**Building an ETL/ELT Pipeline in Azure Data Factory**

**Objective**

You are working for an e-commerce company that wants to build a basic **SalesFact table** for reporting daily sales. The data comes from different systems:

* **Orders** in a CSV file stored in Blob Storage
* **Products** in a JSON file in ADLS Gen2
* **Customer details** in Azure SQL Database  
  Your task is to integrate these datasets using **Azure Data Factory** and load them into a **SalesFact table** in Azure Synapse/SQL DB.

**1. Setup**

1. Create an Azure **Blob Storage container** and upload:
   * orders.csv (5 sample rows provided).
   * products.json.
2. Ensure you have an **Azure SQL Database** with a Customer table (create + insert sample rows as given).
3. Create a **Target schema** in Synapse or SQL DB with:
   * SalesFact table.
   * ErrorLog table (for failed pipeline runs).

CREATE TABLE ErrorLog (

ErrorID INT IDENTITY(1,1) PRIMARY KEY,

ActivityName NVARCHAR(100),

ErrorMessage NVARCHAR(MAX),

ErrorTime DATETIME DEFAULT GETDATE()

);

**2. Linked Services & Datasets**

**Linked Services**

* Azure Blob Storage (Orders).
* ADLS Gen2 (Products).
* Azure SQL Database (Customer).
* Synapse SQL DB (Target).

**Datasets**

* OrdersCSV → Blob → orders.csv.
* ProductsJSON → ADLS → products.json.
* CustomerTable → SQL DB → Customer.
* SalesFact → Synapse → Target table.

**3. Pipeline Design**

1. **Copy Activity 1**: Copy orders.csv → Synapse Staging.
2. **Copy Activity 2**: Copy products.json → Synapse Staging.
3. **Copy Activity 3**: Copy Customer table → Synapse Staging.
4. **Stored Procedure Activity**: Join the three staging tables into SalesFact.

**Retry Policy**

* On each Copy Activity:
  + Retry = **3 times**
  + Retry interval = **60 seconds**

**Error Handling**

* Add an **On Failure branch** after each Copy Activity.
* Insert error details into ErrorLog using a SQL script activity.

INSERT INTO ErrorLog (ActivityName, ErrorMessage)

VALUES (@activityName, @errorMessage);

**4. Transformation (Source-to-Target Mapping)**

| **Source** | **Target (SalesFact)** |
| --- | --- |
| Orders.OrderID | SalesFact.OrderID |
| Customer.CustomerID | SalesFact.CustomerID |
| Customer.CustomerName | SalesFact.CustomerName |
| Product.ProductID | SalesFact.ProductID |
| Product.ProductName | SalesFact.ProductName |
| Product.Category | SalesFact.ProductCategory |
| Orders.OrderDate | SalesFact.OrderDate |
| Orders.OrderAmount | SalesFact.SalesAmount |

**Example Stored Procedure**

CREATE PROCEDURE LoadSalesFact

AS

BEGIN

INSERT INTO SalesFact (OrderID, CustomerID, CustomerName, ProductID, ProductName, ProductCategory, OrderDate, SalesAmount)

SELECT

o.OrderID,

c.CustomerID,

c.CustomerName,

p.ProductID,

p.ProductName,

p.Category,

o.OrderDate,

o.OrderAmount

FROM StagingOrders o

JOIN StagingCustomer c ON o.CustomerID = c.CustomerID

JOIN StagingProducts p ON o.ProductID = p.ProductID;

END

**5. Monitoring & Validation**

* Run the pipeline.
* Use the **ADF Monitor Hub** to track pipeline status.
* Validate results by querying:

SELECT \* FROM SalesFact;

SELECT \* FROM ErrorLog;