a folder with Parquet data files and a `_delta_log` folder where transaction details are logged in JSON format.

Benefits of using Delta tables:

- Relational tables that support querying and data modification (CRUD operations).
- Support for ACID transactions ensuring data consistency and integrity.
- Data versioning and time travel features.
- Support for batch and streaming data processing.
- Standard formats and interoperability with SQL-based analytics.

9.3 Creating Delta Tables

When you create a table in a Microsoft Fabric lakehouse, a Delta table is defined in the metastore and stored in the underlying Parquet files.

Creating a Delta table from a dataframe in PySpark:

```
# Load a file into a dataframe
df = spark.read.load('Files/mydata.csv', format='csv', header=True)

# Save the dataframe as a Delta table
df.write.format("delta").saveAsTable("mytable")
```

Managed vs External Tables:

- Managed tables: The Spark runtime manages both the table definition and the underlying data files.
- External tables: The table definition is created in the metastore, but the data is stored in an external location.

9.4 Optimizing Delta Tables

Delta Lake includes optimizations to improve query performance and storage efficiency:

- **OptimizeWrite**: Reduces the number of small files when writing data.
- **Optimize**: Consolidates small Parquet files into fewer large files.
- **V-Order**: Optimizes Parquet files for faster reads.
- **Vacuum**: Removes old data files to free up storage.
- **Partitioning**: Organizes data into partitions to improve query performance.

Applying V-Order optimization in SQL:

```
%%sql
OPTIMIZE mytable APPLY VORDER;
```

Running VACUUM to clean up old files:

```
%%sql
VACUUM mytable RETAIN 168 HOURS;
```

9.5 Working with Delta Tables in Spark

Using Spark SQL to manipulate Delta tables:

```
%%sql
UPDATE products
SET Price = 2.49 WHERE ProductId = 1;
```

Using the Delta API in PySpark:

```
from delta.tables import *
deltaTable = DeltaTable.forPath(spark, "Files/mytable")
deltaTable.update(condition = "Category == 'Accessories'", set = { "Price": "Price * 0.9" })
```

9.6 Streaming Data with Delta Tables

Delta tables can be used as a streaming source or sink in Spark Structured Streaming.

Creating a streaming DataFrame:

```
stream_df = spark.readStream.format("delta") \
    .option("ignoreChanges", "true") \
    .table("orders_in")
```

Writing streaming data to a Delta table:

```
deltastream = transformed_df.writeStream.format("delta") \
    .option("checkpointLocation", "Files/delta/checkpoint") \
    .start("Tables/orders_processed")
```

Stopping the streaming process:

```
deltastream.stop()
```

9.7 Conclusion

Microsoft Fabric lakehouses, built on Delta Lake, provide a robust solution for big data analytics by combining the flexibility of data lakes with the transactional capabilities of data warehouses. With support for ACID transactions, schema enforcement, and advanced optimization techniques, Delta tables enable efficient data processing, querying, and real-time streaming analytics.

10 Medallion Architecture in Microsoft Fabric

10.1 Introduction

In this data-driven era, the need to harness vast and diverse data sources is critical for business success. The Fabric lakehouse, blending data lakes and data warehouses, offers an ideal platform to manage and analyze this data. The

medallion architecture has become a standard across the industry for lakehouse-based analytics.

The medallion architecture brings structure and efficiency to your lakehouse environment. This section guides you through its bronze, silver, and gold layers, ensuring you know how to organize, refine, and curate data effectively.

10.2 Describe Medallion Architecture

Data lakehouses in Fabric are built on the Delta Lake format, which natively supports ACID (Atomicity, Consistency, Isolation, Durability) transactions. Within this framework, the medallion architecture is a recommended data design pattern used to organize data in a lakehouse logically. It aims to improve data quality as it moves through different layers. The architecture typically has three layers:

- **Bronze Layer**: The raw landing zone for structured, semi-structured, or unstructured data.
- **Silver Layer**: The validated layer where data is cleaned, deduplicated, and transformed into a more consistent format.
- **Gold Layer**: The enriched layer where data is further refined and aggregated for specific business needs.

The medallion architecture ensures that data is reliable, consistent, and efficiently structured for analysis.

10.3 Implementing a Medallion Architecture in Fabric

Set up the foundation: Create a Fabric lakehouse that will serve as the data storage and processing environment.

Design your architecture: Define the three layers and determine how data will flow between them.

Move data across layers: Utilize Fabric's built-in tools:

- **Pipelines**: For orchestrating data movement.
- **Dataflows and Notebooks**: For transforming and loading data.
- **SQL analytics endpoint**: For querying and modeling data.

10.4 Query and Report on Data in Fabric Lakehouse

Query data using SQL: Teams can explore and analyze data in the gold layer using SQL analytics endpoints.

Power BI Integration: Direct Lake mode allows Power BI to cache frequently accessed data while ensuring real-time accuracy.

Multiple Gold Layers: Organizations can create domain-specific gold layers for different departments such as finance, sales, and data science.

10.5 Managing Security and CI/CD in a Lakehouse

Security Best Practices: Implement role-based access controls at the workspace and item level.

Continuous Integration and Continuous Deployment (CI/CD):

- Utilize Git integration for version control.
- Automate deployments for reliable data pipeline execution.
- Monitor and enforce data quality checks.

Conclusion: The medallion architecture in Microsoft Fabric offers a scalable and structured approach to data management, ensuring high-quality, curated data for analytics and reporting.

11 Real-Time Dashboards in Microsoft Fabric

11.1 Introduction

Imagine you're an analyst in a company that provides bicycles for a bike-sharing scheme in a large city. The business depends on real-time data relating to bike availability in different neighborhoods across the city. The challenge is to create a real-time dashboard that provides real-time insights into the number of bikes and parking docks in multiple locations.

This module explores real-time dashboards in Microsoft Fabric, teaching you how to create and use them to visualize and explore real-time data.

11.2 Get Started with Real-Time Dashboards

Real-time dashboards in Microsoft Fabric are built on real-time streaming data sources, such as tables in an Eventhouse populated by an eventstream. Each dashboard consists of one or more tiles, each displaying a real-time data visualization.

To create a real-time dashboard, you'll need a source of real-time data, such as an Eventhouse containing a KQL database. You can then create a real-time dashboard with a data source that references this real-time data.

Authorization Schemes:

- **Pass-through identity**: Data in the dashboard is accessed using the identity of the user viewing it.
- **Dashboard editor's identity**: Data is accessed using the identity of the user who created the dashboard.

Creating Tiles: A dashboard contains at least one tile displaying the results of a KQL query. For example, the following KQL query retrieves the most recent observation within the last 30 minutes for each neighborhood:

```
bikes
| where ingestion_time() between (ago(30min) .. now())
| summarize latest_observation = arg_max(ingestion_time(), *) by Neighbourhood
| project Neighbourhood, latest_observation, No_Bikes, No_Empty_Docks
| order by Neighbourhood asc
```

Initially, the tile displays the results as a table, but you can edit it to define a visualization, such as a bar chart.

11.3 Advanced Features

Fabric real-time dashboards support advanced features such as:

Base Queries: These retrieve general data for multiple tiles, making dashboards easier to maintain.

```
bikes
| where ingestion_time() between (ago(30min) .. now())
| summarize latest_observation = arg_max(ingestion_time(), *) by Neighbourhood
```

Pages: Multiple pages can be added to a dashboard for organizing data based on sources, subjects, or aggregation levels.

Parameters: Parameters allow users to filter displayed data dynamically. Example:

```
bikes
| where ingestion_time() between (ago(30min) .. now())
  and (isempty(['selected_neighbourhoods']) or Neighbourhood in (['selected_neighbourhoods']
| summarize latest_observation = arg_max(ingestion_time(), *) by Neighbourhood
```

Auto Refresh: The dashboard automatically updates without requiring a manual refresh. Editors can set default refresh rates, with minimum thresholds to manage system load.

11.4 Best Practices for Real-Time Dashboards

When implementing real-time dashboards in Microsoft Fabric, consider these best practices:

- **Clarity and Simplicity**: Keep the dashboard simple and avoid clutter.
- **Relevance**: Display only relevant data.
- **Refresh Rate**: Optimize refresh rates for real-time insights without overloading the system.
- **Accessibility**: Ensure dashboards are accessible to all users.
- **Interactivity**: Include filters and drill-down features.
- **Performance**: Optimize queries to reduce load times.

- **Security**: Protect sensitive data through authentication and authorization.
- **Testing**: Regularly test dashboards for functionality and user feedback.

Following these best practices will ensure your real-time dashboards provide timely, actionable insights while remaining efficient and secure.

12 Data Warehousing in Microsoft Fabric

12.1 Introduction

Relational data warehouses are the backbone of most enterprise business intelligence (BI) solutions. Microsoft Fabric provides a modernized version of the traditional data warehouse, centralizing and organizing data from different sources into a single, unified view for analysis and reporting. Fabric's data warehouse supports full SQL semantics and is built on the Lakehouse, using Delta format storage.

12.2 Data Warehouse Fundamentals

The process of building a modern data warehouse typically consists of:

- **Data ingestion** - Moving data from source systems into the warehouse.
- **Data storage** - Optimizing data storage for analytics.
- **Data processing** - Transforming data for analytical consumption.
- **Data analysis and delivery** - Providing insights and reports to the business.

Fabric enables data engineers and analysts to handle all these steps in one integrated platform.

12.3 Fabric's Data Warehouse Experience

Microsoft Fabric's data warehouse supports:

- Full T-SQL transactional capabilities.
- Scalable and highly available architecture.
- Integration with Power BI and SQL-based analytics.

Data engineers can build a relational layer on top of data in the Lakehouse, allowing analysts to query using T-SQL and visualize insights in Power BI.

12.4 Data Warehouse Schema Design

Tables in a Fabric data warehouse are structured to support efficient analysis, typically organized in a **star schema** or **snowflake schema**.

- **Fact tables** store numerical data for analysis.
- **Dimension tables** provide descriptive attributes for fact tables.

Special types of dimensions:

- **Time dimensions** provide information about time-based events.
- **Slowly changing dimensions** track changes in attributes over time.

12.5 Creating and Managing a Data Warehouse in Fabric

To create a data warehouse in Fabric:

1. Navigate to the Fabric portal and create a new data warehouse.
2. Use SQL commands or the Fabric UI to define tables and schemas.
3. Load data using pipelines, dataflows, or the `COPY INTO` command.
4. Perform transformations using T-SQL and optimize query performance.

Example SQL command to load data:

```
COPY INTO dbo.Region
FROM 'https://storageaccount.blob.core.windows.net/data/Region.csv'
WITH (FILE_TYPE = 'CSV', CREDENTIAL = (IDENTITY = 'SAS', SECRET = 'xxx'));
```

12.6 Querying and Transforming Data

Fabric provides multiple ways to query and transform data:

- **SQL Query Editor**: Write and execute T-SQL queries.
- **Visual Query Editor**: Drag-and-drop interface for queries.
- **SQL Analytics Endpoint**: Connect external tools for analysis.

12.7 Building a Semantic Model for Analysis

A **semantic model** defines relationships between tables and contains measures for reporting. Fabric automatically creates a semantic model when a data warehouse is created. Analysts can:

- Define relationships between tables.
- Create measures using Data Analysis Expressions (DAX).
- Hide fields to simplify data models for end users.

12.8 Security and Monitoring in Fabric Data Warehouses

Fabric provides robust security and monitoring features:

- **Role-Based Access Control (RBAC)**: Control access at workspace and item levels.
- **Item Permissions**: Grant read-only or edit permissions for specific users.
- **Monitoring Queries**: Use Dynamic Management Views (DMVs) to monitor performance.

Example query to identify long-running processes:

```
SELECT request_id, session_id, start_time, total_elapsed_time
FROM sys.dm_exec_requests
WHERE status = 'running'
ORDER BY total_elapsed_time DESC;
```

Terminate a long-running session:

```
KILL 'SESSION_ID';
```

(Note: Requires workspace admin permissions.)

12.9 Conclusion

Microsoft Fabric provides a modernized, scalable, and fully managed data warehouse experience that integrates SQL analytics, real-time processing, and AI-driven insights. With a seamless combination of relational modeling, T-SQL querying, Power BI visualization, and security features, organizations can efficiently analyze and manage data at scale.

13 Microsoft Fabric Data Warehouse v2

13.1 Introduction

Microsoft Fabric Data Warehouse is a complete platform for data, analytics, and Artificial Intelligence (AI). It refers to the process of storing, organizing, and managing large volumes of structured and semi-structured data.

Microsoft Fabric's Data Warehouse, powered by Synapse Analytics, offers a rich set of features that simplify data management and analysis. It includes advanced query processing capabilities and supports full transactional T-SQL functionalities like an enterprise data warehouse.

Unlike a dedicated SQL pool in Synapse Analytics, a warehouse in Microsoft Fabric is centered around a single data lake. Data in the Microsoft Fabric warehouse is stored in the Parquet file format, allowing users to focus on tasks such as data preparation, analysis, and reporting. It leverages the SQL engine's extensive capabilities, storing a unique copy of the data in Microsoft OneLake.

13.2 ETL Process in Microsoft Fabric

ETL (Extract, Transform, and Load) provides the foundation for data analytics and data warehouse operations. The main steps include:

- **Data Extraction**: Connecting to source systems and collecting necessary data.
- **Data Transformation**: Converting extracted data into a standardized format through cleaning, deduplication, and validation.
- **Data Loading**: Storing transformed data into fact and dimension tables. Incremental loads apply ongoing changes, ensuring data consistency.
- **Post-Load Optimization**: Enhancing performance through indexing, partitioning, and materialized views.

13.3 Data Load Strategies

Fabric offers multiple ways to load data into a warehouse. The efficiency of data loading directly impacts real-time analytics and decision-making.

Types of Data Loads:

- **Full Load**: Truncates and reloads all tables, losing old data. Useful for initial setups.
- **Incremental Load**: Updates the warehouse with changes since the last load. More efficient for regular updates.

Surrogate keys and business keys are critical for managing relationships within the data warehouse. Surrogate keys provide consistency across datasets, while business keys maintain traceability to source systems.

13.4 Data Pipelines in Microsoft Fabric

Data pipelines enable large-scale data ingestion and transformation with both coding and no-code experiences. Fabric integrates Azure Data Factory functionality, allowing:

- Automated data ingestion through scheduled pipelines.
- Staging areas for temporary storage and transformation.
- Integration with various sources, including Azure SQL, OneLake, and external cloud services.

13.5 Loading Data using T-SQL

SQL developers can efficiently load data using the `COPY` statement to import data from Azure Storage accounts. Example syntax:

```
COPY INTO my_table
FROM 'https://myaccount.blob.core.windows.net/myblobcontainer/folder0/*.csv'
WITH (
    FILE_TYPE = 'CSV',
    CREDENTIAL=(IDENTITY= 'Shared Access Signature', SECRET='<Your_SAS_Token>')
)
```

Data can also be queried across multiple warehouses and lakehouses in the same workspace using cross-database querying.

13.6 Dataflow Gen2 for Transformation

Dataflow Gen2 provides a Power Query experience to simplify data transformations. Users can:

- Import, clean, and reshape data.
- Utilize Copilot for AI-driven transformations.
- Load transformed data into Azure SQL, Lakehouse, Synapse, and Fabric Warehouse.

13.7 Security and Monitoring

Microsoft Fabric includes role-based access control (RBAC), encryption, and integration with Microsoft Entra ID. Monitoring capabilities include:

- Dynamic Management Views (DMVs) for tracking queries and sessions.
- Performance monitoring to optimize SQL execution.
- Error logging and rejected row handling for ETL processes.

13.8 Conclusion

Microsoft Fabric Data Warehouse integrates SQL, AI, and data transformation capabilities, enabling efficient data management and analytics. With ETL pipelines, real-time dashboards, and automated dataflows, Fabric simplifies data engineering while maintaining security and scalability.

14 Microsoft Fabric Data Warehouse v3

14.1 Introduction

Microsoft Fabric Data Warehouse is a complete platform for data, analytics, and Artificial Intelligence (AI). It refers to the process of storing, organizing, and managing large volumes of structured and semi-structured data.

Microsoft Fabric's Data Warehouse, powered by Synapse Analytics, offers a rich set of features that simplify data management and analysis. It includes advanced query processing capabilities and supports full transactional T-SQL functionalities like an enterprise data warehouse.

The process of querying a data warehouse is a key component of business intelligence. It involves extracting and manipulating data stored in the warehouse to gain insights from large volumes of data.

14.2 Star Schema Design

In a typical data warehouse, data is organized using a schema, often a star schema or a snowflake schema. The star schema and snowflake schema are mature modeling approaches widely adopted by relational data warehouses, classifying tables as either dimension or fact.

Fact tables store measurable, quantitative business data, while dimension tables contain descriptive attributes related to fact data.

14.3 Querying Data

Once the dimension and fact tables in a data warehouse are populated with data, T-SQL can be used to query these tables and perform data analysis. The Transact-SQL (T-SQL) syntax in Fabric's warehouse closely resembles SQL syntax used in SQL Server or Azure SQL Database, making it an easy transition for users familiar with these platforms.

Aggregate Measures by Dimension Attributes:

```
SELECT  dates.CalendarYear,
        dates.CalendarQuarter,
        SUM(sales.SalesAmount) AS TotalSales
FROM dbo.FactSales AS sales
JOIN dbo.DimDate AS dates ON sales.OrderDateKey = dates.DateKey
GROUP BY dates.CalendarYear, dates.CalendarQuarter
ORDER BY dates.CalendarYear, dates.CalendarQuarter;
```

Using Ranking Functions:

```
SELECT  ProductCategory,
        ProductName,
        ListPrice,
        ROW_NUMBER() OVER (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS RowNumbe
        RANK() OVER (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS Rank,
```

```
        DENSE_RANK() OVER (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS DenseRa
        NTILE(4) OVER (PARTITION BY ProductCategory ORDER BY ListPrice DESC) AS Quartile
FROM dbo.DimProduct
ORDER BY ProductCategory;
```

14.4 SQL Query Editor in Microsoft Fabric

Fabric's SQL query editor provides an intuitive interface to create and run scripts to query data warehouses. It supports IntelliSense, debugging, and syntax highlighting.

Users can:

- Write and execute SQL queries.
- Save queries as views or tables.
- Export query results to Excel.
- Use cross-database querying within a workspace.

14.5 Visual Query Editor

For users unfamiliar with SQL, Fabric provides a Visual Query Editor with a drag-and-drop interface to simplify query building. The tool automatically generates SQL code, allowing users to:

- Drag and drop tables to build queries visually.
- Automatically generate T-SQL syntax.
- Save queries as reusable views or tables.

14.6 Using Client Tools to Query a Warehouse

Microsoft Fabric supports connections to external tools like SQL Server Management Studio (SSMS) and third-party applications via ODBC or OLE DB drivers. Users can:

- Connect to a Fabric warehouse using a SQL connection string.
- Execute queries from SSMS with Microsoft Entra ID authentication.
- Integrate Fabric with business intelligence tools like Power BI.

14.7 Conclusion

Microsoft Fabric Data Warehouse integrates SQL, AI, and data transformation capabilities, enabling efficient data management and analytics. With ETL pipelines, real-time dashboards, and automated dataflows, Fabric simplifies data engineering while maintaining security and scalability.

15 Microsoft Fabric Data Warehouse v4

15.1 Introduction

A data warehouse is often central to enterprise analysis and reporting, making it a critical business asset. Monitoring a data warehouse helps track and manage costs, identify and resolve query performance issues, and gain insights into data usage.

Microsoft Fabric's Data Warehouse, powered by Synapse Analytics, offers a rich set of features that simplify data management and analysis. It includes advanced query processing capabilities and supports full transactional T-SQL functionalities like an enterprise data warehouse.

15.2 Monitoring Capacity Metrics

Microsoft Fabric operates based on capacity units (CUs), which are used for data read and write activities. The Fabric Capacity Metrics app allows administrators to monitor capacity utilization, track warehouse activity, and optimize performance.

15.3 Monitoring Current Activity

Dynamic Management Views (DMVs) provide insights into the real-time state of the data warehouse:

- **sys.dm**$_e$*xec*$_c$*onnections* : *Retrievesconnectiondetails*.**sys.dm**$_e$*xec*$_s$*essions* : *Listsauthenticatedsessions*.**sys.**
  Example query to track long-running requests:

- ```
  SELECT sessions.session_id, sessions.login_name,
      connections.client_net_address,
      requests.command, requests.start_time, requests.total_elapsed_time
  FROM sys.dm_exec_connections AS connections
  INNER JOIN sys.dm_exec_sessions AS sessions
      ON connections.session_id=sessions.session_id
  INNER JOIN sys.dm_exec_requests AS requests
      ON requests.session_id = sessions.session_id
  WHERE requests.status = 'running'
      AND requests.database_id = DB_ID()
  ORDER BY requests.total_elapsed_time DESC;
  ```

15.4 Monitoring Queries with Query Insights

Fabric provides query insights through system views, helping analyze query performance and optimize frequently run or long-running queries:

- **queryinsights.exec**$_r$*equests*$_h$*istory* : *DetailscompletedSQLqueries*.**queryinsights.long**$_r$*unning*$_q$*ueries* : *Id*
  Example query to track frequently executed queries:
  ```

- ```
  SELECT last_run_command, number_of_runs, number_of_successful_runs, number_of_failed_runs
  FROM queryinsights.frequently_run_queries
  ORDER BY number_of_runs DESC;
  ```

15.5 SQL Query Editor in Microsoft Fabric

Fabric's SQL query editor provides an intuitive interface to create and run scripts to query data warehouses. It supports IntelliSense, debugging, and syntax highlighting.

Users can:

- Write and execute SQL queries.
- Save queries as views or tables.
- Export query results to Excel.
- Use cross-database querying within a workspace.

15.6 Visual Query Editor

For users unfamiliar with SQL, Fabric provides a Visual Query Editor with a drag-and-drop interface to simplify query building. The tool automatically generates SQL code, allowing users to:

- Drag and drop tables to build queries visually.
- Automatically generate T-SQL syntax.
- Save queries as reusable views or tables.

15.7 Using Client Tools to Query a Warehouse

Microsoft Fabric supports connections to external tools like SQL Server Management Studio (SSMS) and third-party applications via ODBC or OLE DB drivers. Users can:

- Connect to a Fabric warehouse using a SQL connection string.
- Execute queries from SSMS with Microsoft Entra ID authentication.
- Integrate Fabric with business intelligence tools like Power BI.

15.8 Conclusion

Microsoft Fabric Data Warehouse integrates SQL, AI, and data transformation capabilities, enabling efficient data management and analytics. With ETL pipelines, real-time dashboards, and automated dataflows, Fabric simplifies data engineering while maintaining security and scalability.
  ```

16 Microsoft Fabric Data Warehouse v5

16.1 Introduction

A data warehouse is often central to enterprise analysis and reporting, making it a critical business asset. Monitoring a data warehouse helps track and manage costs, identify and resolve query performance issues, and gain insights into data usage.

Microsoft Fabric's Data Warehouse, powered by Synapse Analytics, offers a rich set of features that simplify data management and analysis. It includes advanced query processing capabilities and supports full transactional T-SQL functionalities like an enterprise data warehouse.

16.2 Security Features in Microsoft Fabric

Fabric provides multiple layers of security to protect data within the warehouse and SQL analytics endpoints. Key security features include:

- extbfWorkspace Roles: Controls access at the workspace level (Admin, Member, Contributor, Viewer).
- extbfItem Permissions: Grants direct permissions on individual warehouses.
- extbfData Protection: Implements object-level, column-level, and row-level security, along with dynamic data masking.

16.3 Dynamic Data Masking (DDM)

Dynamic Data Masking (DDM) obscures sensitive data for nonprivileged users while keeping the actual data intact. Masking rules can be applied at the column level:

- extbfDefault Masking: Fully masks values based on data type.
- extbfEmail Masking: Shows the first letter and a constant domain.
- extbfCustom Text Masking: Exposes specific characters while replacing others.
- extbfRandom Masking: Replaces numeric values with random numbers.

Example T-SQL to mask sensitive data:

```
ALTER TABLE Customers
ALTER COLUMN Email ADD MASKED WITH (FUNCTION = 'email()');
ALTER TABLE Customers
ALTER COLUMN CreditCardNumber ADD MASKED WITH (FUNCTION = 'partial(0,"XXXX-XXXX-XXXX-",4)');
```

16.4 Row-Level Security (RLS)

Row-Level Security (RLS) restricts access to table rows based on user identity. It is implemented using security predicates:

- extbfFilter Predicates: Restricts access to specific rows.
- extbfSecurity Policies: Enforces row-level access rules.

Example RLS implementation:

```
CREATE FUNCTION sec.tvf_SecurityPredicatebyTenant(@TenantName AS NVARCHAR(50))
RETURNS TABLE
WITH SCHEMABINDING
AS
RETURN SELECT 1 AS result WHERE @TenantName = USER_NAME() OR USER_NAME() = 'TenantAdmin';
```

16.5 Column-Level Security (CLS)

Column-Level Security (CLS) allows granular control over column access, ensuring that only authorized users can view sensitive fields. Example configuration:

```
DENY SELECT ON dbo.Patients (MedicalHistory) TO Receptionist;
DENY SELECT ON dbo.Patients (MedicalHistory) TO Patient;
```

16.6 SQL Granular Permissions

Fabric enables fine-grained access control through SQL permissions:

- extbfSELECT: Grants read access.
- extbfINSERT/UPDATE/DELETE: Controls modification access.
- extbfALTER/CONTROL: Allows schema modifications and full object control.

Example:

```
GRANT SELECT ON dbo.Sales TO SalesAnalyst;
DENY UPDATE ON dbo.Sales TO SalesAnalyst;
```

16.7 SQL Query Editor in Microsoft Fabric

Fabric's SQL query editor provides an intuitive interface to create and run scripts to query data warehouses. It supports IntelliSense, debugging, and syntax highlighting.

Users can:

- Write and execute SQL queries.
- Save queries as views or tables.
- Export query results to Excel.
- Use cross-database querying within a workspace.

16.8 Visual Query Editor

For users unfamiliar with SQL, Fabric provides a Visual Query Editor with a drag-and-drop interface to simplify query building. The tool automatically generates SQL code, allowing users to:

- Drag and drop tables to build queries visually.
- Automatically generate T-SQL syntax.
- Save queries as reusable views or tables.

16.9 Using Client Tools to Query a Warehouse

Microsoft Fabric supports connections to external tools like SQL Server Management Studio (SSMS) and third-party applications via ODBC or OLE DB drivers. Users can:

- Connect to a Fabric warehouse using a SQL connection string.
- Execute queries from SSMS with Microsoft Entra ID authentication.
- Integrate Fabric with business intelligence tools like Power BI.

16.10 Conclusion

Microsoft Fabric Data Warehouse integrates SQL, AI, and data transformation capabilities, enabling efficient data management and analytics. With ETL pipelines, real-time dashboards, and automated dataflows, Fabric simplifies data engineering while maintaining security and scalability.

17 Implement continuous integration and continuous delivery (CI/CD) in Microsoft Fabric

17.1 Introduction

A data warehouse is often central to enterprise analysis and reporting, making it a critical business asset. Monitoring a data warehouse helps track and manage costs, identify and resolve query performance issues, and gain insights into data usage.

Microsoft Fabric's Data Warehouse, powered by Synapse Analytics, offers a rich set of features that simplify data management and analysis. It includes advanced query processing capabilities and supports full transactional T-SQL functionalities like an enterprise data warehouse.

17.2 Security Features in Microsoft Fabric

Fabric provides multiple layers of security to protect data within the warehouse and SQL analytics endpoints. Key security features include:

- **Workspace Roles**: Controls access at the workspace level (Admin, Member, Contributor, Viewer).
- **Item Permissions**: Grants direct permissions on individual warehouses.
- **Data Protection**: Implements object-level, column-level, and row-level security, along with dynamic data masking.

17.3 Dynamic Data Masking (DDM)

Dynamic Data Masking (DDM) obscures sensitive data for nonprivileged users while keeping the actual data intact. Masking rules can be applied at the column level:

- **Default Masking**: Fully masks values based on data type.
- **Email Masking**: Shows the first letter and a constant domain.
- **Custom Text Masking**: Exposes specific characters while replacing others.
- **Random Masking**: Replaces numeric values with random numbers.

Example T-SQL to mask sensitive data:

```
ALTER TABLE Customers
ALTER COLUMN Email ADD MASKED WITH (FUNCTION = 'email()');
ALTER TABLE Customers
ALTER COLUMN CreditCardNumber ADD MASKED WITH (FUNCTION = 'partial(0,"XXXX-XXXX-XXXX-",4)');
```

17.4 Row-Level Security (RLS)

Row-Level Security (RLS) restricts access to table rows based on user identity. It is implemented using security predicates:

- **Filter Predicates**: Restricts access to specific rows.
- **Security Policies**: Enforces row-level access rules.

Example RLS implementation:

```
CREATE FUNCTION sec.tvf_SecurityPredicatebyTenant(@TenantName AS NVARCHAR(50))
RETURNS TABLE
WITH SCHEMABINDING
AS
RETURN SELECT 1 AS result WHERE @TenantName = USER_NAME() OR USER_NAME() = 'TenantAdmin';
```

17.5 Column-Level Security (CLS)

Column-Level Security (CLS) allows granular control over column access, ensuring that only authorized users can view sensitive fields. Example configuration:

```
DENY SELECT ON dbo.Patients (MedicalHistory) TO Receptionist;
DENY SELECT ON dbo.Patients (MedicalHistory) TO Patient;
```

17.6 SQL Granular Permissions

Fabric enables fine-grained access control through SQL permissions:

- **SELECT**: Grants read access.
- **INSERT/UPDATE/DELETE**: Controls modification access.
- **ALTER/CONTROL**: Allows schema modifications and full object control.

Example:

```
GRANT SELECT ON dbo.Sales TO SalesAnalyst;
DENY UPDATE ON dbo.Sales TO SalesAnalyst;
```

17.7 Continuous Integration and Continuous Delivery (CI/CD)

Microsoft Fabric supports CI/CD to streamline the deployment and version control of data warehouse assets. The process includes:

- **Continuous Integration (CI)**: Developers frequently commit changes to a shared repository, reducing integration issues.
- **Continuous Delivery (CD)**: Changes are deployed to a staging environment for automated testing before release.
- **Continuous Deployment**: Automated deployment of tested updates to production.

17.8 Git Integration in Microsoft Fabric

Fabric supports version control through GitHub and Azure DevOps, allowing teams to track and manage changes to workspaces efficiently.

- Developers can work in isolated branches before merging changes into the main branch.
- Integration with Git ensures that changes can be rolled back if necessary.
- Pull requests facilitate code review before merging into production.

Example Git workflow:

```
1. Create a new branch for feature development.
2. Commit changes and push to the remote repository.
3. Issue a Pull Request to merge into the main branch.
4. Deploy the updated branch to Fabric.
```

17.9 Deployment Pipelines

Deployment pipelines in Fabric help automate the movement of content across environments (Development, Test, and Production).

- **Stage-based Deployment**: Move changes from development to production systematically.
- **Automated Testing**: Ensures data integrity and performance before promoting changes.
- **Git Branch Integration**: Deployments can be synchronized with specific Git branches.

Example deployment steps:

```
1. Assign workspaces to different pipeline stages.
2. Deploy content from Development to Test.
3. Perform automated tests.
4. Deploy tested content to Production.
```

17.10 Automating CI/CD Using Fabric REST APIs

Microsoft Fabric provides REST APIs for automating CI/CD processes, improving efficiency and reducing manual effort.

- **Commit and update changes**: Automate Git commits and workspace updates.
- **List deployment pipeline items**: Retrieve a list of staged items.
- **Deploy pipeline stage content**: Automate the deployment of tested content to production.

Example API calls:

```
POST /fabric/git/commit
POST /fabric/deployment/deploy-stage
```

17.11 Conclusion

Microsoft Fabric Data Warehouse integrates SQL, AI, and data transformation capabilities, enabling efficient data management and analytics. With ETL pipelines, real-time dashboards, automated dataflows, CI/CD, and deployment pipelines, Fabric simplifies data engineering while maintaining security and scalability.

18 Monitor activities in Microsoft Fabric

18.1 Introduction

A data warehouse is often central to enterprise analysis and reporting, making it a critical business asset. Monitoring a data warehouse helps track and manage costs, identify and resolve query performance issues, and gain insights into data usage.

Microsoft Fabric's Data Warehouse, powered by Synapse Analytics, offers a rich set of features that simplify data management and analysis. It includes advanced query processing capabilities and supports full transactional T-SQL functionalities like an enterprise data warehouse.

18.2 Security Features in Microsoft Fabric

Fabric provides multiple layers of security to protect data within the warehouse and SQL analytics endpoints. Key security features include:

- **Workspace Roles**: Controls access at the workspace level (Admin, Member, Contributor, Viewer).
- **Item Permissions**: Grants direct permissions on individual warehouses.
- **Data Protection**: Implements object-level, column-level, and row-level security, along with dynamic data masking.

18.3 Dynamic Data Masking (DDM)

Dynamic Data Masking (DDM) obscures sensitive data for nonprivileged users while keeping the actual data intact. Masking rules can be applied at the column level:

- **Default Masking**: Fully masks values based on data type.
- **Email Masking**: Shows the first letter and a constant domain.
- **Custom Text Masking**: Exposes specific characters while replacing others.
- **Random Masking**: Replaces numeric values with random numbers.

Example T-SQL to mask sensitive data:

```
ALTER TABLE Customers
ALTER COLUMN Email ADD MASKED WITH (FUNCTION = 'email()');
ALTER TABLE Customers
ALTER COLUMN CreditCardNumber ADD MASKED WITH (FUNCTION = 'partial(0,"XXXX-XXXX-XXXX-",4)');
```

18.4 Row-Level Security (RLS)

Row-Level Security (RLS) restricts access to table rows based on user identity. It is implemented using security predicates:

- **Filter Predicates**: Restricts access to specific rows.
- **Security Policies**: Enforces row-level access rules.

Example RLS implementation:

```
CREATE FUNCTION sec.tvf_SecurityPredicatebyTenant(@TenantName AS NVARCHAR(50))
RETURNS TABLE
WITH SCHEMABINDING
AS
RETURN SELECT 1 AS result WHERE @TenantName = USER_NAME() OR USER_NAME() = 'TenantAdmin';
```

18.5 Column-Level Security (CLS)

Column-Level Security (CLS) allows granular control over column access, ensuring that only authorized users can view sensitive fields. Example configuration:

```
DENY SELECT ON dbo.Patients (MedicalHistory) TO Receptionist;
DENY SELECT ON dbo.Patients (MedicalHistory) TO Patient;
```

18.6 SQL Granular Permissions

Fabric enables fine-grained access control through SQL permissions:

- **SELECT**: Grants read access.
- **INSERT/UPDATE/DELETE**: Controls modification access.
- **ALTER/CONTROL**: Allows schema modifications and full object control.

Example:

```
GRANT SELECT ON dbo.Sales TO SalesAnalyst;
DENY UPDATE ON dbo.Sales TO SalesAnalyst;
```

18.7 Continuous Integration and Continuous Delivery (CI/CD)

Microsoft Fabric supports CI/CD to streamline the deployment and version control of data warehouse assets. The process includes:

- **Continuous Integration (CI)**: Developers frequently commit changes to a shared repository, reducing integration issues.
- **Continuous Delivery (CD)**: Changes are deployed to a staging environment for automated testing before release.
- **Continuous Deployment**: Automated deployment of tested updates to production.

18.8 Git Integration in Microsoft Fabric

Fabric supports version control through GitHub and Azure DevOps, allowing teams to track and manage changes to workspaces efficiently.

- Developers can work in isolated branches before merging changes into the main branch.
- Integration with Git ensures that changes can be rolled back if necessary.
- Pull requests facilitate code review before merging into production.

Example Git workflow:

```
1. Create a new branch for feature development.
2. Commit changes and push to the remote repository.
3. Issue a Pull Request to merge into the main branch.
4. Deploy the updated branch to Fabric.
```

18.9 Monitoring in Microsoft Fabric

Microsoft Fabric provides built-in monitoring capabilities to ensure data pipeline health and performance.

- **Monitor Hub**: Centralized dashboard for tracking data ingestion, transformation, and job execution.
- **Fabric Activity Logs**: Stores historical execution details to analyze trends and troubleshoot issues.
- **Alerting and Notifications**: Detects anomalies and triggers predefined actions.

18.10 Using Microsoft Fabric Activator

Activator in Fabric is used to define rules and trigger automated actions based on event-driven data.

- **Event Processing**: Detects patterns and anomalies in real-time streams.
- **Automated Actions**: Can trigger Power Automate workflows, emails, or other Fabric processes.
- **Integration with EventStreams**: Processes streaming data dynamically.

Example use case:

```
Trigger an alert when a warehouse temperature sensor reports a reading above the threshold.
```

18.11 Conclusion

Microsoft Fabric Data Warehouse integrates SQL, AI, and data transformation capabilities, enabling efficient data management and analytics. With ETL pipelines, real-time dashboards, automated dataflows, CI/CD, deployment pipelines, and monitoring tools like Monitor Hub and Activator, Fabric simplifies data engineering while maintaining security and scalability.

data access in Microsoft Fabric

18.12 Introduction

A data warehouse is often central to enterprise analysis and reporting, making it a critical business asset. Monitoring a data warehouse helps track and manage costs, identify and resolve query performance issues, and gain insights into data usage.

Microsoft Fabric's Data Warehouse, powered by Synapse Analytics, offers a rich set of features that simplify data management and analysis. It includes advanced query processing capabilities and supports full transactional T-SQL functionalities like an enterprise data warehouse.

18.13 Security in Microsoft Fabric

Security in Microsoft Fabric is optimized for specific use cases. Different users need varying levels of access to perform their job responsibilities. Fabric facilitates secure data access by implementing workspace and item permissions, compute permissions, and OneLake data access roles.

18.13.1 Fabric Security Model

Fabric's multi-layered security model consists of: eginitemize

Microsoft Entra ID Authentication: Ensures identity verification.

Fabric Access: Determines if a user can access Fabric.

Data Security: Evaluates whether a user has the necessary permissions for specific tables or files.

18.13.2 Workspace and Item Permissions

Workspaces allow collaboration on Fabric data items such as lakehouses and warehouses. Permissions are managed through: eginitemize

Workspace Roles: Defines user roles (Admin, Member, Contributor, Viewer) for workspace-wide access.

Item Permissions: Grants granular access to individual Fabric items.
Permissions can be adjusted by selecting "Manage Access" in the Fabric UI.

18.13.3 Granular Permissions

When workspace roles or item permissions are insufficient, additional security features apply: eginitemize

SQL Analytics Endpoint: Grants fine-grained permissions using GRANT, DENY, and REVOKE commands.

OneLake Data Access Roles: Secures specific folders in OneLake with role-based access control.

Warehouse Permissions: Restricts access to warehouse objects via SQL security policies.

Semantic Model Security: Implements row-level security (RLS) for Power BI reports.

18.14 Continuous Integration and Continuous Delivery (CI/CD)

Microsoft Fabric supports CI/CD to streamline deployment and version control of data warehouse assets. The process includes: eginitemize

Continuous Integration (CI): Developers frequently commit changes to a shared repository, reducing integration issues.

Continuous Delivery (CD): Changes are deployed to a staging environment for automated testing before release.

Continuous Deployment: Automated deployment of tested updates to production.

18.15 Git Integration in Microsoft Fabric

Fabric supports version control through GitHub and Azure DevOps, allowing teams to track and manage changes to workspaces efficiently. eginitemize

Developers can work in isolated branches before merging changes into the main branch.

Integration with Git ensures that changes can be rolled back if necessary.

Pull requests facilitate code review before merging into production.

18.16 Monitoring in Microsoft Fabric

Microsoft Fabric provides built-in monitoring capabilities to ensure data pipeline health and performance. eginitemize

Monitor Hub: Centralized dashboard for tracking data ingestion, transformation, and job execution.

Fabric Activity Logs: Stores historical execution details to analyze trends and troubleshoot issues.

Alerting and Notifications: Detects anomalies and triggers predefined actions.

18.17 Using Microsoft Fabric Activator

Activator in Fabric is used to define rules and trigger automated actions based on event-driven data. eginitemize

Event Processing: Detects patterns and anomalies in real-time streams.

Automated Actions: Can trigger Power Automate workflows, emails, or other Fabric processes.

Integration with EventStreams: Processes streaming data dynamically.

18.18 Conclusion

Microsoft Fabric Data Warehouse integrates SQL, AI, and data transformation capabilities, enabling efficient data management and analytics. With ETL pipelines, real-time dashboards, automated dataflows, CI/CD, deployment pipelines, and monitoring tools like Monitor Hub and Activator, Fabric simplifies data engineering while maintaining security and scalability.

19 Administer a Microsoft Fabric environment

19.1 Introduction

Administering a Microsoft Fabric environment involves a range of tasks that are essential for ensuring the efficient and effective use of the Fabric platform within an organization.

As a Fabric administrator (admin), you need to know:

- Fabric architecture
- Security and governance features
- Analytics capabilities
- Various deployment and licensing options

You also need to be familiar with the Fabric admin portal and other administrative tools and be able to configure and manage the Fabric environment to meet the needs of your organization.

Fabric admins work with business users, data analysts, and IT professionals to deploy and use Fabric to meet business objectives and comply with organizational policies and standards.

19.2 Understanding the Fabric Architecture

Microsoft Fabric is a Software-as-a-Service platform, which provides a simple and integrated approach while reducing administrative overhead. Fabric provides an all-in-one analytics solution for enterprises that covers everything from data movement to data science, real-time analytics, and business intelligence. It offers a comprehensive suite of services, including:

- Data warehousing
- Data engineering
- Data integration
- Data science
- Real-time intelligence
- Business intelligence

All data in Fabric is stored in OneLake, which is built on Azure Data Lake Storage (ADLS) Gen2 architecture. OneLake is hierarchical in nature to simplify management across your organization. There is only one OneLake per tenant, and it provides a unified namespace that spans users, regions, and even clouds.

19.3 Fabric Administrator Role and Responsibilities

Fabric administrators manage the platform through a range of tools, including:

- The Fabric admin portal
- Microsoft 365 admin center
- Microsoft 365 Security and Microsoft Purview compliance portal
- Microsoft Entra ID in the Azure portal
- PowerShell cmdlets
- Administrative APIs and SDKs

Key tasks of a Fabric administrator include:

- **Security and Access Control**: Managing user access and role-based access control (RBAC).
- **Data Governance**: Monitoring system usage and enforcing compliance policies.
- **Customization and Configuration**: Setting up data gateways, securing inbound and outbound connectivity, and defining data classification policies.
- **Monitoring and Optimization**: Ensuring optimal performance, managing capacity, and troubleshooting issues.

19.4 Managing Fabric Security

Fabric security is critical for maintaining data integrity and compliance. Admins are responsible for:

- Assigning and managing licenses in the Microsoft 365 admin center.
- Controlling access to workspaces and data items via workspace roles and item permissions.
- Enforcing sharing policies to restrict unauthorized content distribution.

19.5 Data Governance in Fabric

Fabric includes built-in governance features to ensure secure data management. Key governance features include:

- **Endorsement**: Promoting or certifying content as trusted.
- **Scanner API**: Scanning metadata for sensitive information.
- **Data Lineage**: Tracking data movement and dependencies.

19.6 Conclusion

Microsoft Fabric provides a comprehensive solution for managing enterprise data with security, governance, and administrative tools. By leveraging Fabric's features, organizations can optimize data workflows, ensure compliance, and improve analytics efficiency.