Mock-Interim-Previously-Asked

- Mock-Interim-Previously-Asked
  - 1. What is a dataset in azure
  - 2. What is a linked service?
  - 3. Difference between linked service and dataset?
  - 4. Why do you use dataset?
  - 5. What is Physical dataset and inline dataset
  - Physical Dataset
  - Inline Dataset
  - 6. Why is Dataset needed if linked service is already there
  - 7. You created dataset. What are the types of datasets you have worked with in the past?
  - 8. Explain what is Linkedservice in simple terms (Without mentioning dataset)
  - 9. Which tab do you create a dataset and linked service
  - Linked Service
  - Dataset
  - 10. Why do you use linked service
  - 11. What are the other things you can connect via linked service other than data stores
  - 12. What is an Integration Runtime? What are the different types?
  - 13. What is the difference between Linked service and integration runtime?
  - 14. What is the default integration runtime
  - 15. What is time-to-live in integration runtime
  - 16. Where do you configure time to live?
  - 17. Ways to call one pipeline from another pipeline
  - 18. What are the types of triggers?
  - Types of Triggers in Azure Data Factory
  - 19. What is tumbling window?
  - 20. Can you use a tumbling window to run a pipeline from the past
  - 21. Event based trigger
  - 22. How to differentiate Event based triggers and storage based triggers in monitor
  - 23. Different types of Activities
  - 24. What format does getmetadata return?
  - 25. Difference between on success and on completion
  - 26. What are the different tabs in Azure Data Factory (Author, Monitor etc)
  - 27. What all options are there in the right end of activity, when we want to join to another

1. What is a dataset in azure

A dataset in Azure is a reference to the data you want to use in your pipeline. It tells Azure where the data is stored and how to access it.

2. What is a linked service?

A Linked Service in Azure is like a connection string or a bridge that tells Azure how to connect to external data sources.

It stores the details needed to connect to a data source, like a database, storage account.

3. Difference between linked service and dataset?

- A Linked Service defines how to connect to a data source.
  - Example: Azure SQL Database connection info
- A Dataset defines what data to use from that source.
  - Example: A table named SalesData in that SQL database
- Datasets rely on Linked Services for connection details.

4. Why do you use dataset?

- A Dataset is used to define and describe the specific data that you want to work with in your Azure pipeline or project.
- It tells Azure where the data is stored (e.g., file path, table name).
- It defines how the data looks (e.g., CSV, JSON, Parquet, schema, delimiter).
- It keeps the data definition separate from the connection (which is handled by Linked Service).

5. What is Physical dataset and inline dataset

Physical Dataset

- A Physical Dataset refers to a dataset that is defined separately and stored as a reusable object in your Azure Data Factory.
- Example: You create a dataset named SalesData_CSV that points to a CSV file in Azure Data Lake. You can use this dataset in multiple pipelines.

Inline Dataset

- An Inline Dataset is a dataset that is defined directly inside an activity (like a Copy Activity) in Azure Data Factory, instead of being created and saved separately
- Defined inside the activity (not saved as a separate object).
- Used only once in that activity.
- Not reusable across other activities or pipelines.

6. Why is Dataset needed if linked service is already there

- Linked Service = How to connect
  - It contains:
    - Connection info (e.g., storage account, database server)
    - Authentication (e.g., key, username/password)
    - Type of service (e.g., Azure Blob, SQL, REST)
  - It tells Azure how to reach the data source.
- Dataset = What data to use
  - File path or table name
  - File format (CSV, JSON, Parquet, etc.)
  - Schema or structure of the data
  - Reference to a Linked Service

7. You created dataset. What are the types of datasets you have worked with in the past?

- Azure Data Lake Gen2 Dataset
  - Type: File-based or folder-based
  - Use Case: Used for big data processing and hierarchical storage.
  - Example: Dataset pointing to datalake/raw/sales/2025/
- JSON Dataset
  - Type: Semi-structured
  - Use Case: Used for working with nested or hierarchical data.
  - Example: JSON logs or configuration files.
- Parquet Dataset
  - Type: Columnar storage format
  - Use Case: Used for efficient big data analytics.
  - Example: Parquet files stored in Data Lake for Spark processing.

8. Explain what is Linkedservice in simple terms (Without mentioning dataset)

A Linked Service is like a connection setup that tells Azure how to connect to an external system or service.

9. Which tab do you create a dataset and linked service

Linked Service

- Azure Data Factory
  - Manage Tab
    - Connections
      - Linked services
        - Create new linked service
        - From there we can select the data store (Eg: Azure Data lake storage)
        - Select Integration Runtime
        - Then we assign the storage account

Dataset

- Azure Data Factory
    - Author Tab
        - Data sets
            - 3 Dots
                - New Dataset
                    - Select Data store (Eg: Azure Data Lake)
                    - Choose Data Format (Eg: JSON)
                    - Choose the Linked Service
                    - Choose the file path

10. Why do you use linked service

- Main Reasons to Use a Linked Service:
    - To establish a secure connection
    - It stores connection details like server name, URL, and authentication (keys, credentials, etc.).
- To avoid repeating connection info
    - You define it once and reuse it across multiple activities or pipelines.
- To support multiple data sources
    - You can connect to various services like Azure Blob, SQL Server, REST APIs, etc.
- To separate connection logic from business logic
    - This makes your pipelines cleaner and easier to manage.
- To enable centralized management
    - If connection details change, you only need to update the Linked Service — not every activity.

11. What are the other things you can connect via linked service other than data stores

You can connect to compute services (like Databricks, HDInsight), integration services (like Azure Functions, REST APIs), and security services (like Azure Key Vault).

12. What is an Integration Runtime? What are the different types?

An Integration Runtime is the compute infrastructure used by Azure Data Factory to move, transform, and integrate data between different data stores and services.

- Why is it needed?
  - To copy data between sources
  - To run data flows or transformations
  - To connect to on-premises systems
- Types of Integration Runtimes
  - Azure IR
    - Fully managed by Microsoft in the cloud
    - For cloud-to-cloud data movement and transformations
  - Self-hosted IR
    - Installed on your own machine or server
    - For accessing on-premises data sources or private networks
  - Azure-SSIS IR
    - Special IR to run SQL Server Integration Services (SSIS) packages in Azure

13. What is the difference between Linked service and integration runtime?

- Linked Service defines how to connect to a data source, while Integration Runtime defines where and how the data operations are executed.
- Linked Service needs an IR to actually perform the operations like copying or transforming data.
- Role of Linked Service is connecting to data stores. IR is responsible for executing the operations.

14. What is the default integration runtime

- The Default Integration Runtime is the built-in, auto-managed compute provided by Azure Data Factory (ADF) or Azure Synapse. It is created automatically when you set up your data factory — you don’t need to configure it manually.
- Example: AutoResolveIntegrationRuntime

15. What is time-to-live in integration runtime

Time-to-Live refers to the amount of time that the compute cluster stays alive after the last activity finishes.

- Why is TTL Important?
  - When you run a Data Flow, Azure spins up a Spark cluster behind the scenes.
  - After the job finishes, the cluster doesn't shut down immediately — it waits for the TTL duration.
  - If another job starts during that time, it reuses the same cluster, which saves startup time and cost.
- How TTL Works:
- Default TTL: 0 minutes (cluster shuts down immediately after job ends)
- Custom TTL: You can set it (e.g., 10, 15, 30 minutes) to keep the cluster warm for reuse

16. Where do you configure time to live?

- Azure Data factory
  - Connections
    - Integration Runtime
      - Click the integration runtime
        - Go to settings and configure the time to live.

17. Ways to call one pipeline from another pipeline

- Execute pipeline Activity
- REST API Call
- Web Activity to call Azure function, which can trigger a pipeline
- Trigger based execution

18. What are the types of triggers?

Types of Triggers in Azure Data Factory

Trigger Type	Description	Use Case Example
1. Schedule Trigger	Runs pipelines on a **recurring schedule** (e.g., every hour, daily)	Run a pipeline every day at 6 AM

Trigger Type	Description	Use Case Example
2. Tumbling Window Trigger	Runs pipelines in **fixed-size time intervals**	Process hourly data with retry logic
3. Event-Based Trigger	Starts a pipeline when a **specific event** occurs in a storage account (e.g., file created or deleted)	Trigger pipeline when a new file arrives
4. Manual Trigger	Triggered **manually by a user** or via **REST API** or **PowerShell**	Run a pipeline on-demand for testing

19. What is tumbling window?

A Tumbling Window is a type of trigger in Azure Data Factory that runs pipelines in fixed-size, non-overlapping time intervals — like hourly, daily, or weekly windows.

- Example Use Case:
  - You want to process log files every hour.
  - You set a tumbling window trigger with a 1-hour interval.
  - The pipeline runs at 1 PM, 2 PM, 3 PM, etc., processing only the data for that hour.
- How its different from Schedule Trigger?
  - Tumbling Window is stateful and supports dependency and retry logic, while Schedule Trigger is stateless.
  - We can configure dependencies between windows.

20. Can you use a tumbling window to run a pipeline from the past

- Yes, you can configure a Tumbling Window Trigger in Azure Data Factory to run pipelines for past time windows
- When you create a tumbling window trigger, you specify:
  - Start time (can be in the past)
  - End time (optional)
  - Window size (e.g., 1 hour, 1 day)
- Azure will automatically create instances of the pipeline for each time window between the start and end time — even if those windows are in the past.
- Example:
  - You set a tumbling window trigger with:
  - Start time: 2025-06-01 00:00

- Window size: 1 day
- Azure will run the pipeline for:
    - June 1
    - June 2
    - June 3
    - … up to the current date

21. Event based trigger

- An Event-Based Trigger is a type of trigger in Azure Data Factory that automatically starts a pipeline when a specific event occurs in Azure Blob Storage or Azure Data Lake Storage Gen2.
- Events
    - Blob Created – When a new file is uploaded
    - Blob Deleted – When a file is removed

22. How to differentiate Event based triggers and storage based triggers in monitor

- There are tabs for Storage based and custom event in monitor

23. Different types of Activities

- Copy Activity – Copies data between source and destination (e.g., from Blob to SQL).
- Execute Pipeline – Calls another pipeline.
- If Condition – Runs activities based on a condition (like an IF statement).
- Switch Activity – Like a switch-case logic.
- ForEach Activity – Loops through a collection of items.
- Until Activity – Repeats activities until a condition is met.
- Wait Activity – Pauses pipeline for a specified time.
- Web Activity – Calls a REST API or webhook.

24. What format does getmetadata return?

- The Get Metadata activity in Azure Data Factory returns its output in JSON format — a structured format that contains key-value pairs based on the metadata fields you request.

25. Difference between on success and on completion

- On Success
  - The next activity runs only if the previous activity succeeds.
  - If the previous activity fails or is skipped, the next activity will not run.
- On Completion
  - The next activity runs regardless of the outcome of the previous activity.
  - It will run whether the previous activity succeeds, fails, or is skipped.

26. What are the different tabs in Azure Data Factory (Author, Monitor etc)

- Author Tab
  - Purpose: To create and manage pipelines, datasets, linked services, and triggers.
  - Build pipelines using drag-and-drop activities.
  - Define datasets (data sources and destinations).
  - Create triggers to schedule pipeline runs.
- Monitor Tab
  - Purpose: To track the execution and performance of your pipelines.
  - View pipeline run history.
  - Check success/failure status.
  - Debug failed runs by checking error messages.
  - Monitor trigger executions
- Manage Tab
  - Purpose: To configure and manage global settings and resources.
  - Create and manage linked services.
  - Set up integration runtimes (compute environments).

27. What all options are there in the right end of activity, when we want to join to another activity (on success, on failure etc)

- On Success
  - Meaning: The next activity runs only if the current activity succeeds.

    - Use Case: Most common scenario — continue only if everything is working fine.
    - Icon: Green check
- On Failure
    - Meaning: The next activity runs only if the current activity fails.
    - Use Case: For error handling, like sending an alert or logging the error.
    - Icon: Red cross
- On Completion
    - Meaning: The next activity runs regardless of success or failure.
    - Use Case: Cleanup tasks, logging, or notifications that should always run.
    - Icon: Blue right arrow.
- On Skip
    - Meaning: The next activity runs only if the current activity is skipped.
    - Use Case: Conditional branching where some activities might be skipped based on expressions or parameters.
    - Icon: Gray arrow

28. What is LRS, GRS, ZRS, GZRS

1. LRS – Locally Redundant Storage

- **Replication Scope:** Within a single data center.
- **Copies:** 3 copies.
- **Use Case:** Cost-effective, suitable for non-critical data.
- **Protection:** Against hardware failures in the same location.

2. GRS – Geo-Redundant Storage

- **Replication Scope:** Across two regions (primary + secondary).
- **Copies:** 6 copies (3 in each region).
- **Use Case:** Disaster recovery, high availability.
- **Protection:** Against regional outages.

3. ZRS – Zone-Redundant Storage

- **Replication Scope:** Across **three availability zones** in the same region.

- **Copies:** 3 copies.
- **Use Case:** High availability within a region.
- **Protection:** Against zone-level failures (like power or network issues).

4. GZRS – Geo-Zone-Redundant Storage

- **Replication Scope:** Combines **ZRS + GRS**.
- **Copies:** 3 in zones of primary region + 3 in secondary region.
- **Use Case:** Maximum durability and availability.
- **Protection:** Against both zone and regional failures.

Replication Type	Where Copies Are Stored	Protection Scope
LRS	3 copies in **1 AZ**	Hardware failure
ZRS	3 copies across **3 AZs** in **1 region**	Zone failure
GRS	3 copies in **Region A** + 3 in **Region B**	Regional disaster
GZRS	ZRS in **Region A** + 3 in **Region B**	Zone + regional failure

29. What are the different tiers of storage

Tier	Access Frequency	Retention	Storage Cost	Access Cost	Latency	Best For
Hot	Frequent	None	High	Low	Milliseconds	Frequently accessed, active data
Cool	Infrequent	≥ 30 days	Medium	Medium	Milliseconds	Backups, less-accessed files
Cold	Rare	≥ 90 days	Low	High	Milliseconds	Rarely accessed data with fast retrieval needed
Archive	Very Rare	≥ 180 days	Very Low	Very High	Hours	Long-term archival, compliance data

30. Why do you use data flow in pipeline, what is the use?

- You use Data Flows inside a pipeline when you need to perform data transformation — that is, when you want to clean, shape, join, filter, or aggregate your data before loading it to the destination.

31. Without dataflow I cannot do certain operations. What are those?

In Azure Data Factory, while you can do basic data movement using Copy Activity, there are certain advanced data transformation operations that cannot be done without using Mapping Data Flows.

- We cant do things like
  - Complex Joins
  - Aggregrations
  - Surrogate key Generation
  - Filters

32. What is fact and dimensions

Facts

- What is a Fact Table?
  - A Fact Table contains measurable, quantitative data — the "facts" of your business.
  - These are usually numbers you want to analyze, like:
    - Sales amount
    - Quantity sold
    - Revenue
    - Profit
- Characteristics:
  - Contains foreign keys to dimension tables.
  - Contains metrics or measures.
  - Grows vertically (more rows over time).
- A fact table stores numeric data like sales or revenue, and links to dimensions for context.

Dimensions

- What is a Dimension Table?
  - A Dimension Table contains descriptive attributes — the "context" for your facts.
  - Examples:
    - Product name, category
    - Customer name, location
    - Date, month, year
    - Store name, region
  - Characteristics:
    - Contains textual or categorical data.
    - A dimension table gives meaning to facts — like product names, customer details, or dates.

33. What is a surrogate key? Why do we need it?

- A surrogate key is a unique, system-generated identifier used in a database table, especially in data warehousing, to uniquely identify each record in a dimension table.
- Why Do We Need a Surrogate Key?
  - 1. Uniqueness
    - Ensures each row in a dimension table has a unique identifier, even if business data changes.
  - 2. Handles Slowly Changing Dimensions (SCD)
    - When customer or product details change over time, surrogate keys help track historical versions of the data.
  - 3. Keys can change
    - Real-world data changes — a customer might change their ID system, or a product code might be updated.
    - Surrogate keys are stable and never change, which helps maintain data integrity over time.
  - 4. Performance
    - Surrogate keys are usually integers, which are faster for joins and indexing than strings or composite keys.
  - 5. Consistency

- If data comes from multiple sources, business keys might conflict or overlap.
- Surrogate keys ensure consistency in your data warehouse.

34. What is a candidate key?

- A candidate key is a column or a combination of columns in a table that can uniquely identify each row in that table
- It is a "candidate" to become the primary key.
- A table can have multiple candidate keys, but only one is chosen as the primary key.

35. What do you know about parquet?

- Parquet is a columnar file format used in big data tools. It’s efficient for storage and fast for reading because it only loads the columns you need.
- It’s widely used in Azure Data Factory, Spark, and other data platforms.

36. What is a constructor?

- A constructor is a special type of method used in object-oriented programming (OOP) to initialize objects of a class.
- It automatically runs when you create (instantiate) an object.
- It sets up initial values for the object’s properties (also called fields or attributes).
- It ensures the object is in a valid state right from the start.