Rishab IT Group

Rishab Informatica Group

CALL/WHATSAPP - 8970853557 / 9448005273

Azure data factory

Top 50 - Scenario based interview Questions & answers

1. Copy Data from On-premises SQL Server to Azure SQL Database

- **Answer**: Set up a Self-hosted Integration Runtime (SHIR) to connect to the on-premises SQL Server. Create linked services for both the on-prem SQL Server and the Azure SQL Database, then define source and sink datasets. Finally, add a Copy Activity in your pipeline to transfer data.

2. Handling Multiple File Formats (CSV, JSON, Parquet)

- **Answer**: Create parameterized datasets for each file format, configure a Switch activity to handle specific format types, and use mapping data flows to apply necessary transformations for each format before loading them into the target.

3. Intermittent Timeout Errors in Pipeline

- **Answer**: Increase timeout settings in your activities, set a retry policy in Copy Activity, and ensure your integration runtime has sufficient resources. Additionally, check for any network stability issues between the source and ADF to reduce timeouts.

4. Implement Incremental Data Loading

- **Answer**: Use a watermark column (like a timestamp) to track changes. Implement a Lookup activity to capture the last watermark value, then use it in the query of the source dataset to load only new or modified data. Save the updated watermark for the next run.

5. Securing Sensitive Data like API Keys and Passwords

- **Answer**: Store API keys, passwords, and other sensitive information in Azure Key Vault. Use ADF’s managed identity to retrieve these secrets, ensuring that no sensitive data is hardcoded in your pipeline.

6. Optimizing Pipelines with Large Data Volumes

- **Answer**: Enable parallelism in Copy Activity, partition your data for efficient transfer, and choose an integration runtime with sufficient resources. Configure Copy Activity settings for high throughput and, if possible, avoid transformations during the copy process to reduce latency.

7. Scheduling Pipeline for Weekdays Only

- **Answer**: Configure a schedule trigger and set the recurrence to only include weekdays (e.g., Monday to Friday). You can also set the trigger’s time zone to ensure accurate start times across different regions.

8. Data Deduplication During ETL

- **Answer**: Use data flows to remove duplicates by applying Aggregate or Window functions. Group by a unique identifier column, and either keep the first record or apply any required logic to select which duplicate to retain.

9. Implement Retry Mechanism for API Calls

- **Answer**: In the Web Activity, configure the Retry option with a specific retry count and interval. This will handle intermittent API failures by retrying the API call based on your configured retry settings.

10. Handling Schema Drift in Source Data

- **Answer**: Enable schema drift in your data flow, allowing ADF to automatically detect changes in source schema without breaking the pipeline. Use this feature to handle variations in data structure over time.

11. Merge New Data with Existing Data

- **Answer**: Use Delta Lake or a data flow with conditional updates. Define join conditions to identify new and existing records, then apply Upsert logic to update existing data and insert new records in the target.

12. Configuring an End-to-End Data Pipeline with Failure Notifications

- **Answer**: Set up failure alerts with Azure Monitor and configure ADF to notify stakeholders in case of pipeline failures. Use an If Condition activity to handle failure notifications within the pipeline, which could include an email or Teams message on failure.

13. Dynamic Copying from Multiple Containers

- **Answer**: Parameterize the container name in the dataset and use a For Each activity to loop through the list of containers. This allows ADF to dynamically access and copy data from multiple containers in one pipeline execution.

14. Automating Data Load Based on File Arrival

- **Answer**: Set up an Event Trigger that starts the pipeline as soon as a new file is added to blob storage. Define the file path as a parameter in the trigger to handle dynamic file names.

15. Transforming JSON Data to Relational Format

- **Answer**: Use data flows with flatten transformation to convert JSON hierarchies into tabular format. This involves expanding nested JSON objects into separate columns to fit a relational schema.

16. Loading Data from REST API with Pagination

- **Answer**: Set up a Web Activity with pagination, specifying next page parameters according to API documentation. Use variables or dynamic expressions to fetch data page-by-page until all records are retrieved.

17. Processing Only Changed Files in Blob Storage

- **Answer**: Use the Get Metadata activity to retrieve the last modified date of each file. Use an If Condition activity to process only files that have been modified since the last pipeline run.

18. Migrating Data from Amazon S3 to Azure Data Lake Storage

- **Answer**: Set up Amazon S3 as a linked service in ADF, define datasets for S3 as source and ADLS as sink, and use Copy Activity to transfer the data. Ensure network connectivity between AWS and Azure is optimized.

19. Error Handling in Complex Pipelines

- **Answer**: Use Try-Catch patterns, leveraging If Condition and Execute Pipeline activities. Configure the error handling logic to re-run specific activities or trigger an alert if any activity fails.

20. Handling Slowly Changing Dimensions (SCD)

- **Answer**: Use data flows to implement SCD Type 1 (overwrite) or SCD Type 2 (track history). For Type 2, add effective date columns and update these when changes are detected.

21. Data Partitioning for Improved Performance

- **Answer**: Partition data based on key fields (e.g., date, region) in data flows. This enables parallel processing, which improves data movement efficiency and performance.

22. Real-Time Data Ingestion to Synapse Analytics

- **Answer**: Use Event Hub and Stream Analytics to capture and process streaming data in near real-time, then load it to Synapse Analytics through ADF or Synapse integration.

23. Generating Unique Row Identifiers

- **Answer**: In data flows, use the Derived Column transformation to create unique identifiers, like UUIDs, or concatenate fields to produce a composite key.

24. Monitoring Data Pipeline Performance

- **Answer**: Leverage Azure Monitor and Log Analytics to track pipeline execution times, error rates, and resource consumption. Set up alerts for critical metrics.

25. Conditional Execution Based on Data Size

- **Answer**: Use Get Metadata to retrieve file size, and implement an If Condition activity to only proceed with data processing if the file meets a certain size threshold.

26. Data Validation before Loading

- **Answer**: Use data flows with conditional splits to validate incoming records based on defined criteria. Route valid records to the destination and send invalid records to a log or error table.

27. Transferring Data with Custom Encryption

- **Answer**: Encrypt data at the source or in-transit, and use ADF integration with Key Vault to decrypt and access keys as needed during data transfer.

28. Using Polybase for High-Volume Data Loads

- **Answer**: Enable Polybase in the Copy Activity for efficient bulk loading into Synapse Analytics, especially useful for handling terabyte-scale datasets.

29. Implementing Slowly Changing Dimensions (SCD) Type 2

- **Answer**: Use a data flow to add start and end dates for rows and update records based on key fields to retain historical data alongside current data.

30. Automating Pipeline Execution Based on Database Trigger

- **Answer**: Configure an Azure Logic App that listens to changes in an Azure SQL Database. Upon detecting a change, the Logic App triggers the ADF pipeline.

31. Data Masking for Privacy Requirements

- **Answer**: Use transformations in data flows to mask or obfuscate sensitive data, ensuring privacy before storing it in a target location.

32. Testing Pipeline in Dev vs. Prod Environments

- **Answer**: Use parameters to handle environment-specific configurations (e.g., database connection strings) and separate linked services for each environment. Link your ADF instance to a Git repository for source control to test in the Dev environment before deploying to Prod.

33. Migrating Pipelines Between Different Azure Subscriptions

- **Answer**: Export the pipeline as an ARM template and import it into the target subscription. Modify environment-specific configurations, such as storage accounts and linked services, to align with the new subscription settings.

34. Creating Fault Tolerance for Batch Jobs

- **Answer**: In the Copy Activity, enable fault tolerance by configuring retry options, including the retry count and intervals. Additionally, you can implement error-handling paths using activities like If Condition and store failed records for later analysis.

35. Extracting Metadata from Files in Blob Storage

- **Answer**: Use the Get Metadata activity to retrieve file properties like size, last modified date, and schema. This is especially useful for validating files before processing or for incremental data load scenarios.

36. Using Managed Identity for Secure Authentication

- **Answer**: Enable managed identity in ADF for secure access to Azure resources. Use this identity with Azure Key Vault to retrieve credentials and avoid hardcoding sensitive information like database credentials or storage keys.

37. Building Reusable Parameterized Pipelines

- **Answer**: Use global parameters or pipeline parameters to make pipelines more modular and reusable. For instance, you can parameterize connection strings, file paths, and table names, allowing the same pipeline to be used across different projects or data sources.

38. Splitting Large Files for Parallel Processing

- **Answer**: Use data partitioning within data flows or custom partition columns to split files into chunks based on row groups, dates, or specific identifiers. This allows ADF to process large files in parallel, improving data load efficiency.

39. Optimizing Cross-region Data Transfers

- **Answer**: Use regional integration runtimes close to the data source and target to minimize latency. Additionally, consider compressing data before transfer and using managed virtual networks for secure, optimized traffic between Azure regions.

40. Automated Data Archiving for Historical Records

- **Answer**: Set up a Copy Activity to move older data to an archive location (such as cold storage in ADLS) and delete it from the primary storage after successful transfer. Use a scheduled trigger to automate this process regularly, freeing up storage for newer data.

41. Dynamic Column Mapping in Copy Activity

- **Answer**: Use JSON-based column mapping in the Copy Activity to dynamically map columns between the source and destination. This is especially useful when source and destination columns don’t match exactly or when schema drift occurs.

42. Implementing Pipeline Versioning and Source Control

- **Answer**: Use Git integration in ADF to maintain version control for pipelines. Create branches for development, testing, and production, allowing changes to be tracked, tested, and deployed systematically.

43. Handling Null Values in Data

- **Answer**: In data flows, use the Conditional Split transformation to filter records with null values or use the Derived Column transformation to replace null values with default values before loading the data into the destination.

44. Parameterizing Connection Strings for Multiple Databases

- **Answer**: Set up parameters in linked services to handle different connection strings dynamically. Define database-specific configurations as pipeline parameters and pass them into linked services at runtime, allowing for flexible database connections within the same pipeline.

45. Scheduling Pipelines with Complex Recurrence Patterns

- **Answer**: Use custom recurrence settings in the pipeline trigger to schedule complex patterns, like running a pipeline every third day or at specific intervals. Alternatively, you can use Logic Apps to control ADF pipeline triggers for more granular scheduling requirements.

46. In-place Transformation Without Data Movement

- **Answer**: Load data into Synapse or Azure SQL Database, then perform transformations using stored procedures or T-SQL scripts directly on the target. This approach minimizes data movement and uses ELT (Extract, Load, Transform) principles for efficiency.

47. Using Stored Procedures in Azure SQL Database

- **Answer**: Use the Stored Procedure activity in ADF to call SQL scripts for data transformations. This can be used for tasks like data aggregation, cleanup, or applying complex business logic before moving data into final destinations.

48. Ensuring GDPR Compliance in Data Pipelines

- **Answer**: Use ADF features like data masking, data retention policies, and encryption at rest and in transit. Implement ADF's secure access features to limit who can view or edit pipeline configurations containing sensitive information.

49. Enabling Detailed Logging for Pipeline Monitoring

- **Answer**: Enable logging with Azure Monitor, Log Analytics, or Application Insights to track pipeline activities and errors in detail. Set up metrics and alerts to notify stakeholders about pipeline health, execution duration, or failed runs.

50. Pipeline Execution Adjustments for Holiday Schedules

- **Answer**: Use a custom trigger or Logic App to manage pipeline execution around holidays. You can configure the pipeline to skip specific dates or adjust execution frequency, helping to avoid processing data during non-operational days.

Prepared by: Hari.A | Founder – Rishab Informatica Group– www.rishabinformaticagroup.com
🔗 www.linkedin.com/in/hariinformatica | 🐦 https://x.com/hariinformatica

Folow on linked in / Visit : www.rishabinformaticagroup.com
PLEASE CONTACT US FOR MORE USFUL DOCS + 91-8970853557 / 9448005273