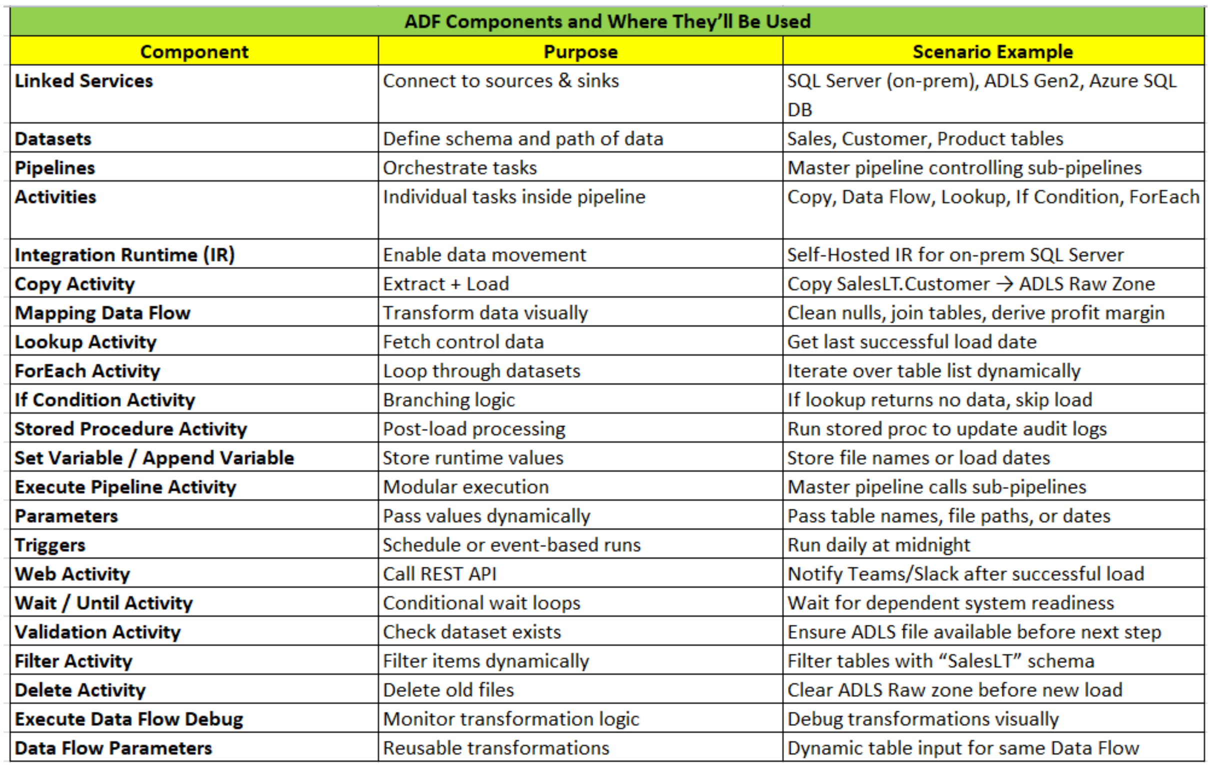
Adventure Works Data Integration and Analytics Pipeline

ADF pipeline pulls daily sales data from on-prim AdventureWorksLT2022, stages it in ADLS, cleans and transforms it with Mapping Data Flow, aggregates by region and customer, and loads it into Azure SQL for Power BI dashboards. The pipeline logs all runs, sends Teams alerts, and is fully parameterized.

Build an end-to-end ADF pipeline that:

1. Extracts data from your on-prim AdventureWorksLT2022 SQL Server.
2. Loads it into Azure Data Lake Storage (ADLS) as raw data (Bronze Layer).
3. Cleanses and transforms it into Silver and Gold layers using Data Flow.
4. Loads final curated data into Azure Synapse / Azure SQL Database for reporting.
5. Automates the workflow with Triggers, Parameters, Variables, Logging, and Monitoring.



ADF Components and Where They'll Be Used

Component	Purpose	Scenario Example
Linked Services	Connect to sources & sinks	SQL Server (on-prem), ADLS Gen2, Azure SQL DB
Datasets	Define schema and path of data	Sales, Customer, Product tables
Pipelines	Orchestrate tasks	Master pipeline controlling sub-pipelines
Activities	Individual tasks inside pipeline	Copy, Data Flow, Lookup, If Condition, ForEach
Integration Runtime (IR)	Enable data movement	Self-Hosted IR for on-prem SQL Server
Copy Activity	Extract + Load	Copy SalesLT.Customer → ADLS Raw Zone
Mapping Data Flow	Transform data visually	Clean nulls, join tables, derive profit margin
Lookup Activity	Fetch control data	Get last successful load date
ForEach Activity	Loop through datasets	Iterate over table list dynamically
If Condition Activity	Branching logic	If lookup returns no data, skip load
Stored Procedure Activity	Post-load processing	Run stored proc to update audit logs
Set Variable / Append Variable	Store runtime values	Store file names or load dates
Execute Pipeline Activity	Modular execution	Master pipeline calls sub-pipelines
Parameters	Pass values dynamically	Pass table names, file paths, or dates
Triggers	Schedule or event-based runs	Run daily at midnight
Web Activity	Call REST API	Notify Teams/Slack after successful load
Wait / Until Activity	Conditional wait loops	Wait for dependent system readiness
Validation Activity	Check dataset exists	Ensure ADLS file available before next step
Filter Activity	Filter items dynamically	Filter tables with "SalesLT" schema
Delete Activity	Delete old files	Clear ADLS Raw zone before new load
Execute Data Flow Debug	Monitor transformation logic	Debug transformations visually
Data Flow Parameters	Reusable transformations	Dynamic table input for same Data Flow

Pipeline Architecture (3-Layer Medallion Style):

1. Bronze Layer (Raw Data)
   Goal: Extract from SQL Server → ADLS
   Tables:
   SalesLT.Customer, SalesLT.Product, SalesLT.SalesOrderHeader, SalesLT.SalesOrderDetail

   Activities:

- Lookup: Get last load date
- Copy Activity: Incremental load → /adls/bronze/{table}/{yyyymmdd}.csv
- Stored Procedure: Log load metadata

2. Silver Layer (Cleansed Data)
   Goal: Clean, standardize, and join tables
   Data Flow:
   - Join SalesOrderHeader + SalesOrderDetail
   - Derive TotalAmount, ProfitMargin
   - Filter null CustomerIDs
   - Sink to /adls/silver/sales/

3. Gold Layer (Aggregated / Reporting Data)
   Goal: Create curated reporting dataset
   Data Flow:
   - Aggregate by Customer → Total Sales, Average Order Value
   - Sink: Azure SQL Database table dbo.CustomerSalesSummary

Master Pipeline Structure

Pipeline Name: PL_Master_AdventureWorksETL

Steps:

1. Lookup Tables List
2. ForEach Table Loop
3. Execute → PL_Bronze_Load
4. Execute Pipeline: PL_Silver_Transform
5. Execute Pipeline: PL_Gold_Load
6. Web Activity: Send Success Notification
7. If Condition (Failure): Send Failure Notification

Optional Enhancements (Advanced)

Feature	Description
Metadata-driven pipeline	Use SQL config table with source/target info to loop
Parameterization	Table name, file path, incremental date
Logging Table	Store pipeline run details (RunID, TableName, Status,
Integration with Synapse / Power BI	Build Power BI Dashboard using Gold layer
Event Trigger	Trigger pipeline when new file lands in ADLS
Data Validation	Compare row counts between source and sink