# Azure Data Factory Advanced Hands-on Labs

This lab guide provides detailed step-by-step exercises for advanced ADF concepts including Mapping Data Flows, parameterization, control flows, incremental loads, and Git integration.

## Lab 1: Mapping Data Flow – Join and Aggregate

### Objective:

Create a mapping data flow to join two datasets and perform aggregations.

### Steps:

1. Open **ADF Studio** and go to the **Author** hub.
2. Under **Factory Resources**, click **+ Data Flow** → **Mapping Data Flow**.
3. Add two **Source** transformations:
   * Source 1: Employee\_CSV\_Dataset (EmployeeID, DepartmentID, Salary).
   * Source 2: Department\_JSON\_Dataset (DepartmentID, DepartmentName).
4. Add a **Join** transformation:
   * Join on DepartmentID from both datasets.
   * Join type: Inner.
5. Add an **Aggregate** transformation:
   * Group by DepartmentName.
   * Aggregate: avg(Salary) as AvgSalary, count(EmployeeID) as EmpCount.
6. Add a **Sink** transformation:
   * Sink to Azure SQL Database table DeptSalarySummary.
7. Debug and publish the Data Flow.
8. Trigger pipeline to execute.

## Lab 2: Parameterized Pipeline – Dynamic File Names

### Objective:

Pass file names dynamically to datasets for reusability.

### Steps:

1. In **Author** hub, create a new **Pipeline** named ParameterizedFilePipeline.
2. Add a **Pipeline Parameter** named FileName (type: String).
3. Edit dataset Employee\_CSV\_Dataset:
   * Parameterize the file path.
   * Create dataset parameter FileNameParam.
   * Use @dataset().FileNameParam in file path.
4. In pipeline, set dataset parameter mapping:
   * Map FileNameParam = @pipeline().parameters.FileName.
5. Add a **Copy Data** activity using this dataset.
6. Trigger pipeline manually:
   * Pass employee\_data\_2023.csv as parameter.
   * Verify it dynamically picks the file.

## Lab 3: Control Flow – ForEach Loop for Multiple Files

### Objective:

Process multiple files in a folder using ForEach.

### Steps:

1. In ParameterizedFilePipeline, add a **Get Metadata** activity:
   * Point to ADLS folder containing multiple CSV files.
   * Configure **Child Items**.
2. Add a **ForEach** activity:
   * Items: @activity('Get Metadata1').output.childItems.
   * Inside ForEach, add a **Copy Data** activity.
   * Source dataset: Parameterized CSV dataset.
   * Map dataset parameter: @item().name.
3. Publish and trigger pipeline.
4. Verify that all files in the folder are copied one by one.

## Lab 4: Incremental Load – Watermark Strategy

### Objective:

Implement incremental loading from SQL to ADLS based on a watermark column.

### Steps:

1. In SQL source table Employee, ensure there is a column LastModifiedDate.
2. In ADF, create a **Pipeline Parameter** named WatermarkValue.
3. In SQL dataset, parameterize query:

* SELECT \* FROM Employee WHERE LastModifiedDate > '@{pipeline().parameters.WatermarkValue}'

1. Create a **Copy Data** activity:
   * Source: SQL dataset with query filter.
   * Sink: ADLS folder incremental\_load/.
2. Store the latest LastModifiedDate after each run (could use Lookup activity and Set Variable).
3. Test by running pipeline with different watermark values.

## Lab 5: Git Integration – CI/CD Setup

### Objective:

Connect ADF to Git repository for source control.

### Steps:

1. In ADF Studio, go to **Manage** hub.
2. Select **Git Configuration**.
3. Click **Configure Git repository**.
4. Choose **Azure DevOps Git** or **GitHub**.
5. Provide:
   * Repository URL.
   * Collaboration branch (e.g., main or master).
   * Root folder path.
6. Save configuration.
7. In **Author** hub, make a pipeline change (e.g., new activity).
8. Click **Save All** → Commit message → **Commit to branch**.
9. Verify commit appears in Git repository.

# End of Labs

You have now successfully: - Built mapping data flows with joins and aggregations. - Created parameterized pipelines for dynamic datasets. - Processed multiple files with ForEach loops. - Implemented incremental load with watermark strategy. - Integrated ADF with Git for CI/CD workflows.