What is the main purpose of Dataflows Gen2 in Microsoft Fabric?

Answer:

Dataflows Gen2 in Microsoft Fabric are used to ingest, transform, and prepare data from multiple sources into a clean, standardized format before landing it into a destination such as a Lakehouse, Warehouse, or Power BI semantic model. They support reusable data transformation logic and simplify data ingestion for analytics.

Why are Dataflows Gen2 important in an end-to-end analytics solution?

Answer:
Dataflows Gen2 provides the data ingestion and transformation layer in an end-to-end analytics architecture. They:

- Connect to multiple data sources.
- Clean and transform raw data.
- Standardize schemas across sources.
- Load data into a centralized destination for reporting and analytics.
- Enable reuse of transformation logic.

Without them, manual extraction and transformation would be inefficient and error-prone.

---

3 In a global retail company, why is data standardization critical?

Answer:
Because different stores may:

- Use different systems
- Use different naming conventions
- Store data in different formats (currency, date format, product codes)

Dataflows Gen2 standardize this data so that:

- Sales data from all regions can be combined
- Metrics are calculated consistently

- A unified semantic model can be created for reporting

What role do Dataflows Gen2 play in building a semantic model?

Answer:
Dataflows Gen2:

- Prepare and cleanse raw store data
- Transform it into a consistent schema
- Stage it in a destination (e.g., Lakehouse/Warehouse)
- Serve as a reusable data source for Power BI

This ensures the semantic model is built on clean, trusted, consistent data.

How do Dataflows Gen2 reduce operational complexity?

Answer:
They:

- Eliminate manual extraction processes
- Automate transformations
- Allow scheduled refresh
- Enable reuse across multiple reports
- Integrate into pipelines for orchestration

This reduces errors and improves maintainability.

What are the benefits of incorporating Dataflows Gen2 into a data pipeline?

Answer:
When incorporated into a data pipeline:

- Dataflows can run as part of a larger workflow
- You can orchestrate dependencies
- You can manage complex ingestion scenarios
- You can automate refresh across multiple systems

Pipelines add orchestration; Dataflows add transformation.

What would happen if Dataflows Gen2 were not used in this scenario?

Answer:
Without Dataflows:

- Each store's data would need manual extraction.
- Transformations would be repeated in multiple places.
- High risk of inconsistencies in reporting.
- Increased maintenance overhead.
- Longer development cycles.

How do Dataflows Gen2 improve data reuse?

Answer:
Once a Dataflow prepares and stages data:

- Multiple reports can connect to the same cleansed dataset.
- Multiple semantic models can reuse the prepared data.
- Updates to transformation logic automatically propagate to downstream reports.

This ensures consistency and scalability.

What destinations can Dataflows Gen2 load data into within Microsoft Fabric?

Answer:
Dataflows Gen2 can load data into:

- Lakehouse (Delta tables)
- Warehouse
- Other Fabric destinations
- Be used directly as a data source in Power BI

Why is Dataflows Gen2 suitable for a retail business with global stores?

Answer:
Because:

- Retail generates data from POS systems, inventory systems, online platforms, etc.
- Each region may operate independently.
- Dataflows Gen2 consolidate, standardize, and transform all store data into one unified dataset.
- This enables centralized reporting, performance tracking, and executive dashboards.

What key objective does this module aim to achieve?

Answer:
The key objective is to teach how to use Dataflows Gen2 in Microsoft Fabric to:

- Ingest data from multiple sources
- Transform and cleanse the data
- Land it in a preferred destination
- Enable consistent semantic modeling
- Support scalable analytics solutions

Q1: What is a Dataflow Gen2 in Microsoft Fabric?

A1: Dataflow Gen2 is a low-to-no-code solution within Microsoft Fabric that allows users to ingest, transform, and load data into Fabric data stores such as lakehouses, warehouses, and SQL databases. It leverages Power Query Online for transformations and can be created in Data Factory, Power BI workspace, or directly in a lakehouse.

Q2: Which Microsoft Fabric workloads support Dataflows Gen2?

A2: Dataflows Gen2 can be created in:

- Data Factory workload (focus for data ingestion)
- Power BI workspace
- Lakehouse

Q3: What is the primary benefit of using Dataflows Gen2?

A3: They provide a low-to-no-code ETL solution that allows developers to perform data transformations upstream, improving report performance and enabling easier management of data pipelines.

---

2. Power Query Online Interface

Q4: What are the main components of the Power Query Online interface in Dataflows Gen2?
A4: The main components are:

1. Power Query ribbon – contains connectors, transformations, parameters, and data destination settings.
2. Queries pane – shows the data sources/queries and allows duplicating, referencing, or disabling them.
3. Diagram view – visualizes how queries and transformations are connected.
4. Data Preview pane – previews a subset of the data and shows applied transformations interactively.
5. Query Settings pane – displays the Applied Steps, allows modification, deletion, and shows data destination options.

Q5: Name three transformations that can be performed in the Power Query ribbon.
A5: Examples include:

- Filter and sort rows
- Pivot and unpivot
- Merge and append queries

Q6: What is query folding, and how is it supported in Dataflows Gen2?

A6: Query folding is the ability to push transformations back to the source system for better performance. In Dataflows Gen2, when a source supports query folding, it can be viewed and optimized in the Query Settings pane.

---

3. Queries and Data Management

Q7: What is the difference between a query and a table in Dataflows Gen2?
A7: A query represents a data source and its transformations in Power Query Online, while a table is the resulting dataset after loading a query into a data store.

Q8: Why might you duplicate or reference a query in Dataflows Gen2?
A8: To create multiple copies of the same data for scenarios like building a star schema or splitting data into smaller tables without redoing transformations.

Q9: How can you prevent a query from being loaded into a data destination?
A9: By disabling the load of the query in the Queries pane. This is useful when a query is needed temporarily for transformations but does not need to be persisted.

---

4. Diagram View and Data Preview

Q10: What is the purpose of the Diagram View in Dataflows Gen2?
A10: It visually shows how queries are connected and the transformations applied. Each query is represented as a shape, and lines represent relationships or duplicates. This helps users understand and troubleshoot their dataflow logic.

Q11: What functionality does the Data Preview pane provide?
A11: The Data Preview pane:

- Displays a subset of data to visualize transformations
- Allows drag-and-drop column reordering
- Enables right-click actions to filter or modify columns

- Shows applied transformations for the selected query

---

5. Query Settings and Applied Steps

Q12: What information is displayed in the Query Settings pane?
A12: The Query Settings pane shows:

- Applied Steps – each transformation as a step

- Gear icons to modify steps

- Contextual menu to rename, reorder, or delete steps

- Data destination options for the query

Q13: What are the available data destination options for Dataflows Gen2 in Fabric?
A13: Data can be loaded into:

- Lakehouse

- Warehouse

- SQL database
  Additionally, Dataflows Gen2 can also load to Azure SQL Database, Azure Data Explorer, or Azure Synapse Analytics.

Q14: How can you view or edit the M code of a query?
A14: By opening the Advanced editor in the Query Settings pane, which displays the underlying M code for all applied steps.

---

6. Practical Use Cases

Q15: How does Dataflows Gen2 benefit Power BI developers?
A15: It allows developers to perform transformations upstream, reducing the load on reports, improving performance, and ensuring consistent and reusable datasets.

Q16: Name three types of data sources commonly supported in Dataflows Gen2.
A16: Examples include:

- Cloud or on-premises relational databases
- Excel or flat files
- SharePoint or Salesforce

1. What is the main purpose of Dataflows Gen2 in Microsoft Fabric?

Answer:
Dataflows Gen2 are primarily used for data ingestion and transformation. They allow users to prepare, clean, and transform data before landing it into a Fabric data store, enabling consistent and reusable data pipelines.

---

2. How can a dataflow be incorporated into a pipeline?

Answer:
A dataflow can be added as an activity in a pipeline. This allows the pipeline to orchestrate the dataflow execution along with additional activities, such as executing scripts, running notebooks, or copying data, in a specific sequence.

---

3. What is the benefit of running a dataflow via a pipeline instead of manually?

Answer:
Running a dataflow via a pipeline enables automation and scheduling, ensuring data is refreshed automatically according to triggers or schedules. This reduces manual intervention, especially for enterprise or frequently changing data.

---

4. Name three common activities that can be included in a Microsoft Fabric pipeline.

Answer:

1. Copy data
2. Incorporate Dataflow
3. Execute a script or stored procedure
   (Additional options include: Add Notebook, Get metadata)

---

5. Why are pipelines considered important in data engineering within Microsoft Fabric?

Answer:
Pipelines provide a visual orchestration of activities, ensuring tasks like data transformation, script execution, and metadata operations are executed in the correct order, supporting automation, reliability, and scalability of data processes.

---

6. What is the role of triggers in Microsoft Fabric pipelines?

Answer:
Triggers initiate pipeline execution based on a schedule, event, or condition. They allow dataflows and other pipeline activities to run automatically without manual intervention, enabling timely data refreshes.

---

7. How does integrating a dataflow into a pipeline help in enterprise scenarios?

Answer:
Integration allows automation of recurring tasks, ensuring data is always up-to-date and reducing the risk of errors from manual execution. It frees data engineers to focus on analysis, optimization, and other responsibilities.

---

8. Can pipelines in Microsoft Fabric execute scripts or stored procedures after a dataflow?

Answer:
Yes. Pipelines can execute additional activities, such as scripts or stored procedures, after a dataflow completes, enabling further data processing, calculations, or transformations.

---

9. What type of operations is a pipeline best suited for in conjunction with a dataflow?

Answer:
Pipelines are best suited for orchestration tasks such as scheduling, conditional execution, multi-step transformations, metadata operations, or running additional scripts after the dataflow completes.

---

10. Explain the difference between a standalone dataflow and a dataflow run through a pipeline.

Answer:

- Standalone Dataflow: Must be run manually and is limited to ingestion and transformation.
- Dataflow in Pipeline: Can be scheduled, automated, and orchestrated alongside other activities, offering enterprise-level automation and integration.

🔹 SECTION 1: Pipeline Activities (Core Understanding)

1. What are activities in a Microsoft Fabric pipeline?

Answer:
Activities are executable tasks within a pipeline. They define the operational steps required for data ingestion, transformation, or orchestration. Activities can be connected sequentially, and their outcomes (success, failure, or completion) determine the next step in the workflow.

---

2. How does activity outcome affect pipeline flow?

Answer:
Each activity produces an outcome:

- Success
- Failure
- Completion

These outcomes can be used to conditionally direct the next activity in the pipeline, enabling branching logic and error handling.

---

3. What are the two main categories of pipeline activities?

Answer:

1. Data Transformation Activities
2. Control Flow Activities

SECTION 2: Data Transformation Activities

4. What is a Copy Data activity?

Answer:
A Copy Data activity extracts data from a source and loads it directly to a destination without complex transformations. It is used for straightforward data movement scenarios.

---

5. What is a Data Flow (Gen2) activity?

Answer:
A Data Flow (Gen2) activity encapsulates a dataflow that applies transformations to data while transferring it from source to destination. It allows cleansing, shaping, and transforming data during ingestion.

---

6. What are examples of other data transformation activities?

Answer:
Other transformation activities include:

- Notebook activity – Runs a Spark notebook for advanced transformations.
- Stored Procedure activity – Executes SQL code in a database.
- Delete Data activity – Removes existing data before loading new data.
- Other specialized transformation activities depending on the workload.

---

7. What destinations can be configured in OneLake?

Answer:
In OneLake, the destination for data transformation activities can be configured to:

- Lakehouse
- Warehouse
- SQL Database
- Other supported storage options

SECTION 3: Control Flow Activities

8. What are control flow activities used for?

Answer:
Control flow activities manage pipeline logic. They allow implementation of:

- Loops
- Conditional branching
- Variable handling
- Parameter management

They help orchestrate complex ingestion and transformation workflows.

---

9. Why are control flow activities important in enterprise data pipelines?

Answer:
They enable complex logic, such as retry mechanisms, conditional processing, dynamic execution paths, and orchestration of multiple dependent tasks, making pipelines scalable and production-ready.

SECTION 4: Parameters and Reusability

10. What are pipeline parameters?

Answer:
Parameters are dynamic values passed into a pipeline at runtime. They allow customization of pipeline execution without modifying the pipeline structure.

---

11. Why are parameters important in pipeline design?

Answer:
Parameters increase reusability and flexibility. For example, instead of hardcoding a folder name, a parameter allows users to specify a folder dynamically each time the pipeline runs.

---

12. Give an example use case of pipeline parameterization.

Answer:
A pipeline that saves ingested data to a folder can use a folder name parameter. Each time the pipeline runs, a different folder name can be provided, making the pipeline reusable across environments or dates.

🔹 SECTION 5: Pipeline Runs and Monitoring

13. What is a pipeline run?

Answer:
A pipeline run is initiated each time a pipeline is executed, either manually (on-demand) or through a scheduled trigger.

---

14. How can a pipeline run be initiated?

Answer:
Pipeline runs can be initiated:

- On-demand via the Fabric user interface
- Automatically via scheduled execution

---

15. What is the purpose of a Run ID?

Answer:
The Run ID is a unique identifier for each pipeline execution. It allows users to:

- Monitor execution status
- Review run details
- Confirm success or failure
- Investigate configuration settings used during execution

---

16. Why is monitoring pipeline runs important?

Answer:
Monitoring ensures:

- Data was ingested successfully
- Transformations completed correctly
- Failures are identified and resolved
- Compliance and audit tracking through run history

🔹 SECTION 6: Applied Scenario-Based Questions

17. How would you design a pipeline that deletes old data before loading new data?

Answer:
Use a Delete Data activity first.
Upon successful completion, connect it to a Copy Data or Data Flow (Gen2) activity to load fresh data. The success outcome of the delete activity directs execution to the next step.

---

18. How would you implement conditional logic in a pipeline?

Answer:
Use control flow activities to evaluate conditions and branch execution paths based on activity outcomes (success, failure, completion).

---

19. How can you make a pipeline reusable across multiple environments?

Answer:
By parameterizing elements such as:

- Folder paths
- Table names
- File names
- Environment-specific configurations

---

20. How do data transformation and control flow activities work together?

Answer:
Data transformation activities perform the actual data movement and transformation, while control flow activities orchestrate when and how those transformations occur, including sequencing, branching, looping, and conditional execution.

1️⃣ What is the primary purpose of the Copy Data activity?

Answer:
The Copy Data activity is used to ingest data from a supported external source into a destination such as a lakehouse file or table in Microsoft Fabric. It enables direct data movement without applying transformations during the copy process.

2️⃣ What is a common pipeline design pattern using Copy Data?

Answer:
A common pipeline design includes:

1. Delete Data activity – Removes existing data.
2. Copy Data activity – Copies fresh data from an external source.
3. Notebook activity – Runs Spark code to transform and load data into a table.

This creates a repeatable data ingestion process.

3️⃣ When should you use the Copy Data activity?

Answer:
Use Copy Data when:

- You need to move data directly between supported source and destination.
- No transformations are required during ingestion.
- You want to land raw data first and apply transformations later in the pipeline.

4️⃣ When should you NOT use the Copy Data activity?

Answer:
Do not use Copy Data when:

- You need to apply transformations during ingestion.
- You need to merge data from multiple sources.

In such cases, use a Data Flow activity (Dataflow Gen2) instead.

5️⃣ What tool helps configure the Copy Data activity?

Answer:
A graphical configuration tool guides users through setting up the source and destination connections when the Copy Data activity is added to a pipeline.

6️⃣ What types of sources are supported by Copy Data in OneLake?

Answer:
Copy Data supports a wide range of source connections, including:

- Lakehouse
- Warehouse

- SQL Database
- Other common supported data sources

This allows ingestion from most typical enterprise systems.

7️⃣ Where can you edit the Copy Data activity settings after adding it to a pipeline?

Answer:
After adding the Copy Data activity to the pipeline canvas, you can select it and edit its settings in the configuration pane located underneath the canvas.

8️⃣ Can Copy Data activity perform data transformations?

Answer:
No. Copy Data does not perform transformations. It strictly moves data. Transformations must be handled in:

- A Notebook activity (using Spark), or
- A Data Flow activity (Dataflow Gen2).

9️⃣ What is the benefit of combining Copy Data with other activities?

Answer:
Combining Copy Data with activities like Delete Data and Notebook enables:

- Automated ingestion workflows
- Data refresh patterns
- Structured data processing pipelines
- Reusable and repeatable ingestion processes

🔟 What activity should be used for complex transformations in Microsoft Fabric?

Answer:
Use the Data Flow activity (Dataflow Gen2) when:

- Multiple transformation steps are required
- Data must be merged from multiple sources
- Power Query transformations are needed

1. What is the main purpose of pipeline templates in Microsoft Fabric?

Answer:
Pipeline templates in Microsoft Fabric provide predefined pipeline structures for common data ingestion and transformation scenarios. They help users quickly implement standard workflows without building pipelines from scratch, while still allowing customization to meet specific business requirements.

---

2. Why would you use a predefined pipeline template instead of creating a pipeline from scratch?

Answer:
Using a predefined pipeline template:

- Reduces development time
- Ensures best-practice design patterns
- Minimizes configuration errors
- Provides a structured starting point for common scenarios

Templates accelerate implementation while still allowing full customization in the pipeline canvas.

---

3. Where do you access pipeline templates in Microsoft Fabric?

Answer:
Pipeline templates are accessed by:

1. Creating a new pipeline.
2. Selecting the Templates tile in the "Choose a task to start" section.
3. Choosing the appropriate predefined template from the list displayed.

---

4. Can pipeline templates be customized after selection?

Answer:
Yes. After selecting a template, the pipeline opens in the pipeline canvas, where you can:

- Add or remove activities
- Modify data sources and destinations
- Adjust transformation logic
- Configure parameters and triggers

Templates provide a foundation, but full customization is supported.

---

5. What is the advantage of using templates for common pipeline scenarios?

Answer:
Templates are optimized for frequently used data integration patterns, such as:

- Data ingestion from multiple sources
- Data transformation workflows
- Data movement between systems

This ensures efficiency, consistency, and alignment with Microsoft Fabric best practices.

---

6. How do pipeline templates support custom data ingestion processes?

Answer:
Although templates provide predefined activity flows, users can:

- Combine different activities
- Modify transformation logic
- Integrate additional orchestration steps

This flexibility allows organizations to tailor ingestion and transformation pipelines to their unique requirements.

---

7. What happens after selecting a pipeline template?

Answer:
After selecting a template:

- The pipeline is automatically generated.
- It opens in the pipeline canvas.
- You can edit, configure, and extend the activities as needed.
- The pipeline can then be validated, published, and triggered.

---

8. How do pipeline templates improve productivity in Microsoft Fabric?

Answer:
They:

- Eliminate repetitive configuration work
- Provide structured workflow design
- Accelerate deployment
- Allow focus on business logic instead of setup

This significantly reduces implementation time for standard ingestion and transformation use cases.

---

9. Can pipeline templates be used for complex orchestration?

Answer:
Yes. Templates can serve as a starting point for complex orchestration. Users can add conditional activities, loops, dependencies, and additional data transformation steps within the pipeline canvas.

---

10. What is the key difference between custom pipelines and template-based pipelines?

Answer:

Custom Pipeline	Template-Based Pipeline
Built entirely from scratch	Predefined structure
Fully manual design	Pre-configured workflow
More setup time required	Faster implementation
Complete flexibility	Flexible but structured starting point

1. What should you do before running a pipeline in Microsoft Fabric?

Answer:
You should use the Validate option to ensure the pipeline configuration is correct and free of errors before running it.

---

2. What is the purpose of the Validate option in a pipeline?

Answer:
The Validate option checks whether the pipeline activities, connections, parameters, and dependencies are properly configured. It helps detect configuration errors before execution.

---

3. After validation, what are the two ways to execute a pipeline?

Answer:
After validation, you can:

1. Run the pipeline interactively (manual run).
2. Configure a schedule to run it automatically at defined intervals.

---

4. When would you use an interactive run?

Answer:
An interactive run is used when:

- Testing a pipeline.
- Running ad-hoc processes.
- Verifying recent changes before scheduling.
- Troubleshooting issues.

---

5. When should you use a scheduled run?

Answer:
A scheduled run should be used when:

- The pipeline needs to run automatically.
- There are recurring data loads (e.g., daily, hourly).
- You want consistent and automated execution without manual intervention.

---

6. Where can you view pipeline run history?

Answer:
You can view pipeline run history:

- From the pipeline canvas.
- From the pipeline item listed in the workspace page.

---

7. What information can you obtain from run history?

Answer:
Run history provides:

- Execution status (Succeeded, Failed, In Progress).
- Start and end time.
- Duration of execution.
- Error details (if failed).

---

8. Why is monitoring pipeline runs important?

Answer:
Monitoring ensures:

- Pipelines are running as expected.
- Failures are detected quickly.
- Performance issues are identified.
- Data processes remain reliable and consistent.

---

9. How can run history help in troubleshooting?

Answer:
Run history helps by:

- Showing which activity failed.
- Providing error messages.
- Identifying execution duration issues.
- Highlighting patterns of recurring failures.

---

10. What is the benefit of viewing run history from the workspace page?

Answer:
Viewing run history from the workspace page allows you to:

- Monitor multiple pipelines centrally.
- Compare execution performance.
- Quickly identify failed or delayed pipelines.
-

1️⃣ What is the primary purpose of a Lakehouse in Microsoft Fabric?

Answer:
A Lakehouse in Microsoft Fabric combines data lake storage (OneLake) with structured table capabilities using Delta format. It enables storing raw files and structured tables in the same environment for analytics and Spark processing.

2️⃣ Where is Lakehouse data physically stored in Fabric?

Answer:
Lakehouse data is stored in OneLake, which is the unified storage layer in Microsoft Fabric.

3️⃣ Why must Fabric capacity (Trial, Premium, or Fabric) be enabled when creating a workspace?

Answer:
Fabric capacity is required because Spark workloads, pipelines, and lakehouse operations depend on Fabric compute resources. Without Fabric capacity, data engineering features are unavailable.

4️⃣ Why was the “Lakehouse schemas (Public Preview)” option disabled in this lab?

Answer:
Because the lab focuses on basic table creation and ingestion without advanced schema management features.

✅ SECTION 2: Pipeline & Copy Data Activity

5️⃣ What is the purpose of a Copy Data activity in a Fabric pipeline?

Answer:
The Copy Data activity extracts data from a source system and loads it into a destination (e.g., Lakehouse files). It supports ETL and ELT ingestion patterns.

6️⃣ In this lab, what was the source type used?

Answer:
The source type used was HTTP, connecting to:

https://raw.githubusercontent.com/MicrosoftLearning/dp-data/main/sales.csv

---

7️⃣ Why was authentication set to Anonymous?

Answer:
Because the CSV file hosted on GitHub was publicly accessible and did not require authentication.

8️⃣ What file format settings were configured for ingestion?

Answer:

- File format: DelimitedText
- Column delimiter: Comma (,)
- Row delimiter: Line feed (\n)
- First row as header: Selected
- Compression: None

9️⃣ Why was a subfolder named `new_data` created?

Answer:
To organize ingested raw files within the Lakehouse Files section and maintain structured data storage practices.

🔟 What does “Save + Run” do in the Copy Data wizard?

Answer:
It saves the pipeline configuration and immediately executes the pipeline.

✅ SECTION 3: Spark Notebook Transformation

1️⃣1️⃣ Why was a parameter cell used in the notebook?

Answer:
The parameter cell allows variables (like `table_name`) to be dynamically overridden when the notebook is executed from a pipeline.

1️⃣2️⃣ What Spark function was used to extract Year and Month?

Answer:

year(col("OrderDate"))

month(col("OrderDate"))

1️⃣3️⃣ What Spark function was used to split the customer name?

Answer:

split(col("CustomerName"), " ")

1️⃣4️⃣ Why was the data written using Delta format?

Answer:
Delta format provides:

- ACID transactions

- Schema enforcement
- Time travel
- Better performance for analytics

15️⃣ What does `.mode("append")` do?

Answer:
It appends new data to the existing table instead of overwriting it.

✅ SECTION 4: ETL Orchestration Logic

16️⃣ Why was a Delete Data activity added before Copy Data?

Answer:
To remove old CSV files before copying new ones, preventing duplicate ingestion.

17️⃣ What does wildcard file path with `*.csv` accomplish?

Answer:
It deletes all CSV files inside the specified folder.

18️⃣ Why was the Notebook activity connected after Copy Data?

Answer:
To ensure transformation only runs after data has been successfully copied (sequential orchestration).

19️⃣ What is the purpose of base parameters in the Notebook activity?

Answer:
They override notebook parameter cell values dynamically during pipeline execution.

Example used:

Name	Type	Value
table_name	String	new_sales

20️⃣ What table was created when the notebook ran from the pipeline?

Answer:
`new_sales`

✅ SECTION 5: Troubleshooting & Execution

21️⃣ Why might you receive this error?

> “Spark SQL queries are only possible in the context of a lakehouse.”

Answer:
Because the notebook is not attached to a lakehouse. You must remove and reattach the lakehouse in notebook settings.

22️⃣ Why does the first Spark execution take longer?

Answer:
Because the Spark pool must initialize when running Spark code for the first time.

✅ SECTION 6: ETL vs ELT in Fabric Context

23️⃣ In this lab, is this ETL or ELT?

Answer:
It is ELT:

- Extract → Copy to Lakehouse
- Load → Store raw CSV in Files
- Transform → Use Spark to process data into Delta table

2️⃣4️⃣ What role does Spark play in this ingestion process?

Answer:
Spark performs scalable transformations on ingested data before loading into structured Delta tables.

✅ SECTION 7: Conceptual Understanding for Exam

2️⃣5️⃣ Why combine Pipelines and Notebooks in Fabric?

Answer:
Pipelines handle orchestration and ingestion.
Notebooks handle complex transformation logic.
Together, they enable scalable, reusable ETL processes.

2️⃣6️⃣ What storage layer backs the Lakehouse?

Answer:
OneLake.

2️⃣7️⃣ What is the benefit of using Delta tables in a Lakehouse?

Answer:

- ACID compliance

- Schema validation
- Efficient incremental loads
- Optimized analytics performance

---

28 What would happen if Delete Data activity is removed?

Answer:
Duplicate CSV files may accumulate, potentially leading to duplicate data processing.

---

29 What is the difference between Files and Tables in Lakehouse?

Answer:

- Files → Raw unstructured/semi-structured data (CSV, JSON, Parquet)
- Tables → Structured Delta tables for analytics

---

30 Why is parameterization important in production ETL?

Answer:
It allows reusable pipelines and dynamic data loading without hardcoding values.

What is Apache Spark?

Answer:
Apache Spark is an open-source distributed processing framework designed for large-scale data processing. It performs in-memory computation, supports batch and streaming workloads, and provides APIs in Python, Scala, Java, and SQL.

---

2. Why is Spark popular in big data processing?

Answer:
Spark is popular because:

- It performs in-memory distributed computation

- It supports batch and real-time processing
- It integrates with data lakes and warehouses
- It supports machine learning and graph processing
- It scales horizontally across clusters

---

3. What are the core components of Spark?

Answer:
Core components include:

- Spark Core (distributed processing engine)
- Spark SQL (structured data processing)
- Spark Streaming
- MLlib (machine learning)
- GraphX

---

4. What is a Spark DataFrame?

Answer:
A DataFrame is a distributed collection of structured data organized into named columns, similar to a relational table.

---

5. What is the difference between RDD and DataFrame?

Answer:

RDD	DataFrame
Low-level API	High-level API

No schema enforcement	Schema enforced
Slower optimization	Uses Catalyst optimizer
More control	Better performance

🔹 Section 2: Spark in Microsoft Fabric

6. How is Spark used in Microsoft Fabric?

Answer:
In Microsoft Fabric, Spark is used to:

- Ingest data into a lakehouse
- Transform data
- Create Delta tables
- Run SQL queries
- Build data engineering pipelines

Spark runs natively inside Fabric workspaces.

---

7. What is a Lakehouse in Fabric?

Answer:
A Lakehouse in Fabric combines:

- Data lake storage (files)
- Data warehouse capabilities (tables, SQL)
- Spark processing engine

It stores data in Delta format and allows both Spark and SQL access.

---

8. What file format does Fabric Lakehouse use for tables?

Answer:
Fabric Lakehouse uses:

- Delta Lake format

Delta Lake provides:

- ACID transactions
- Schema enforcement
- Time travel
- Data versioning

---

9. How does Spark integrate with other Fabric services?

Answer:
Spark integrates with:

- Data pipelines
- Power BI semantic models
- Warehouse endpoints
- Notebooks
- OneLake storage

This unified environment reduces data movement.

---

10. What is OneLake in Fabric?

Answer:
OneLake is Fabric's unified data lake storage layer.
All Lakehouse and Warehouse data is stored in OneLake.

🔹 Section 3: Spark Processing in Fabric

11. How do you ingest data into a Lakehouse using Spark?

Answer:
Steps:

1. Create a Lakehouse
2. Open a Spark notebook
3. Read data using Spark (CSV, Parquet, JSON, etc.)
4. Transform data
5. Write data to Delta table

Example:

```
df = spark.read.csv("Files/sales.csv", header=True)

df.write.format("delta").saveAsTable("sales_table")
```

---

12. What is the benefit of using Spark notebooks in Fabric?

Answer:
Benefits:

- Interactive development
- Built-in cluster management
- Integration with Lakehouse
- Easy pipeline orchestration
- Shared workspace environment

---

13. What is the Catalyst Optimizer?

Answer:
Catalyst is Spark's query optimization engine that automatically improves SQL and DataFrame execution plans.

---

14. What are transformations vs actions in Spark?

Answer:

Transformations	Actions
Lazy evaluation	Trigger execution
filter(), select()	show(), collect(), write()
Define logic	Execute logic

Spark executes only when an action is called.

🔹 Section 4: Fabric-Specific Scenarios (Exam Focus)

15. Why use Spark in Fabric instead of traditional ETL tools?

Answer:

- Better for large-scale distributed processing
- Supports advanced analytics
- Handles structured and unstructured data
- Native integration with Lakehouse
- Scalable compute clusters

---

16. How does Fabric simplify Spark usage compared to Azure HDInsight?

Answer:
Compared to Azure HDInsight:

- No manual cluster provisioning
- Built-in workspace integration
- Unified governance
- Automatic storage in OneLake
- Integrated Power BI connectivity

---

17. What is the role of Spark in a Lakehouse architecture?

Answer:
Spark:

- Ingests raw data
- Cleans and transforms data
- Writes optimized Delta tables
- Enables analytics and machine learning

---

18. Can Spark and SQL access the same data in Fabric?

Answer:
Yes. Because data is stored in Delta format in OneLake:

- Spark can process it
- SQL endpoints can query it
- Power BI can visualize it

---

19. What is ACID transaction support in Delta Lake?

Answer:
ACID means:

- Atomicity
- Consistency
- Isolation
- Durability

Delta Lake ensures reliable updates and prevents data corruption.

---

20. When should you use Spark in Fabric?

Answer:
Use Spark when:

- Processing large datasets
- Performing complex transformations
- Running data engineering pipelines
- Implementing machine learning workloads
- Performing batch or streaming ingestion

🔹 Advanced Concept Questions (Likely in Professional Exam)

21. How does Spark improve performance in distributed processing?

Answer:

- In-memory computation
- Parallel execution

- DAG execution model
- Partition-based processing
- Catalyst optimization

---

22. What happens when you write a DataFrame as a Delta table?

Answer:
Spark:

1. Converts data to Parquet
2. Adds Delta transaction logs
3. Registers metadata in Lakehouse
4. Enables ACID and versioning

---

23. How does Spark support scalability in Fabric?

Answer:

- Automatic cluster scaling
- Distributed execution
- Partitioned storage
- Parallel task scheduling

🔥 High-Probability Exam Scenario Question

24. Scenario:

You need to ingest 5 million transaction records daily, clean the data, and make it available for reporting in Power BI. What should you use?

Answer:

- Spark notebook for transformation
- Write data as Delta table in Lakehouse
- Expose Lakehouse SQL endpoint
- Connect Power BI semantic model

. What is Apache Spark?

Answer:
Apache Spark is a distributed data processing framework that uses a “divide and conquer” approach to process large volumes of data by distributing tasks across multiple computers (nodes) in a cluster.

---

2. What is a Spark pool in Microsoft Fabric?

Answer:
A Spark pool is a cluster of compute nodes in Microsoft Fabric that distributes and processes Spark workloads.

---

3. What are the two types of nodes in a Spark pool?

Answer:

- Head node – Runs the driver program and coordinates distributed processes.
- Worker nodes – Run executor processes that perform actual data processing tasks.

---

4. What role does the driver program play?

Answer:
The driver program runs on the head node and coordinates distributed tasks across worker nodes.

---

5. What languages can Spark run?

Answer:
Spark supports:

- Java
- Scala
- Spark R
- Spark SQL
- PySpark

In practice, most workloads use PySpark and Spark SQL.

---

6. What storage system does Spark commonly process data from in Microsoft Fabric?

Answer:
A lakehouse based in OneLake.

🔹 Spark Pools in Microsoft Fabric

7. What is a starter pool in Microsoft Fabric?

Answer:
A pre-configured Spark pool available in each workspace that allows Spark jobs to run quickly with minimal setup.

---

8. Can Spark pool customization be restricted?

Answer:
Yes, Fabric administrators can disable customization at the Fabric Capacity level.

---

9. Where can Spark pools be managed in Fabric?

Answer:
In the Admin portal → Workspace settings → Capacity settings → Data Engineering/Science Settings.

---

10. What is Node Family in Spark pool configuration?

Answer:
Node Family specifies the type of virtual machines used in the Spark cluster. Memory-optimized nodes are typically recommended.

---

11. What is Autoscale in Spark pools?

Answer:
Autoscale automatically provisions nodes as needed, based on workload demands, with configurable initial and maximum node limits.

---

12. What is Dynamic Allocation?

Answer:
Dynamic allocation automatically adjusts the number of executor processes on worker nodes based on data volumes.

---

13. What happens if multiple Spark pools exist in a workspace?

Answer:
You can designate one as the default pool, which will be used when no specific pool is specified for a job.

🔹 Runtimes and Environments

14. What does a Spark runtime define?

Answer:
A Spark runtime defines:

- Spark version
- Delta Lake version
- Python version
- Other core software components

---

15. Why might an organization need multiple environments?

Answer:
To support diverse data processing tasks requiring different runtimes, libraries, or configurations.

---

16. What can be configured when creating a custom environment?

Answer:
You can:

- Specify Spark runtime
- Install public PyPI libraries
- Upload custom libraries
- Override Spark configuration properties
- Upload resource files
- Specify the Spark pool

---

17. Where can you set the default Spark runtime?

Answer:
In workspace settings.

---

18. After creating custom environments, what can you do?

Answer:
Set one as the default environment for the workspace.

🔹 Native Execution Engine

19. What is the native execution engine in Fabric?

Answer:
A vectorized processing engine that runs Spark operations directly on lakehouse infrastructure, improving performance for large datasets in Parquet or Delta formats.

---

20. How do you enable the native execution engine at the environment level?

Answer:
By setting:

- `spark.native.enabled: true`
- `spark.shuffle.manager: org.apache.spark.shuffle.sort.ColumnarShuffleManager`

---

21. Can native execution engine be enabled per notebook?

Answer:
Yes, by configuring Spark properties at the beginning of the notebook using `%%configure`.

🔹 High Concurrency Mode

22. What is high concurrency mode?

Answer:
A feature that allows multiple users or processes to share the same Spark session efficiently while maintaining isolation.

---

23. Why is high concurrency mode useful?

Answer:
It optimizes Spark resource usage when multiple notebooks or jobs run concurrently.

---

24. Where can high concurrency mode be enabled?

Answer:
In the Data Engineering/Science section of workspace settings.

🔹 MLFlow in Fabric

25. What is MLFlow?

Answer:
An open-source library used for managing machine learning experiments, model training, and deployment.

---

26. How does Fabric use MLFlow by default?

Answer:
Fabric automatically logs machine learning experiment activity without requiring explicit code.

---

27. Can automatic MLFlow logging be disabled?

Answer:
Yes, in workspace settings.

🔹 Spark Administration

28. Who can manage Spark settings at a higher level?

Answer:
Fabric administrators can manage Spark settings at the capacity level.

---

29. What can administrators do at the capacity level?

Answer:
They can restrict or override Spark settings in workspaces within the organization.

🎯 Exam-Focused Scenario Questions

30. If a workload requires specific Python libraries not available by default, what should you do?

Answer:
Create a custom environment and install the required libraries from PyPI or upload custom packages.

---

31. If query performance on large Delta files is slow, what should you enable?

Answer:
Enable the native execution engine.

---

32. If multiple users are running notebooks simultaneously and resources are inefficiently used, what should be enabled?

Answer:
High concurrency mode.

---

33. If you need cost control with fluctuating workloads, what Spark pool setting is most important?

Answer:
Autoscale.

---

34. If executor processes need to scale based on workload size automatically, what should be enabled?

Answer:
Dynamic allocation.

1️⃣ What are the two primary ways to run Spark code in Microsoft Fabric?

Answer:
You can run Spark code in Microsoft Fabric using:

1. Notebooks (interactive development and analysis)
2. Spark Job Definitions (automated or scheduled execution)

2️⃣ When should you use a notebook instead of a Spark job?

Answer:
Use a notebook when:

- You need interactive data exploration

- You want to test transformations step-by-step
- You need to visualize results immediately
- You are collaborating with team members

Use a Spark job when:

- You need automation
- You want scheduled execution
- The process must run without manual intervention

3️⃣ What is the purpose of Spark in Microsoft Fabric?

Answer:
Spark in Microsoft Fabric is used for:

- Data ingestion
- Data transformation
- Large-scale distributed data processing
- Analytical workloads on lakehouse data

🔹 SECTION 2: Notebooks in Microsoft Fabric

4️⃣ What is a notebook in Microsoft Fabric?

Answer:
A notebook is an interactive development environment that allows users to:

- Write and execute Spark code
- Combine code, text, and images

- Collaborate and share analytical workflows

5️⃣ What are notebook cells?

Answer:
Notebook cells are individual sections within a notebook that can contain:

- Markdown content (text, documentation, explanations)
- Executable Spark code

6️⃣ What languages can be used in Fabric notebooks?

Answer:
Fabric notebooks support multiple languages, including:

- Python (PySpark)
- Scala
- SQL
- R

7️⃣ What is the benefit of markdown cells in a notebook?

Answer:
Markdown cells allow users to:

- Document logic
- Explain transformations
- Add formatted text and images
- Improve collaboration and readability

8️⃣ What happens when you run a code cell in a notebook?

Answer:
When a code cell is executed:

- Spark processes the code
- Results are displayed immediately
- Output can include tables, logs, or visualizations

9️⃣ Why are notebooks suitable for data exploration?

Answer:
Because they:

- Allow step-by-step execution
- Provide immediate feedback
- Enable iterative testing
- Support visual inspection of data

🔹 SECTION 3: Spark Job Definition in Microsoft Fabric

🔟 What is a Spark Job Definition?

Answer:
A Spark Job Definition is a configuration that defines how a Spark script should run as an automated or scheduled job in Microsoft Fabric.

1️⃣1️⃣ When should you use a Spark Job Definition?

Answer:
Use a Spark Job Definition when:

- Running batch ingestion pipelines
- Scheduling ETL processes
- Automating data transformations
- Executing production workloads

---

12 What must be specified when creating a Spark Job Definition?

Answer:
You must specify:

- The Spark script to run
- Optional reference files (e.g., Python modules)
- The lakehouse containing the data

---

13 What is a reference file in a Spark Job Definition?

Answer:
A reference file is an external file (such as a Python script) that contains reusable functions or dependencies used by the main Spark script.

---

14 Can Spark jobs run on a schedule?

Answer:
Yes. Spark Job Definitions can be:

- Run on-demand
- Scheduled to run automatically

---

15 What is the role of the lakehouse in a Spark job?

Answer:
The lakehouse provides:

- The data source for the Spark job
- The storage location for processed results
- A structured environment for Spark processing

🔹 SECTION 4: Scenario-Based Questions (Exam Style)

16️⃣ You need to test a transformation before deploying it to production. What should you use?

Answer:
Use a Notebook because it allows interactive testing and immediate result inspection.

17️⃣ You need to run a transformation every night at 2 AM automatically. What should you use?

Answer:
Use a Spark Job Definition with a scheduled trigger.

18️⃣ You want to collaborate with teammates and document your transformation logic clearly. What should you use?

Answer:
Use a Notebook, as it supports markdown documentation and sharing.

19️⃣ You have a Python file containing reusable transformation functions. How can you use it in a Spark job?

Answer:
Attach it as a reference file in the Spark Job Definition.

---

20 What is the key difference between notebooks and Spark Job Definitions?

Answer:

Notebook	Spark Job Definition
Interactive	Automated
Manual execution	On-demand or scheduled
Used for exploration	Used for production workloads
Immediate results	Runs in background

Apache Spark Fundamentals

Q1. What is Apache Spark primarily used for?

Answer:
Apache Spark is a distributed data processing framework used for large-scale data analytics. It distributes processing tasks across multiple nodes in a cluster (Spark pool) using a divide-and-conquer approach.

Q2. In Microsoft Fabric, what is a Spark cluster called?

Answer:
A Spark cluster in Microsoft Fabric is called a Spark pool.

Q3. What are the two types of nodes in a Spark pool?

Answer:

- Head node – Runs the driver program and coordinates distributed processes.

- Worker nodes – Run executor processes that perform the actual data processing tasks.

Q4. Which languages can Spark run?

Answer:

- Java
- Scala
- Spark R
- Spark SQL
- PySpark

In practice, most workloads use PySpark and Spark SQL.

2️⃣ Spark Pools in Microsoft Fabric

Q5. What is the purpose of the starter pool in Microsoft Fabric?

Answer:
The starter pool allows Spark jobs to run quickly with minimal configuration. It is automatically available in each workspace.

Q6. What are the key configuration settings for a Spark pool?

Answer:

- Node Family (VM type; memory optimized recommended)
- Autoscale (initial and maximum nodes)
- Dynamic allocation (executor allocation based on data volume)

Q7. What does Autoscale do in a Spark pool?

Answer:
Autoscale automatically provisions additional nodes when needed and scales down when demand decreases.

Q8. What is Dynamic Allocation?

Answer:
Dynamic allocation automatically adjusts the number of executor processes based on workload size and data volume.

Q9. Where can Spark pools be managed in Microsoft Fabric?

Answer:
In Workspace Settings → Capacity Settings → Data Engineering/Science Settings.

Q10. Can administrators restrict Spark pool customization?

Answer:
Yes. Fabric administrators can disable Spark pool customization at the Fabric Capacity level.

3️⃣ Spark Runtimes & Environments

Q11. What does a Spark runtime define?

Answer:
A Spark runtime defines:

- Spark version
- Delta Lake version
- Python version
- Core software components

Q12. Why would an organization create multiple environments?

Answer:
To support different data processing tasks that require:

- Different Spark runtime versions
- Specific libraries
- Custom configuration settings

Q13. What can you configure when creating a custom environment?

Answer:

- Spark runtime version
- Install PyPI libraries
- Upload custom libraries
- Specify Spark pool
- Override Spark configuration properties
- Upload resource files

Q14. How do you set a default environment?

Answer:
After creating at least one custom environment, you can designate it as the default in workspace settings.

---

4 Native Execution Engine

Q15. What is the native execution engine in Microsoft Fabric?

Answer:
It is a vectorized processing engine that runs Spark operations directly on lakehouse infrastructure for improved performance.

Q16. When is the native execution engine most beneficial?

Answer:
When working with large datasets stored in:

- Parquet
- Delta file formats

Q17. Which Spark properties enable the native execution engine?

Answer:

spark.native.enabled: true

spark.shuffle.manager: org.apache.spark.shuffle.sort.ColumnarShuffleManager

---

Q18. Can the native execution engine be enabled at the notebook level?

Answer:
Yes. It can be enabled:

- At environment level
- Inside a specific notebook using `%%configure`

5️⃣ High Concurrency Mode

Q19. What happens when a notebook runs in Spark?

Answer:
A Spark session is initiated.

Q20. What is High Concurrency Mode?

Answer:
High concurrency mode allows multiple users or processes to share the same Spark session while maintaining isolation between notebooks.

Q21. What is the benefit of High Concurrency Mode?

Answer:

- Better resource efficiency
- Supports multiple concurrent users
- Ensures variable isolation between notebooks

Q22. Can High Concurrency Mode be enabled for Spark jobs?

Answer:
Yes. It can be enabled for both notebooks and non-interactive Spark jobs.

---

6 MLFlow Logging

Q23. What is MLFlow used for?

Answer:
MLFlow is used to manage machine learning training and model deployment, including logging experiment activity.

Q24. How does Microsoft Fabric use MLFlow?

Answer:
Fabric automatically logs machine learning experiment activity using MLFlow without requiring explicit logging code.

Q25. Can automatic MLFlow logging be disabled?

Answer:
Yes, it can be disabled in workspace settings.

---

7 Administrative Control

Q26. Who can manage Spark settings across multiple workspaces?

Answer:
Fabric capacity administrators.

Q27. What can Fabric capacity administrators do?

Answer:
They can:

- Restrict Spark pool customization
- Override Spark settings
- Control Spark configuration at capacity level

---

8 Scenario-Based Objective Questions

Q28. A data engineer needs different Python libraries for separate projects. What should they create?

Answer:
Custom environments.

Q29. A workload requires memory-intensive processing. What Node Family should be selected?

Answer:
Memory-optimized nodes.

Q30. Multiple users need to run notebooks simultaneously without resource waste. What should be enabled?

Answer:
High concurrency mode.

Q31. A data engineer wants faster performance when querying Delta tables in a lakehouse. What should they enable?

Answer:
Native execution engine.

Q32. A team wants Spark to automatically scale based on workload demand. What should they configure?

Answer:
Enable Autoscale in the Spark pool.

1. What data structure is most commonly used for structured data in Spark?

Answer:
The most commonly used data structure for structured data in Spark is the DataFrame, provided as part of the Spark SQL library.

---

2. What is Spark's native data structure?

Answer:
Spark's native data structure is the Resilient Distributed Dataset (RDD).

---

3. How are Spark DataFrames similar to Pandas DataFrames?

Answer:
Spark DataFrames are similar to Pandas DataFrames in structure and usage, but they are optimized for distributed processing in Spark's environment.

---

4. What is the purpose of `%%pyspark` in a Spark notebook?

Answer:
`%%pyspark` is a magic command that specifies the language in the notebook cell as PySpark (Python for Spark).

---

5. What magic command is used to specify Scala in a Spark notebook?

Answer:
`%%spark` is used to specify Scala.

---

6. How do you load a CSV file into a Spark DataFrame with header inference?

Answer:

```
df = spark.read.load('Files/data/products.csv',
  format='csv',
  header=True
)
```

---

7. What does `header=True` do when loading a CSV file?

Answer:
It tells Spark that the first row of the file contains column names.

---

8. Why would you specify an explicit schema instead of inferring it?

Answer:

- When the file does not contain column names
- To improve performance

- To enforce specific data types

---

9. Which classes are used to define an explicit schema in PySpark?

Answer:

- `StructType`
- `StructField`
- Data types like `IntegerType`, `StringType`, and `FloatType`

---

10. How do you define an explicit schema for a DataFrame?

Answer:

```
productSchema = StructType([
  StructField("ProductID", IntegerType()),
  StructField("ProductName", StringType()),
  StructField("Category", StringType()),
  StructField("ListPrice", FloatType())
])
```

---

11. What method is used to select specific columns from a DataFrame?

Answer:
The `select()` method.

Example:

```
df.select("ProductID", "ListPrice")
```

---

12. What shorter syntax can be used to select columns?

Answer:

df["ProductID", "ListPrice"]

---

13. What does the `select()` method return?

Answer:
It returns a new DataFrame.

---

14. How do you filter records in a DataFrame?

Answer:
By using the `where()` method.

Example:

```
df.where(df["Category"] == "Mountain Bikes")
```

---

15. How can you filter multiple conditions?

Answer:
Using logical operators like `|` (OR) and `&` (AND).

Example:

```
.where((df["Category"]=="Mountain Bikes") |
    (df["Category"]=="Road Bikes"))
```

---

16. What method is used to group data in a DataFrame?

Answer:
`groupBy()`

---

17. How do you count records per category?

Answer:

df.select("ProductID", "Category").groupBy("Category").count()

---

18. What format is recommended for saving analytical data?

Answer:
Parquet format.

---

19. Why is Parquet preferred?

Answer:

- Efficient storage format
- Optimized for analytics
- Supported by most large-scale analytics systems

---

20. How do you save a DataFrame as a Parquet file?

Answer:

bikes_df.write.mode("overwrite").parquet('Files/product_data/bikes.parquet')

---

21. What does `mode("overwrite")` do?

Answer:
It replaces any existing file with the same name.

---

22. What is partitioning in Spark?

Answer:
Partitioning is an optimization technique that improves performance by reducing unnecessary disk I/O and maximizing distributed processing.

---

23. How do you write partitioned data?

Answer:

bikes_df.write.partitionBy("Category").mode("overwrite").parquet("Files/bike_data")

---

24. How are partitioned folders structured?

Answer:
Folders are created using a `column=value` format.

Example:

Category=Mountain Bikes

Category=Road Bikes

---

25. Can you partition by multiple columns?

Answer:
Yes. This creates a hierarchical folder structure (e.g., Year → Month).

---

26. How do you load partitioned data for a specific partition?

Answer:

spark.read.parquet('Files/bike_data/Category=Road Bikes')

---

27. What happens to partition columns when loading a specific partition folder?

Answer:
The partition column (e.g., Category) is omitted from the resulting DataFrame because it is already implied by the folder path

1. What is Spark SQL?

Q: What is Spark SQL and why is it important?
A: Spark SQL is a Spark library that enables users to query and manipulate data using SQL expressions. It integrates the DataFrame API with SQL functionality, allowing developers and analysts to work with relational data using SQL syntax.

---

2. What is the Spark Catalog?

Q: What is the Spark catalog?
A: The Spark catalog is a metastore for relational data objects such as tables and views. It enables Spark to manage metadata and allows SQL queries to access these objects.

---

3. Temporary Views

Q: How can you make a DataFrame available for SQL querying in Spark?
A: By creating a temporary view using:

```
df.createOrReplaceTempView("products_view")
```

Q: What happens to a temporary view after the session ends?
A: It is automatically deleted at the end of the current Spark session.

---

4. Managed Tables

Q: How can you create a managed table from a DataFrame?
A: By using:

```
df.write.format("delta").saveAsTable("products")
```

Q: Where is data for managed tables stored in Microsoft Fabric?
A: In the Tables storage location of the data lake.

Q: What happens when you delete a managed table?
A: Both the table metadata and its underlying data are deleted.

---

5. Creating Tables in Spark

Q: Which method creates an empty table in the Spark catalog?
A: `spark.catalog.createTable`

Q: Which method saves a DataFrame as a table?
A: `saveAsTable`

---

6. Delta Lake Format

Q: What is the preferred table format in Microsoft Fabric?
A: Delta format.

Q: What technology is associated with delta format?
A: Delta Lake.

Q: What features do Delta tables support?
A:

- Transactions
- Versioning
- Streaming data support

---

7. External Tables

Q: How do you create an external table in Spark?
A: By using:

spark.catalog.createExternalTable

Q: Where is data for external tables stored?
A: In an external storage location, typically a folder in the Files area of a lakehouse.

Q: What happens when an external table is deleted?
A: Only the metadata is deleted; the underlying data remains intact.

---

8. Partitioning

Q: Why would you partition delta lake tables?
A: To improve query performance.

---

9. Querying Data Using Spark SQL API

Q: How can you execute a SQL query in PySpark?
A: By using:

spark.sql("SELECT ...")

Q: What does `spark.sql()` return?
A: A DataFrame containing the query results.

---

10. Using SQL in Notebooks

Q: How can you run SQL code directly in a notebook?
A: By using the `%%sql` magic command.

Example:

%%sql

SELECT Category, COUNT(ProductID) AS ProductCount

FROM products

GROUP BY Category

ORDER BY Category

Q: How are results displayed when using `%%sql`?
A: Automatically as a table in the notebook.

---

11. Key Differences: Temporary View vs Managed Table vs External Table

Feature	Temporary View	Managed Table	External Table
Stored in catalog	Yes	Yes	Yes
Underlying data stored by Spark	No	Yes	No
Deleted at session end	Yes	No	No
Deleting table removes data	N/A	Yes	No

1. What is the default way query results are displayed in a Spark notebook?

Answer:
Query results are displayed as a table by default under the code cell.

---

2. Can notebook results be displayed as something other than a table?

Answer:
Yes. Results can be changed from a table view to a chart view using the notebook's built-in charting functionality.

---

3. When is built-in notebook charting most useful?

Answer:
It is most useful for quick visual summaries of query results.

---

4. What limitation does built-in notebook charting have?

Answer:
It provides limited customization compared to using Python graphics libraries.

---

5. When should you use a graphics package instead of built-in charts?

Answer:
When you need more control over formatting and customization of the visualization.

✅ SECTION B: Graphics Packages in Code

6. Which base library are most Python visualization packages built on?

Answer:
Most Python visualization packages are built on Matplotlib.

---

7. Why are graphics libraries useful in notebooks?

Answer:
They allow combining:

- Data ingestion
- Data manipulation
- Inline data visualizations
- Markdown commentary

---

8. What must Spark DataFrames be converted to before using Matplotlib?

Answer:
They must be converted to a Pandas DataFrame.

---

9. Which method is used to convert a Spark DataFrame to a Pandas DataFrame?

Answer:
The `.toPandas()` method.

---

10. Why is `.toPandas()` necessary?

Answer:
Because Matplotlib requires data in Pandas format, not Spark DataFrame format.

✅ SECTION C: Understanding the Provided PySpark + Matplotlib Code

11. What SQL operation is performed in the example?

Answer:
The SQL query:

- Groups products by Category
- Counts ProductID
- Orders results by Category

---

12. What does `plt.clf()` do?

Answer:
It clears the current plot area before creating a new chart.

---

13. What does `fig = plt.figure(figsize=(12,8))` do?

Answer:
It creates a figure with a size of 12 inches by 8 inches.

---

14. What type of chart is created in the example?

Answer:
A bar chart.

---

15. What are the X-axis and Y-axis values in the example?

Answer:

- X-axis: Category
- Y-axis: ProductCount

---

16. What does `plt.xticks(rotation=70)` do?

Answer:
It rotates the X-axis labels by 70 degrees for better readability.

---

17. What is the purpose of `plt.grid()` in the example?

Answer:
It adds a styled grid to improve readability of the chart.

---

18. What function displays the chart in the notebook?

Answer:
`plt.show()`

✅ SECTION D: Conceptual Understanding

19. What is the advantage of using Matplotlib over built-in notebook charts?

Answer:
Matplotlib allows:

- Full customization of colors
- Control of figure size
- Grid styling
- Axis formatting
- Label rotation
- Advanced formatting options

---

20. Name another Python visualization library mentioned.

Answer:
Seaborn

---

21. Why might someone prefer Seaborn?

Answer:
To create highly customized and visually appealing charts.

---

22. What is the key workflow for visualizing Spark data using Matplotlib?

Answer:

1. Run SQL query in Spark
2. Convert result to Pandas using `.toPandas()`
3. Create figure
4. Plot chart
5. Customize chart
6. Display with `plt.show()`

✅ SECTION E: Scenario-Based Questions

23. If you only need a quick visual check of grouped data, what should you use?

Answer:
Built-in notebook chart functionality.

---

24. If you need to control chart size, colors, grid styling, and label rotation, what should you use?

Answer:
Matplotlib or another Python graphics library.

25. Why is visualization important in Spark notebooks?

Answer:
Because it makes query results easier to analyze and interpret visually rather than only as raw tables.

🔹 SECTION 1: Workspace & Lakehouse

1️⃣ What is required before starting this lab?

Answer:
Access to a Microsoft Fabric tenant with Fabric capacity enabled (Trial, Premium, or Fabric).

2️⃣ Why must Fabric capacity be enabled when creating a workspace?

Answer:
Because Spark workloads, lakehouses, notebooks, and Delta tables require Fabric compute capacity to run.

3️⃣ What is a Lakehouse in Fabric?

Answer:
A Lakehouse combines:

- The flexibility of a data lake (file storage like CSV, Parquet)
- The structure and SQL capabilities of a data warehouse

4️⃣ Why must “Lakehouse schemas (Public Preview)” be disabled in this lab?

Answer:
To ensure compatibility with the steps in the lab and avoid schema preview features that may change behavior.

🔹 SECTION 2: Data Ingestion

5️⃣ How were the CSV files ingested into the lakehouse?

Answer:
By:

- Downloading the `orders.zip`
- Extracting locally
- Uploading the entire folder using Upload Folder into the Lakehouse Files section.

6️⃣ Where are uploaded CSV files stored in Fabric?

Answer:
Inside the Files section of the Lakehouse (data lake storage).

7️⃣ What is automatically generated when selecting “Load data > Spark”?

Answer:
PySpark code:

```
df = spark.read.format("csv").option("header","true").load("Files/orders/2019.csv")
```

🔹 SECTION 3: Working with DataFrames

8️⃣ What happens when `.option("header","true")` is used?

Answer:
Spark treats the first row as column headers.

9️⃣ Why was `"header","false"` later used?

Answer:
Because the file didn’t properly define headers, and we needed to manually define schema instead.

🔟 Why is defining a schema important?

Answer:
Because:

- It assigns meaningful column names
- It enforces correct data types
- It improves query performance
- It avoids incorrect type inference

1️⃣1️⃣ What Spark object defines schema structure?

Answer:
`StructType` and `StructField` from:

```
from pyspark.sql.types import *
```

1️⃣2️⃣ How do you read multiple CSV files at once?

Answer:
Using wildcard:

```
.load("Files/orders/*.csv")
```

🔹 SECTION 4: Data Exploration

1️⃣3️⃣ Does filtering a DataFrame modify the original DataFrame?

Answer:
No.
Spark DataFrames are immutable — transformations return a new DataFrame.

1️⃣4️⃣ What is the difference between `select()` and `where()`?

Answer:

- `select()` → chooses columns
- `where()` → filters rows

They can be chained.

15️⃣ What does `distinct()` do?

Answer:
Removes duplicate rows.

16️⃣ What does `groupBy("Item").sum()` do?

Answer:
Groups records by product and sums numeric columns (e.g., Quantity).

17️⃣ Why was `alias("Year")` used?

Answer:
To rename the derived column from `year(OrderDate)` into a meaningful column name.

🔹 SECTION 5: Data Transformation

18️⃣ What does `withColumn()` do?

Answer:
Creates or replaces a column in a DataFrame.

19️⃣ How were Year and Month derived?

Answer:

year(col("OrderDate"))

month(col("OrderDate"))

2️⃣0️⃣ How were FirstName and LastName created?

Answer:

Using:

split(col("CustomerName"), " ")

2️⃣1️⃣ Why reorder columns after transformation?

Answer:
To structure data properly for downstream consumption and analytics.

🔹 SECTION 6: Saving Data

2️⃣2️⃣ Why is Parquet preferred over CSV?

Answer:

- Columnar storage
- Faster query performance
- Compressed storage
- Better suited for analytics

2️⃣3️⃣ How do you overwrite existing files?

Answer:

.write.mode("overwrite")

2️⃣4️⃣ What is partitioning?

Answer:
Splitting data into folder structures based on column values (e.g., Year=2021/Month=1).

25️⃣ Why partition by Year and Month?

Answer:

- Faster filtering
- Reduced scan size
- Better performance for large datasets

26️⃣ Why are partition columns not included in the loaded DataFrame?

Answer:
Because they are encoded in the file path structure and inferred during read.

🔹 SECTION 7: Working with Tables & Delta

27️⃣ What format is recommended for Fabric Lakehouse tables?

Answer:
Delta format.

28️⃣ Why use Delta format?

Answer:

- ACID transactions
- Time travel
- Schema enforcement
- Row versioning

---

29 What does `saveAsTable("salesorders")` do?

Answer:
Creates a managed table in the Spark metastore.

---

30 What is the Spark metastore?

Answer:
A catalog that stores metadata about tables in the lakehouse.

---

31 What does `%%sql` do?

Answer:
Changes the notebook cell language to Spark SQL instead of PySpark.

🔹 SECTION 8: SQL & Aggregation

32 How was GrossRevenue calculated?

Answer:

SUM((UnitPrice * Quantity) + Tax)

---

33 Why use `GROUP BY YEAR(OrderDate)`?

Answer:
To aggregate revenue per year.

🔹 SECTION 9: Visualization

34 Why convert Spark DataFrame to Pandas?

Answer:
Because matplotlib and seaborn require a Pandas DataFrame.

3️⃣5️⃣ What is matplotlib?

Answer:
A Python plotting library for customizable visualizations.

3️⃣6️⃣ What advantage does seaborn provide?

Answer:

- Simpler syntax
- Better default styling
- Built on top of matplotlib

3️⃣7️⃣ What does `plt.subplots()` allow?

Answer:
Creation of multiple charts in one figure.

🔹 SECTION 10: Architecture & Concepts

3️⃣8️⃣ Why is Lakehouse considered hybrid architecture?

Answer:
It combines:

- File-based storage (data lake)
- Structured tables with SQL (data warehouse)

3️⃣9️⃣ What is the benefit of using notebooks in Fabric?

Answer:

- Interactive development
- Supports multiple languages (PySpark, SQL, Scala, R)
- Unified environment for engineering & analytics

4️⃣0️⃣ What are the main Spark DataFrame operations demonstrated in this lab?

Answer:

- `read()`
- `schema()`
- `select()`
- `where()`
- `groupBy()`
- `sum()`
- `count()`
- `withColumn()`
- `write()`
- `partitionBy()`
- `saveAsTable()`

1. What is an Eventhouse in Microsoft Fabric?

Answer:
An Eventhouse is a container in Microsoft Fabric that stores large volumes of data. It houses one or more KQL databases optimized for storing and analyzing real-time data that arrives continuously from various sources.

---

2. What does an Eventhouse contain?

Answer:
An Eventhouse contains one or more KQL databases, each designed for real-time data ingestion and analysis.

---

3. What type of data is a KQL database optimized for?

Answer:
A KQL database is optimized for real-time data that arrives continuously from different sources.

---

4. How can data be loaded into a KQL database?

Answer:
Data can be loaded into a KQL database using:

- An Eventstream
- Direct ingestion into the KQL database

---

5. After data is ingested into a KQL database, what actions can be performed?

Answer:
After ingestion, you can:

- Query the data using KQL or T-SQL in a KQL queryset.
- Visualize data using Real-Time Dashboards.
- Automate actions using Fabric Activator.

---

6. Why is understanding KQL databases important?

Answer:
Understanding KQL databases helps you write effective queries to analyze real-time data and work with database objects such as materialized views and stored functions.

---

7. How do KQL databases automatically organize data?

Answer:
KQL databases automatically partition data by ingestion time, organizing it into separate storage locations based on when it arrived.

---

8. What is the benefit of partitioning data by ingestion time?

Answer:
Partitioning allows:

- Faster access to recent data.
- Efficient querying without scanning all historical data.
- Better performance for real-time analysis.

---

9. Why is recent data quickly accessible in a KQL database?

Answer:
Because data is partitioned by ingestion time, the system knows exactly where recent data is stored and does not need to scan the entire dataset.

---

10. What characteristic of real-time data enables automatic time-based organization?

Answer:
Real-time data represents immutable events that occurred at specific moments in time.

---

11. What does "immutable" mean in the context of real-time data?

Answer:
Immutable means that once an event occurs, it cannot be changed. For example, a temperature reading at 3:15 PM remains permanently tied to that time and value.

---

12. What is time-series data?

Answer:
Time-series data is data where the timestamp is as important as the event itself, because each event is permanently associated with the time it occurred.

---

13. What data pattern does time-series data follow?

Answer:
Time-series data follows an append-only pattern, where new events are continuously added and rarely updated or deleted.

---

14. Why is real-time data rarely updated or deleted?

Answer:
Because real-time data represents actual events that occurred at specific moments in time, and those events do not change after they happen.

---

15. How is a KQL database different from a traditional relational database?

Answer:
A KQL database:

- Uses an append-only pattern.
- Stores immutable, time-based events.
- Automatically partitions data by ingestion time.

Traditional relational databases:

- Frequently update existing records.
- Maintain relationships between tables.
- Do not primarily operate on immutable time-series data.

---

16. What analogy explains how KQL databases handle real-time data?

Answer:
A digital conveyor belt analogy:

- Events flow in continuously.
- They are automatically organized by arrival time.
- They become immediately available for analysis while new data continues to stream in.

1. Database Shortcuts

Q: What is a database shortcut in KQL?
A: A database shortcut is a reference to an existing KQL database in another eventhouse or Azure Data Explorer database, allowing you to query external data as if it were local, without copying the data.

Q: Why would you use a database shortcut instead of copying the data?
A: Database shortcuts avoid data duplication, save storage, and let you access and query external data in real-time without moving it.

---

2. OneLake Availability

Q: What does enabling OneLake availability do for a KQL database or table?
A: Enabling OneLake availability makes the database or table accessible across the Fabric ecosystem, allowing integration with Power BI, Warehouse, Lakehouse, and other Fabric services.

Q: Can OneLake availability be enabled for individual tables within a KQL database?
A: Yes, OneLake availability can be enabled for both individual databases and specific tables.

---

3. Querying KQL Databases

Q: How can you query a KQL database?
A: You can query a KQL database using KQL or T-SQL within a KQL queryset.

Q: What is automatically created when you create a KQL database to assist with queries?
A: An attached KQL queryset is automatically created for running and saving queries.

---

4. Basic KQL Syntax

Q: How does KQL process data in a query?
A: KQL uses a pipeline approach where data flows from one operation to the next using the pipe (`|`) character, with each operator filtering, rearranging, or summarizing the data.

Q: Is KQL case-sensitive?
A: Yes, KQL is case-sensitive for table names, column names, function names, operators, keywords, and string values. Identifiers must match exactly.

Q: How do you retrieve a sample of data from a table without processing the entire dataset?
A: Use the `take` operator, e.g., `TaxiTrips | take 100`.

Q: How do you aggregate data in KQL?
A: Use the `summarize` operator, e.g., `TaxiTrips | summarize trip_count = count() by taxi_id`.

---

5. KQL Querysets

Q: What is the purpose of a KQL queryset?
A: A KQL queryset provides a workspace for running, managing, and saving queries, organizing multiple query tabs, sharing queries for collaboration, and creating data visualizations.

Q: Can you use T-SQL in a KQL queryset?
A: Yes, KQL querysets support T-SQL queries alongside KQL syntax.

---

6. Data Visualization

Q: How can you visualize data when exploring a KQL database?
A: You can render query results as charts, tables, and other visual formats directly within a KQL queryset.

---

7. Copilot for Real-Time Intelligence

Q: What is Copilot in the context of KQL?
A: Copilot is an AI tool that assists with KQL queries by generating KQL code in response to questions about your data.

Q: How do you access Copilot in a KQL queryset?
A: When enabled by an administrator, Copilot appears as a pane in the queryset menu bar, allowing you to ask questions and receive generated KQL code.

1. Why does query optimization matter in KQL?

Answer: Query optimization matters because it improves performance by reducing the amount of data processed, uses fewer resources, and ensures queries remain reliable as data grows. Less data scanned means faster queries.

---

2. What is the key principle to follow when writing KQL queries?

Answer: The key principle is: the less data your query processes, the faster it runs.

---

3. How does filtering early improve KQL query performance?

Answer: Filtering early reduces the number of rows that subsequent operations need to process. KQL databases use indexes and data organization, so early filtering leverages these efficiencies.

---

4. Give an example of an effective early filter in time-series data.

Answer: Using a time-based filter:

```
TaxiTrips

| where pickup_datetime > ago(30min)
```

This uses the time index to quickly reduce rows.

---

5. How should you order filters in KQL?

Answer: Order filters from most data eliminated to least. Start with filters that remove the most rows, then apply more specific or value-based filters.

Example:

```
TaxiTrips

| where pickup_datetime > ago(1d)  // Most rows eliminated

| where vendor_id == "VTS"         // Specific filter

| where fare_amount > 0            // Least rows eliminated
```

---

6. Why is projecting (reducing) columns early important?

Answer: Selecting only needed columns reduces resource usage and improves performance, especially in wide tables with many columns.

Example:

TaxiTrips

| project trip_id, pickup_datetime, fare_amount

| where pickup_datetime > ago(1d)

---

7. What makes aggregations resource-intensive in KQL?

Answer: Aggregations are resource-intensive because they process and combine large amounts of data. Optimizing them by limiting results can improve performance.

Example:

TaxiTrips

| where pickup_datetime > ago(1d)

| summarize trip_count = count() by trip_id, vendor_id

| limit 1000

---

8. How should you structure joins in KQL for better performance?

Answer: Put the smaller table first in the join. KQL processes the first table against the second, so starting with fewer rows reduces resource usage.

Good example:

VendorInfo

| join kind=inner TaxiTrips on vendor_id

Avoid:

TaxiTrips

| join kind=inner VendorInfo on vendor_id

---

9. What is the impact of scanning unnecessary columns or rows?

Answer: Scanning unnecessary columns or rows increases processing time, consumes more resources, and reduces query performance.

---

10. Summarize the main KQL optimization techniques.

Answer:

1. Filter early and effectively, especially using time filters.
2. Order filters by how much data they eliminate.
3. Reduce columns early with `project`.
4. Optimize aggregations by limiting results when exploring data.
5. Optimize joins by starting with the smaller table.

1. What is a materialized view in KQL?

Answer:
A materialized view is a precomputed aggregation of data that stores the results of a query and automatically updates as new data arrives. It provides fast query responses for large datasets by avoiding repeated full computations on historical data.

---

2. Why are materialized views important in eventhouses?

Answer:
Eventhouses often contain millions or billions of streaming data rows (e.g., IoT sensors, logs). Aggregation queries on such large datasets can be slow. Materialized views improve performance by storing precomputed results, providing near-instant query results while maintaining up-to-date data.

---

3. How do materialized views maintain up-to-date results?

Answer:
Materialized views combine:

- Materialized part: precomputed aggregation from processed data
- Delta part: new incoming data since the last update
  The system merges these parts at query time and periodically moves delta data into the

materialized part to keep results current.

---

4. What is the difference between a regular table and a materialized view when querying?

Answer:
A materialized view is queried like a regular table but returns precomputed aggregation results, which reduces computation time and provides faster responses for analytics over large datasets.

---

5. How do you create a materialized view in KQL?

Answer:
You use the `.create materialized-view` command with a `summarize` query. Example:

```
.create materialized-view TripsByVendor on table TaxiTrips

{

  TaxiTrips

  | summarize trips = count(), avg_fare = avg(fare_amount), total_revenue = sum(fare_amount)

  by vendor_id, pickup_date = format_datetime(pickup_datetime, "yyyy-MM-dd")

}
```

---

6. How can you query a materialized view?

Answer:
You query it like a normal table. Example:

```
TripsByVendor

| where pickup_date >= ago(7d)

| project pickup_date, vendor_id, trips, avg_fare, total_revenue

| sort by pickup_date desc, total_revenue desc
```

---

7. What is a stored function in KQL?

Answer:
A stored function is a reusable query that can include parameters. It allows repeated logic to be executed consistently without rewriting the query multiple times.

---

8. Why are stored functions useful in eventhouses?

Answer:
They simplify query management for streaming data, reduce repeated query writing, ensure consistent calculations across team members, and allow parameterized queries for flexible data retrieval.

---

9. How do you create a stored function in KQL?

Answer:
You use `.create-or-alter function` with optional parameters. Example:

```
.create-or-alter function trips_by_min_passenger_count(num_passengers:long)

{

  TaxiTrips

  | where passenger_count >= num_passengers

  | project trip_id, pickup_datetime

}
```

---

10. How do you call a stored function in KQL?

Answer:
You use the function like a table and pass the required parameters. Example:

```
trips_by_min_passenger_count(3)

| take 10
```

---

11. What is the difference between a materialized view and a stored function?

Answer:

- Materialized view: stores precomputed aggregated results and updates automatically for fast queries.

- Stored function: encapsulates query logic for reuse, optionally with parameters, but doesn't store precomputed results.

Q: What is a Fabric lakehouse, and why is it important in the data-driven era?
A: A Fabric lakehouse is a platform that combines the capabilities of data lakes and data warehouses, allowing organizations to manage, store, and analyze large and diverse data sources efficiently. It is important because it enables data-driven decision-making and business insights.

Q: What are the three layers of the medallion architecture?
A: The three layers are Bronze (raw data), Silver (refined/cleaned data), and Gold (curated and business-ready data).

Q: What is the main purpose of the Bronze layer in medallion architecture?
A: The Bronze layer stores raw, unprocessed data ingested from multiple sources. It acts as a single source of truth before any transformation.

Q: How does the Silver layer differ from the Bronze layer?
A: The Silver layer contains cleaned, transformed, and enriched data. It is structured to remove errors, duplicates, and inconsistencies, making it ready for analytical use.

Q: What is the function of the Gold layer?
A: The Gold layer contains curated, aggregated, and business-ready data for reporting, dashboards, and decision-making.

Q: Why is medallion architecture considered a standard for lakehouse-based analytics?
A: Because it provides structure, efficiency, and scalability in managing data, enabling systematic refinement from raw to business-ready data.

Q: Who can benefit from learning the medallion architecture in Fabric lakehouses?
A: Both data engineers and analytics newcomers can benefit, as the architecture supports practical data organization, analysis, and reporting.

Q: What prior knowledge is recommended before starting this module?
A: Familiarity with the Fabric data lakehouse and basic understanding of SQL and Power BI.

Q: What types of activities will you perform in this module?
A: You will explore and build a medallion architecture, query and report on data, and learn best

practices for security and governance in a Fabric lakehouse.

Q: How does medallion architecture improve efficiency in a Fabric lakehouse?
A: By organizing data into layers, it reduces redundancy, improves data quality, and ensures data is refined and ready for analytics, saving time and resources during reporting.

1. What is medallion architecture?

Answer:
Medallion architecture is a recommended data design pattern used in Fabric lakehouses (built on Delta Lake format) to logically organize data into layers that improve data quality as it moves through them. It typically consists of bronze (raw), silver (validated), and gold (enriched) layers.

---

2. On what format are Fabric lakehouses built?

Answer:
Fabric lakehouses are built on the **Delta Lake** format, which natively supports ACID (Atomicity, Consistency, Isolation, Durability) transactions.

---

3. What does ACID stand for?

Answer:
ACID stands for:

- Atomicity
- Consistency
- Isolation
- Durability

---

4. What are the three standard layers in medallion architecture?

Answer:
The three standard layers are:

- Bronze (raw)
- Silver (validated)
- Gold (enriched)

---

5. Why is medallion architecture sometimes called a “multi-hop” architecture?

Answer:
It is called a multi-hop architecture because data moves between multiple layers (bronze → silver → gold) as it is refined and improved in quality.

---

6. What is the primary purpose of medallion architecture?

Answer:
Its primary purpose is to improve data quality progressively as data moves through layers while ensuring reliability, consistency, and easier analysis.

---

7. Does medallion architecture replace existing data models?

Answer:
No. It complements existing data organization methods rather than replacing them. It acts as a framework for data cleaning and refinement.

---

8. What is stored in the bronze layer?

Answer:
The bronze layer stores raw data in its original format (structured, semi-structured, or unstructured) without any modifications.

---

9. What transformations occur in the silver layer?

Answer:
In the silver layer:

- Data is validated and refined
- Null values may be removed
- Duplicate records are eliminated
- Data from multiple sources may be combined or merged
- Validation rules are enforced

---

10. What is the purpose of the silver layer?

Answer:
The silver layer serves as a validated, consistent central repository where data is cleaned and prepared for further modeling in the gold layer.

---

11. What happens in the gold layer?

Answer:
In the gold layer:

- Data is further refined
- Data is aggregated (e.g., daily or hourly)
- Data may be enriched with external information
- Data is prepared for analytics, data science, or MLOps

---

12. Who uses data from the gold layer?

Answer:
Downstream teams such as:

- Analytics teams

- Data science teams
- MLOps teams

---

13. Can medallion architecture have more than three layers?

Answer:
Yes. It is flexible and customizable. Organizations may add layers such as:

- A separate raw layer before bronze
- A platinum layer for highly refined, use-case-specific data

---

14. What factors should be considered when moving data across layers in Fabric?

Answer:
Key considerations include:

- Volume of data
- Complexity of transformations
- Frequency of movement
- Preferred tools

---

15. What is data transformation?

Answer:
Data transformation involves altering the structure or content of data to meet specific requirements.

---

16. Which tools are used for data transformation in Fabric?

Answer:

- Dataflows (Gen2)
- Notebooks

---

17. When should Dataflows (Gen2) be used?

Answer:
Dataflows (Gen2) are best suited for:

- Smaller semantic models
- Simple transformations

---

18. When should notebooks be used?

Answer:
Notebooks are better suited for:

- Larger semantic models
- More complex transformations
- Saving transformed data as managed Delta tables

---

19. What is data orchestration?

Answer:
Data orchestration is the coordination and management of multiple data-related processes to ensure they work together to achieve a desired outcome.

---

20. What tool is primarily used for data orchestration in Fabric?

Answer:
Pipelines are the primary tool used for data orchestration in Fabric.

---

21. What is a pipeline in Fabric?

Answer:
A pipeline is a series of steps that move data from one location to another, such as between layers of the medallion architecture.

---

22. How can pipelines be executed?

Answer:
Pipelines can:

- Run on a schedule
- Be triggered by an event

1. What is the purpose of the Bronze layer in a medallion architecture?
Answer: The Bronze layer ingests raw data as-is from the source, without transformation. It serves as the foundation for all subsequent layers.

---

2. Which tools can be used to ingest data into the Bronze layer in Fabric?
Answer: Pipelines, dataflows, or notebooks can be used to ingest data into the Bronze layer.

---

3. What is the main focus of the Silver layer?
Answer: The Silver layer focuses on cleansing and validating data to ensure quality and consistency, without performing extensive modeling.

---

4. Which tools are typically used for transformations in the Silver layer?
Answer: Dataflows or notebooks are used to perform transformations in the Silver layer.

---

5. What is the Gold layer used for in a medallion architecture?
Answer: The Gold layer is used for additional transformations, modeling, and preparing data for reporting or analytics. It often follows a dimensional (star schema) model.

---

6. How can data be consumed from the Gold layer?
Answer: Data can be consumed via SQL analytics endpoints, semantic models, or by connecting tools like Power BI for reporting and analysis.

---

7. Why might an organization create multiple Gold layers?
Answer: Multiple Gold layers can be created to serve different audiences or domains, such as finance, sales, or data science, with optimizations for reporting or machine learning.

---

8. What is the role of SQL analytics endpoints in Fabric's medallion architecture?
Answer: SQL analytics endpoints allow querying and modeling of the Gold layer data for reporting, visualization, or further analytics.

---

9. How does the Medallion architecture support downstream data consumption?
Answer: Downstream consumption is enabled via workspace or item permissions, SQL analytics endpoints, or connections to Gold layer Delta tables.

---

10. What is the recommended schema for the Gold layer for reporting purposes?
Answer: The Gold layer should follow a star schema and include relationships, measures, and other elements necessary for effective reporting.

---

11. Can the same lakehouse be used for multiple medallion architectures in Fabric?
Answer: Yes, the same lakehouse can host multiple medallion architectures, or different lakehouses can be used depending on the use case.

---

12. What is the key difference between transformations in Silver and Gold layers?
Answer: Silver focuses on data quality and consistency, whereas Gold focuses on data modeling, dimensional relationships, and analytics optimization.

---

13. What types of transformations should not be done in the Silver layer?
Answer: Extensive data modeling for reporting or analytics should not be done in the Silver layer; that is reserved for the Gold layer.

1. What is the purpose of querying data in the gold layer of a lakehouse?

Answer:
The gold layer contains curated, high-quality data optimized for analytics and reporting. Querying this layer allows data teams and business users to explore, analyze, and generate insights using SQL or semantic models while ensuring data accuracy and reliability.

---

2. What tools does Fabric provide to query and report on lakehouse data?

Answer:
Fabric provides several tools, including:

- **SQL analytics endpoint** – for writing queries, managing semantic models, and using the visual query experience.
- **Power BI semantic models** – which allow users to explore data with business-friendly terminology, often in star schema format.
- **Direct Lake mode in Power BI** – enabling semantic models to access data directly from delta tables in the lakehouse.

---

3. Can you modify data directly using the SQL analytics endpoint?

Answer:
No. The SQL analytics endpoint operates in **read-only mode** over lakehouse delta tables. To modify data, you need to use **dataflows, notebooks, or pipelines**.

---

4. What is a semantic model in the context of Fabric lakehouse?

Answer:
A semantic model is a representation of lakehouse data using **business-friendly terminology**. It typically follows a star schema structure with **fact tables** representing domain-specific data and **dimension tables** for analysis. Semantic models simplify reporting and make data more understandable for business users.

---

5. What is the benefit of using Direct Lake mode in Power BI?

Answer:
Direct Lake mode allows Power BI to access lakehouse delta tables directly via the semantic model, providing:

- **Real-time access to data** without data duplication.
- **Simplified reporting** for analysts using business-friendly views.
- **Reduced data movement** since queries are executed directly on the lakehouse.

---

6. How can medallion layers be tailored for different needs?

Answer:
Medallion layers (Bronze, Silver, Gold) can be customized to optimize **data processing and access** for specific user groups or analytical purposes. For example, different Gold layers can

be created for finance, sales, or data science, each structured to meet the specific requirements of its audience.

---

7. Why might multiple Gold layers be created in a lakehouse?

Answer:
Multiple Gold layers allow **diverse audiences or domains** to have optimized data structures tailored to their analytical requirements. This improves **performance, usability, and data relevance** for stakeholders with varying needs.

---

8. How does Fabric ensure data is usable for different applications or tools?

Answer:
Fabric leverages the medallion architecture to **generate cleansed and properly formatted data**, ensuring that external applications, third-party tools, or systems can consume the data in the required format.

---

9. What types of SQL operations are possible in the Fabric SQL analytics endpoint?

Answer:
Users can perform **exploratory queries, create views, save functions, and apply SQL security**. All operations are **read-only**, so no insert, update, or delete operations are allowed directly through this endpoint.

---

10. What is the advantage of using business-friendly terminology in a semantic model?

Answer:
It allows business users to **understand and interact with data without needing deep technical knowledge**, making analytics more accessible and reducing reliance on data engineers for reporting tasks.

1. What is the purpose of a data lakehouse in Microsoft Fabric?

Answer: A data lakehouse is a common analytical data store in Microsoft Fabric used for cloud-scale analytics solutions. It combines the scalability of a data lake with the structure of a

data warehouse, enabling data engineers to ingest, store, and analyze data from multiple operational sources.

---

2. What are the two main methods for ingesting data in Microsoft Fabric?

Answer: The two main methods are:

1. **ETL (Extract, Transform, Load)** – data is transformed before loading into the lakehouse.
2. **ELT (Extract, Load, Transform)** – data is first loaded into the lakehouse, then transformed using tools like Spark notebooks.

---

3. How do you create a workspace in Microsoft Fabric?

Answer:

1. Sign in to the Microsoft Fabric home page.
2. Select **Workspaces** from the left menu.
3. Click **Create Workspace**, give it a name, and select a licensing mode with Fabric capacity (Trial, Premium, or Fabric).
4. The new workspace will open as an empty workspace.

---

4. How do you create a lakehouse in Fabric?

Answer:

1. In the workspace, select **Create > Lakehouse** under Data Engineering.
2. Provide a unique name.
3. Ensure “Lakehouse schemas (Public Preview)” is disabled.
4. After creation, the lakehouse will initially have no tables or files.

---

5. What is the first step in creating a pipeline to ingest data?

Answer: On the Home page of your lakehouse, select **Get Data > New Data Pipeline**, then give it a name (e.g., *Ingest Sales Data*), and optionally use the Copy Data wizard to configure the pipeline.

---

6. How do you configure a Copy Data activity to ingest a CSV file from an HTTP source?

Answer:

1. In the pipeline, select **Copy Data > Use Copy Assistant**.
2. Choose **HTTP** as the data source.
3. Enter the source URL (e.g., `https://raw.githubusercontent.com/MicrosoftLearning/dp-data/main/sales.csv`) and select **Anonymous** authentication.
4. Configure file format options: DelimitedText, comma delimiter, first row as header, no compression.
5. Set the destination folder and file in the lakehouse.
6. Save and run the pipeline.

---

7. How do you verify that the data has been ingested successfully?

Answer:

1. Monitor the pipeline run in the Output pane under the pipeline designer.
2. Refresh the status until it shows **Succeeded**.
3. Expand the **Files** node in the lakehouse Explorer pane to check if the ingested file (e.g., `sales.csv`) exists in the target folder.

---

8. What is the purpose of a notebook in Microsoft Fabric?

Answer: Notebooks allow you to process and transform ingested data using **Spark code**. They support parameterized cells, enabling automation when run from a pipeline, and can save transformed data into lakehouse tables.

---

9. How do you create and configure a notebook for data transformation?

Answer:

1. In the lakehouse, select **Open notebook > New notebook**.

2. Replace the default code in the first cell with a parameter declaration (e.g., `table_name = "sales"`).

3. Toggle the cell as a **parameter cell**.

4. Add a code cell to read the CSV, transform it (e.g., split names, add Year/Month columns), reorder columns, and save to a delta table using `df.write.format("delta").mode("append").saveAsTable(table_name)`.

---

10. How do you integrate a notebook into a pipeline?

Answer:

1. In the pipeline designer, add a **Notebook activity**.
2. Connect the **On Completion** output of the Copy Data activity to the Notebook activity.
3. Set the notebook to run (e.g., *Load Sales*) and pass parameters like `table_name` for dynamic table creation.
4. Save and run the pipeline to perform a full ETL process.

---

11. Why would you add a Delete Data activity before copying data?

Answer: To ensure that old CSV files in the destination folder are removed before ingesting new data, preventing duplicates or stale data in the lakehouse.

---

12. What is the sequence of activities for a reusable ETL pipeline in Fabric?

Answer:

1. **Delete Data** – remove old files.
2. **Copy Data** – ingest new files from the source.
3. **Notebook** – transform the data and load it into a lakehouse table.

---

13. How do you troubleshoot Spark SQL errors in a pipeline?

Answer: Ensure the notebook is attached to a lakehouse. If errors occur:

1. Open the notebook, remove all lakehouses, then re-add the correct lakehouse.
2. Return to the pipeline and re-run.

---

14. How do you verify that the transformed data has been loaded into a table?

Answer:

1. Expand **Tables** in the lakehouse Explorer pane.
2. Select the target table (e.g., `sales` or `new_sales`) and preview the rows.
3. Confirm that the data reflects transformations applied in the notebook.

1. What is a Delta Table in Apache Spark / Microsoft Fabric?

Answer:
A Delta Table is a table stored in the **Delta Lake format**, which supports **ACID transactions**, **schema enforcement**, and **time travel**. In Microsoft Fabric, Delta Tables allow both **batch and streaming data processing** in Lakehouse tables.

---

2. What are the main differences between managed and external Delta tables in Fabric?

Answer:

- **Managed tables**: Fabric manages both schema metadata and data files. The data is stored in the **Tables folder** of the lakehouse. Deleting the table deletes both metadata and data.

- **External tables**: Only the schema metadata is managed by Fabric. Data files are stored externally (e.g., in a **Files folder**). Deleting the table deletes only the metadata, not the data.

---

3. How do you create a managed Delta table from a Spark DataFrame?

Answer:

df.write.format("delta").saveAsTable("managed_products")

This creates a **managed table**, storing the data and metadata under the lakehouse `Tables` folder.

---

4. How do you create an external Delta table from a Spark DataFrame?

Answer:

df.write.format("delta").saveAsTable("external_products", path="abfs_path/external_products")

Here, `path` specifies where the data is stored, while Fabric manages only the table metadata.

---

5. How can you verify where a Delta table's data is physically stored?

Answer:
Use SQL to describe the table:

%%sql

DESCRIBE FORMATTED managed_products;

DESCRIBE FORMATTED external_products;

- **Managed table** location ends with `/Tables/managed_products`.
- **External table** location ends with `/Files/external_products`.

---

6. How do you create a Delta table using SQL instead of PySpark?

Answer:

%%sql

CREATE TABLE products

USING DELTA

LOCATION 'Files/external_products';

This registers the Delta table in Fabric from the external data location.

---

7. How can you explore Delta table versioning / history?

Answer:

- Use the **transaction log** stored in `_delta_log` folder.
- View history with SQL:

%%sql

DESCRIBE HISTORY products;

- Access specific versions:

```
original_data = spark.read.format("delta").option("versionAsOf",
0).load('Files/external_products')
```

---

8. How do you modify data in a Delta table?

Answer:
You can perform SQL updates, which are recorded in the Delta log:

```
%%sql

UPDATE products

SET ListPrice = ListPrice * 0.9

WHERE Category = 'Mountain Bikes';
```

---

9. How do you create a temporary view from a Delta table for SQL queries?

Answer:

```
%%sql

CREATE OR REPLACE TEMPORARY VIEW products_view AS

SELECT Category, COUNT(*) AS NumProducts, MIN(ListPrice) AS MinPrice,

    MAX(ListPrice) AS MaxPrice, AVG(ListPrice) AS AvgPrice

FROM products

GROUP BY Category;
```

You can then run queries on `products_view` using SQL or PySpark.

---

10. How do you use Delta tables as a sink for streaming data?

Answer:

- Create a Spark Structured Streaming source (e.g., reading from a folder).
- Write the stream to Delta table:

```
deltastream = iotstream.writeStream.format("delta") \
        .option("checkpointLocation", checkpointpath) \
        .start('Tables/iotdevicedata')
```

- Query it with SQL:

```
%%sql
SELECT * FROM IotDeviceData;
```

---

11. How do you stop a streaming write to a Delta table?

Answer:

```
deltastream.stop()
```

---

12. What are the advantages of Delta Lake in Microsoft Fabric?

Answer:

- ACID transactions for reliable batch and streaming.
- **Time travel** to query previous versions of data.
- Schema enforcement to prevent bad data.
- Supports both managed and external tables.
- Seamless integration with **SQL queries**, **PySpark**, and **notebooks** in Fabric.

---

1. Workspace & Lakehouse Basics

Q1: What is the first step before working with data in Microsoft Fabric?
A1: You need to create a workspace with Fabric trial or Fabric capacity enabled.

Q2: How do you create a new lakehouse in a Microsoft Fabric workspace?
A2: In the workspace, select **+ New item** → **Lakehouse**, provide a name, and ensure the "Lakehouse schemas" option is disabled.

Q3: Where do you upload raw data in a medallion architecture?
A3: Raw data is uploaded to the **bronze layer** of the lakehouse.

Q4: Which files were used in this exercise to upload to the bronze layer?
A4: Three CSV files containing sales data for 2019, 2020, and 2021 (`2019.csv`, `2020.csv`, `2021.csv`).

---

2. Bronze to Silver Transformation

Q5: What tool/language is used in Fabric to transform data from bronze to silver?
A5: **Notebooks** using **PySpark** or SQL are used to transform data.

Q6: Why are new columns like `FileName`, `IsFlagged`, `CreatedTS`, and `ModifiedTS` added to the silver layer?
A6: To track the source file, flag early orders for validation, and maintain timestamps for record creation and modification.

Q7: How is `CustomerName` cleaned during silver transformation?
A7: Null or empty `CustomerName` values are replaced with `"Unknown"`.

Q8: What format is the silver table created in?
A8: Delta Lake format.

Q9: How do you update existing records or insert new ones in the silver Delta table?
A9: Using a **merge (upsert) operation** on key columns like `SalesOrderNumber`, `OrderDate`, `CustomerName`, and `Item`.

---

3. Silver Layer Exploration

Q10: How can you explore data in the silver layer using SQL?
A10: By using the **SQL analytics endpoint** in the Fabric workspace to run queries.

Q11: Provide an example SQL query to calculate total sales per year.
A11:

```
SELECT YEAR(OrderDate) AS Year,
    CAST(SUM(Quantity * (UnitPrice + Tax)) AS DECIMAL(12, 2)) AS TotalSales
FROM sales_silver
GROUP BY YEAR(OrderDate)
ORDER BY YEAR(OrderDate)
```

Q12: How can you identify top customers by quantity purchased?
A12: Using a SQL query to sum `Quantity` per `CustomerName` and order by descending total:

```
SELECT TOP 10 CustomerName, SUM(Quantity) AS TotalQuantity
FROM sales_silver
GROUP BY CustomerName
ORDER BY TotalQuantity DESC
```

---

4. Silver to Gold Transformation

Q13: Why is a separate notebook used for silver-to-gold transformation?
A13: To modularize the process for debugging, troubleshooting, and reusability.

Q14: What are the first steps in creating a gold layer?
A14: Load data from the `sales_silver` table into a dataframe and start building dimensions (star schema).

Q15: Which dimension tables are created in the gold layer?
A15:

1. `dimdate_gold` – Date dimension
2. `dimcustomer_gold` – Customer dimension
3. `dimproduct_gold` – Product dimension

Q16: How is the date dimension (`dimdate_gold`) populated?
A16: By dropping duplicates on `OrderDate` and extracting `Day`, `Month`, `Year`, and formatted strings `mmmyyyy` and `yyyymm`.

Q17: How is `CustomerID` generated for the customer dimension?
A17: Using `monotonically_increasing_id()` added to the current max `CustomerID` in the table.

Q18: How is product information split for the product dimension?
A18: The `Item` column is split into `ItemName` and `ItemInfo`, with `ItemID` generated similarly to `CustomerID`.

Q19: How is the fact table `factsales_gold` created?
A19: By joining the silver sales dataframe with `dimcustomer_gold` and `dimproduct_gold` to include `CustomerID` and `ItemID`, and selecting relevant columns (`OrderDate`, `Quantity`, `UnitPrice`, `Tax`).

Q20: How are new records synchronized in gold tables?
A20: Using Delta Lake **merge (upsert)** operations, comparing key columns like `CustomerID`, `ItemID`, and `OrderDate`.

---

5. Semantic Model (Optional)

Q21: What is the purpose of a semantic model in Fabric?
A21: To create relationships and measures for reporting and analysis using the gold tables.

Q22: Which tables are included in the semantic model for this exercise?
A22: `dimdate_gold`, `dimcustomer_gold`, `dimproduct_gold`, and `factsales_gold`.

Q23: Can you use the default semantic model generated with the lakehouse?
A23: No, you must create a **new semantic model** that includes the transformed gold tables.

---

6. General Medallion Architecture Knowledge

Q24: What are the three layers of medallion architecture?
A24:

1. **Bronze** – Raw data ingestion
2. **Silver** – Cleaned and transformed data
3. **Gold** – Curated, modeled data for reporting/analytics

Q25: Why is medallion architecture used in Fabric lakehouses?
A25: To separate raw ingestion, transformations, and curated analytics for better data quality, maintainability, and modularity.

Key Objectives Questions & Answers – Microsoft Fabric Activator

1. Objective: Understand the purpose of Activator in Microsoft Fabric.
Q: What is the primary function of an Activator in Microsoft Fabric?
A: An Activator monitors data in real-time and creates triggers to react to data changes, such as sending notifications when certain conditions are met.

---

2. Objective: Know how to create a workspace in Fabric.
Q: What are the steps to create a workspace in Microsoft Fabric for using the Activator?
A:

1. Sign in at Microsoft Fabric.

2. Select **Workspaces** from the left menu.
3. Click **Create**, give a workspace name, and select a licensing mode (Trial, Premium, or Fabric).
4. Open the new workspace, which will be empty initially.

---

3. Objective: Learn to create an Activator.
Q: How do you create an Activator in a Fabric workspace?
A:

1. Click **Create** → under **Real-Time Intelligence**, select **Activator**.
2. Wait for the Activator to be created.
3. Optionally, select **Try sample** to populate it with sample data.
4. Rename the Activator to a descriptive name (e.g., “Contoso Shipping Activator”).

---

4. Objective: Explore and understand the Activator home screen.
Q: What does the Event details live table in Activator show?
A: It displays real-time package delivery events, including properties like Temperature, City, ColdChainType, and SpecialCare, which can be used to create rules.

---

5. Objective: Understand object creation in Activator.
Q: How do you create a new object in Activator and why would you do it?
A:

- Select the eventstream → click **New object**.
- Enter **Object name**, **Unique Identifier**, and **Properties**.
- Click **Create**.
- You create objects to organize eventstream data and define which attributes rules will monitor.
  Example: “Redmond Packages” object with properties City, ColdChainType, SpecialCare, and Temperature.

---

6. Objective: Understand rules in Activator.
Q: How do you create a rule to monitor temperature for refrigerated medicine packages?
A:

1. Select the object and property (Temperature).
2. Click **New Rule**.
3. Set **Condition** (e.g., “Increases above” 20).
4. Set **Occurrence** (e.g., “Every time the condition is met”).
5. Set **Action** (e.g., Send me an email).
6. Apply **filters** for City = Redmond, SpecialCare = Medicine, ColdChainType = Refrigerated.
7. Save and start the rule.

---

7. Objective: Understand property filters in rules.
Q: What filters are needed to ensure the temperature alert only applies to Redmond medicine packages?
A:

1. City = Redmond
2. SpecialCare = Medicine
3. ColdChainType = Refrigerated

---

8. Objective: Configure actions for rule triggers.
Q: What action can you take when a rule fires in Activator?
A: You can send an **email**, a **Teams message**, or other notifications. In this lab, the action is to **send an email** to the shipping department with details about the trigger.

---

9. Objective: Test and manage rules.
Q: How can you verify that a rule in Activator works?
A: Select **Send me a test action** in the Action section. You should receive the message defined in the rule with details like activation time and package ID.

Q: How do you stop a rule once verified?
A: Click the **Stop** button on the ribbon in the Activator interface.

---

10. Objective: Understand the significance of the Activator in a business scenario.
Q: Why is it important to use Activator for monitoring Redmond medical packages?
A: It ensures that prescription medicine packages are maintained within the required temperature range (33–41°C), preventing spoilage and maintaining patient safety.

1. What is Eventstream in Microsoft Fabric?
Answer: Eventstream is a feature in Microsoft Fabric that captures, transforms, and routes real-time events to various destinations. It allows you to connect event data sources, perform transformations, and send the data to destinations like Eventhouse tables for analysis.

---

2. What is the first step to work with data in Microsoft Fabric?
Answer: The first step is to create a workspace in Microsoft Fabric with Fabric capacity enabled (Trial, Premium, or Fabric). The workspace is required to store Eventhouse databases and Eventstreams.

---

3. What is an Eventhouse in Microsoft Fabric?
Answer: An Eventhouse is a component in Microsoft Fabric that contains a KQL (Kusto Query Language) database where eventstream data can be ingested and stored for real-time analysis.

---

4. How do you create an Eventstream in Microsoft Fabric?
Answer:

1. Open the KQL database in your workspace.
2. Select **Get data** > **Eventstream** > **New eventstream**.
3. Name the eventstream (e.g., `Bicycle-data`) and start the primary editor.
4. Add a data source and a destination table in Eventhouse.

---

5. How do you add a source to an Eventstream?
Answer: In the Eventstream canvas, select **Use sample data**, give it a name (e.g., `Bicycles`), and select the sample data to be ingested.

---

6. How do you add a destination to an Eventstream?
Answer:

1. Select **Transform events or add destination** > search for **Eventhouse**.
2. Configure the destination:
    - Data ingestion mode: Event processing before ingestion
    - Destination name: e.g., `bikes-table`
    - Workspace, Eventhouse, KQL database: select your existing ones
    - Destination table: create a new table (e.g., `bikes`)
    - Input data format: JSON
3. Save and **Publish** the eventstream.

---

7. How can you view the data being ingested by the Eventstream?
Answer: Select the destination node (e.g., `bikes-table`) on the design canvas and use the **Data preview pane**. Refresh periodically to see new data as the stream runs perpetually.

---

8. How do you query real-time data in the Eventhouse table?
Answer:

1. Navigate to your KQL database.
2. Select the table (e.g., `bikes`).
3. Use the built-in query option like:

```
bikes
| where ingestion_time() between (now(-1d) .. now())
```

This retrieves records ingested in the last 24 hours.

---

9. Why would you transform event data before ingestion?
Answer: Transforming data allows you to aggregate, filter, or modify events to meet business requirements before storing them in the Eventhouse. For example, summing the number of bikes per street every 5 seconds.

---

10. How do you apply a "Group by" transformation in Eventstream?
Answer:

1. Edit the eventstream and select **Transform events** > **Group by**.
2. Configure:
    - Operation name: `GroupByStreet`
    - Aggregate type: `Sum`
    - Field: `No_Bikes`
    - Group by: `Street`
    - Time window: Tumbling, Duration: 5 seconds
3. Connect the output to a new Eventhouse node to save the transformed data.

---

11. How is the transformed data stored?
Answer: The transformed data is stored in a new Eventhouse table (e.g., `bikes-by-street`) with fields:

- `Street` (grouping field)
- `SUM_No_Bikes` (aggregated sum)
- `Window_End_Time` (timestamp for the tumbling window)

---

12. How do you query transformed data for analysis?
Answer: Use a KQL query like:

['bikes-by-street']

| summarize TotalBikes = sum(tolong(SUM_No_Bikes)) by Window_End_Time, Street

| sort by Window_End_Time desc, Street asc

This shows the total number of bikes per street for each 5-second window.

---

13. What is a tumbling window in Eventstream transformations?
Answer: A tumbling window is a fixed, non-overlapping time interval used to aggregate events. In this exercise, it calculates the sum of bikes every 5 seconds.

---

14. What format should eventstream data use for Eventhouse ingestion?
Answer: JSON is the input data format for Eventhouse tables in this exercise.

---

15. How long does it take to complete the bicycle eventstream lab in Microsoft Fabric?
Answer: Approximately 30 minutes.