**Use Case**: Data Ingestion and Transformation

**Scenario**: A company wants to build a data pipeline to ingest and transform data from various sources into their Azure Data Lake Storage. The data will be used for analytics and reporting purposes.

**Solution using Azure Data Factory:**

**Data Source Configuration:**

Configure the necessary connections to the data sources such as databases, file systems, or APIs that contain the required data.

**Pipeline Creation:**

Create a new pipeline in Azure Data Factory to orchestrate the data ingestion and transformation process.

**Data Ingestion:**

Add a data ingestion activity to the pipeline to pull data from the various sources. This could involve extracting data from databases, reading files from storage systems, or consuming data from APIs.

Configure the data ingestion activity to specify the source dataset, the destination dataset in Azure Data Lake Storage, and any required transformations or filters.

**Data Transformation:**

Add data transformation activities to the pipeline to cleanse, transform, or enrich the ingested data as needed. Azure Data Factory supports various data transformation activities such as mapping data flows, executing SQL queries, or invoking custom code using Azure Functions.

**Data Storage:**

Define the output dataset in Azure Data Lake Storage where the transformed data will be stored.

**Data Orchestration:**

Configure the dependencies and sequencing of activities within the pipeline to ensure the data ingestion and transformation process is executed in the desired order.

**Scheduling and Monitoring:**

Schedule the pipeline to run at specific intervals or trigger it based on events or data availability.

Monitor the pipeline execution and track the progress, ensuring that data is ingested and transformed successfully.

**Error Handling and Retry:**

Implement error handling mechanisms and retries to handle any potential failures during the data ingestion or transformation process. Azure Data Factory provides built-in mechanisms for error handling and retry policies.

**Data Quality and Validation:**

Include data quality checks and validation activities within the pipeline to ensure the accuracy and integrity of the transformed data. This could involve performing data validation rules, checking for data anomalies, or comparing data against predefined thresholds.

**Notification and Alerts:**

Configure notifications or alerts to be triggered in case of pipeline failures, data quality issues, or any other exceptions that require attention.

**Sample Data For Above Use Case:**

****

**Data Sources:**

Source 1: CSV file for customer details stored in Azure Blob Storage(customer.csv)

Source 2: CSV file for sales details stored in Azure Blob Storage(sales.csv)

**Data Ingestion:**

Source 1: Customer data CSV file Azure Blob Storage

Source 2: Sales data CSV file in Azure Blob Storage

**Data Transformation:**

Cleanse and standardize customer data by removing duplicates and formatting names consistently.

Convert and aggregate sales data to calculate total sales by product and region.

**Data Storage:**

Destination: Azure Data Lake Storage account and container.

**Data Orchestration:**

Schedule the pipeline to run daily at a specific time to ingest and transform the latest data.

**Scheduling and Monitoring:**

Schedule the pipeline to run every day at 2 AM UTC.

Monitor the pipeline execution through Azure Data Factory's monitoring interface, which provides information on the status, execution time, and any errors or warnings.

**Error Handling and Retry:**

Implement retry policies for failed activities to handle temporary issues such as network errors or resource unavailability.

**Data Quality and Validation:**

Perform data validation checks to ensure the integrity and quality of the transformed data, such as checking for missing or invalid values, data type consistency, or referential integrity.

**Notification and Alerts:**

Configure email notifications to be sent to the data engineering team in case of pipeline failures or data quality issues.

**Sample Data Formats:**

1. Customers table in the SQL Server database:

Columns: CustomerID, FirstName, LastName, Email, Address, City, Country

1. Sales data CSV files in Azure Blob Storage:

Columns: OrderID, CustomerID, ProductID, Quantity, Price, OrderDate, Region

**Sample Transformations:**

Data Transformation Examples:

a. Cleanse and standardize customer data:

* Remove duplicate customer records based on the CustomerID from customer data.
* Standardize the formatting of customer names by capitalizing the first letter of each name and converting the rest to lowercase.
* Remove leading or trailing spaces from the email addresses.
* Format the address by removing unnecessary characters or special characters.

b. Convert and aggregate sales data:

* Group the sales data by product and region to calculate the total sales for each combination.
* Sum the quantity and calculate the total sales amount for each product and region.
* Calculate additional metrics such as average sales price, maximum sales quantity, or minimum sales price.

**Data Mapping:**

* Transform the "FirstName" and "LastName" columns from the Customers table to a new column called "FullName" by concatenating them.
* Map the "State" column values to their corresponding two-letter state abbreviations using a lookup table

**Filtering:**

* Filter the sales data to include only orders with a quantity greater than 10.
* Apply a filter to exclude orders with a price less than $10.
* Aggregation:

1. Calculate the average order quantity and total revenue per customer.

**Joining:**

* Join the Customers table with the Sales table based on the "CustomerID" column to enrich the sales data with customer information.

**Date Manipulation:**

* Extract the year from the "OrderDate" column and store it in a separate column.
* Calculate the number of days between the order date and the current date.

**Data Cleansing:**

* Remove any duplicate records from the sales dataset based on a combination of columns, such as OrderID and CustomerID.
* Cleanse the email addresses by removing any leading or trailing spaces and converting them to lowercase.

|  |  |
| --- | --- |
|  |  |