Data Migration Project

Project Overview

This project demonstrates a **data migration process** in SQL Server, where data from the HumanResources.Employee table of the **AdventureWorks database** is migrated into a newly created table named EmployeeMigration. The project covers **table creation, indexing, data transfer, and transaction management** to ensure consistency and reliability.

Problem Statement

- Create a new table EmployeeMigration with the required schema.
- Apply constraints:
  - EmployeeID → Primary Key
  - NationalIDNumber → Unique
- Create a **non-clustered index** on the column JobTitle.
- Migrate employee data with:
  - **HireDate ≥ 2008**
  - **Exclude BusinessEntityID 224, 234, 250**
- Use a **transaction**:
  - Commit if successful
  - Rollback if any error occurs

Solution Steps:

Step 1: Table Creation

```
CREATE TABLE EmployeeMigration (

    EmployeeID INT PRIMARY KEY NOT NULL,

    NationalIDNumber NVARCHAR(40) NOT NULL UNIQUE,

    JobTitle NVARCHAR(200),

    Department NVARCHAR(50),

    Shift NVARCHAR(20),
    HireDate DATETIME,

    ModifiedDate DATETIME

);
```

Step 2: Create Index

```
CREATE NONCLUSTERED INDEX
IX_EmployeeMigration_JobTitle

ON EmployeeMigration (JobTitle);
```

Step 3: Data Migration with Transaction

```
BEGIN TRANSACTION;

BEGIN TRY

    INSERT INTO EmployeeMigration (EmployeeID,
NationalIDNumber, JobTitle,

                                   Department, Shift, HireDate,
ModifiedDate)

    SELECT

        e.BusinessEntityID AS EmployeeID,

        e.NationalIDNumber,

        e.JobTitle,

        d.Name AS Department,

        s.Name AS Shift,
        e.HireDate,
        e.ModifiedDate
    FROM HumanResources.Employee e
    JOIN HumanResources.EmployeeDepartmentHistory edh
        ON e.BusinessEntityID = edh.BusinessEntityID
    JOIN HumanResources.Department d
        ON edh.DepartmentID = d.DepartmentID
    JOIN HumanResources.Shift s
        ON edh.ShiftID = s.ShiftID
    WHERE YEAR(e.HireDate) >= 2008
      AND e.BusinessEntityID NOT IN (224, 234, 250)
      AND edh.EndDate IS NULL; -- Only current department
records
    COMMIT TRANSACTION;
    PRINT 'Data migration successful!';
END TRY
BEGIN CATCH
    ROLLBACK TRANSACTION;
    PRINT 'Error occurred. Transaction rolled back!';
```

```
END CATCH;
```

Key Learnings

- Database design with **constraints and indexing**.
- Handling **real-time data migration** using **transactions**.
- Ensuring **data consistency** and rollback on errors.
- Practical SQL skills applied on **AdventureWorks DB**.

Tools & Technologies

- SQL Server
- AdventureWorks Database

Expected Output

- Successfully migrated employee data with conditions applied.
- Indexed JobTitle for faster query performance.
- Rollback mechanism for handling errors gracefully.