Project 3 - Database Design and Implementation

Group 3 - *Damian Rozpedowski, Essmer Sanchez, Hannah Kurian, Hasnatul Hosna*

Stored Procedures

```
     -- =============================================
     -- Author:      Group Three
     -- Create date: 05-12-24
     -- Description: Creates a table to show the process of removing

     --              foreign keys, truncating, loading, and adding
foreign keys
     -- =============================================
ALTER   PROCEDURE [Process].[usp_TrackWorkFlow]
    -- Add the parameters for the stored procedure here
    @WorkflowDescription NVARCHAR(100),
    @WorkFlowStepTableRowCount INT,
    @StartingDateTime DATETIME2,
    @EndingDateTime DATETIME2,
    @UserAuthorizationKey INT
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    -- Insert statements for procedure here
    INSERT INTO [Process].[WorkflowSteps]
        (
        WorkFlowStepDescription,
        WorkFlowStepTableRowCount,
        StartingDateTime,
        EndingDateTime,
        UserAuthorizationKey
        )
    VALUES
        (@WorkflowDescription,
        @WorkFlowStepTableRowCount,
        @StartingDateTime,
        @EndingDateTime,
        @UserAuthorizationKey);
```

```
END;

      -- =============================================
      -- Author:      Group Three
      -- Create date: 05-12-24
      -- Description:  Displays the Workflow steps table
      -- =============================================
ALTER PROCEDURE [Process].[usp_ShowWorkflowSteps]
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;
    SELECT *
    FROM [Process].[WorkFlowSteps];
END

      -- =============================================
      -- Author:      Group Three
      -- Create date: 05-12-24
      -- Description:  Add Foreign Keys to the Schema
      -- =============================================
ALTER PROCEDURE [Project3].[AddForeignKeys]
    @UserAuthorizationKey INT
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    DECLARE @StartingDateTime DATETIME2 = SYSDATETIME();

    ALTER TABLE [Process].[WorkflowSteps]
    ADD CONSTRAINT FK_WorkFlowSteps_UserAuthorization
        FOREIGN KEY (UserAuthorizationKey)
        REFERENCES [DbSecurity].[UserAuthorization]
(UserAuthorizationKey);

    ALTER TABLE [CollegeClasses].[Course]
    ADD CONSTRAINT FK_Course_UserAuthorization
        FOREIGN KEY (UserAuthorizationKey)
        REFERENCES [DbSecurity].[UserAuthorization]
(UserAuthorizationKey);

    ALTER TABLE [CollegeClasses].[Course]
    ADD CONSTRAINT FK_Course_DepartmentID
        FOREIGN KEY (DepartmentID)
        REFERENCES [Departmental].[Department] (DepartmentID);

    ALTER TABLE Departmental.[Department]
    ADD CONSTRAINT FK_Department_UserAuthorization
```

```
        FOREIGN KEY (UserAuthorizationKey)
        REFERENCES [DbSecurity].[UserAuthorization]
(UserAuthorizationKey);

    ALTER TABLE [CollegeClasses].[ModeOfInstruction]
    ADD CONSTRAINT FK_ModeOfInst_UserAuthorization
        FOREIGN KEY([UserAuthorizationKey])
        REFERENCES [DbSecurity].[UserAuthorization]
([UserAuthorizationKey]);

    ALTER TABLE [Location].[RoomLocation]
    ADD CONSTRAINT FK_RoomLocation_UserAuthorization
        FOREIGN KEY([UserAuthorizationKey])
        REFERENCES [DbSecurity].[UserAuthorization]
([UserAuthorizationKey]);

    ALTER TABLE [Location].[BuildingLocation]
    ADD CONSTRAINT FK_BuildingLocation_UserAuthorization
        FOREIGN KEY (UserAuthorizationKey)
        REFERENCES [DbSecurity].[UserAuthorization]
(UserAuthorizationKey);

    ALTER TABLE [CollegeClasses].[Class]
    ADD CONSTRAINT FK_Class_UserAuthorization
        FOREIGN KEY (UserAuthorizationKey)
        REFERENCES [DbSecurity].[UserAuthorization]
(UserAuthorizationKey);

    ALTER TABLE [CollegeClasses].[Class]
    ADD CONSTRAINT FK_Class_Course
        FOREIGN KEY (CourseID)
        REFERENCES [CollegeClasses].[Course] (CourseID);

    ALTER TABLE [CollegeClasses].[Class]
    ADD CONSTRAINT FK_Class_Instructor
        FOREIGN KEY (InstructorID)
        REFERENCES [Departmental].[Instructor] (InstructorID);

    ALTER TABLE [CollegeClasses].[Class]
    ADD CONSTRAINT FK_Class_RoomLocation
        FOREIGN KEY (RoomID)
        REFERENCES [Location].[RoomLocation] (RoomID);

    ALTER TABLE [CollegeClasses].[Class]
    ADD CONSTRAINT FK_Class_ModeOfInstruction
        FOREIGN KEY (ModeID)
        REFERENCES [CollegeClasses].[ModeOfInstruction] (ModeID);

    -- add more here...
    DECLARE @WorkFlowStepTableRowCount INT;
    SET @WorkFlowStepTableRowCount = 0;
    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();
    DECLARE @QueryTime BIGINT = CAST(DATEDIFF(MILLISECOND,
@StartingDateTime, @EndingDateTime) AS bigint);
    EXEC [Process].[usp_TrackWorkFlow] 'Add Foreign Keys',
                                       @WorkFlowStepTableRowCount,
                                       @StartingDateTime,
                                       @EndingDateTime,
                                       @UserAuthorizationKey;
END;

     -- ===============================================
     -- Author:      Group Three
     -- Create date: 05-12-24
     -- Description:  Drop Foreign Keys to the Schema
     -- ===============================================
ALTER PROCEDURE [Project3].[DropForeignKeys]
    @UserAuthorizationKey INT
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @StartingDateTime DATETIME2 = SYSDATETIME();

    -- Dropping foreign key constraints using IF EXISTS
        ALTER TABLE [Process].[WorkflowSteps]
    DROP CONSTRAINT IF EXISTS [FK_WorkFlowSteps_UserAuthorization];

    ALTER TABLE [CollegeClasses].[Course]
    DROP CONSTRAINT IF EXISTS [FK_Course_UserAuthorization];

    ALTER TABLE [CollegeClasses].[Course]
    DROP CONSTRAINT IF EXISTS [FK_Course_DepartmentID];

    ALTER TABLE [CollegeClasses].[ModeOfInstruction]
    DROP CONSTRAINT IF EXISTS [FK_ModeOfInst_UserAuthorization];

    ALTER TABLE [Location].[RoomLocation]
    DROP CONSTRAINT IF EXISTS [FK_RoomLocation_UserAuthorization];

    ALTER TABLE [Location].[RoomLocation]
    DROP CONSTRAINT IF EXISTS [FK_RoomLocation_BuildingCode];

    ALTER TABLE [Departmental].[Department]
    DROP CONSTRAINT IF EXISTS [FK_Department_UserAuthorization];

    ALTER TABLE [Location].[BuildingLocation]
    DROP CONSTRAINT IF EXISTS [FK_BuildingLocation_UserAuthorization];
    ALTER TABLE [CollegeClasses].[Class]
    DROP CONSTRAINT IF EXISTS [FK_Class_UserAuthorization];

    ALTER TABLE [CollegeClasses].[Class]
    DROP CONSTRAINT IF EXISTS [FK_Class_Course];

    ALTER TABLE [CollegeClasses].[Class]
    DROP CONSTRAINT IF EXISTS [FK_Class_Section];

    ALTER TABLE [CollegeClasses].[Class]
    DROP CONSTRAINT IF EXISTS [FK_Class_Instructor];

    ALTER TABLE [CollegeClasses].[Class]
    DROP CONSTRAINT IF EXISTS [FK_Class_RoomLocation];

    ALTER TABLE [CollegeClasses].[Class]
    DROP CONSTRAINT IF EXISTS [FK_Class_ModeOfInstruction];


    -- Tracking workflow execution
    DECLARE @WorkFlowStepTableRowCount INT;
    SET @WorkFlowStepTableRowCount = 0;
    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();
    DECLARE @QueryTime BIGINT = CAST(DATEDIFF(MILLISECOND,
@StartingDateTime, @EndingDateTime) AS bigint);
    EXEC [Process].[usp_TrackWorkFlow] 'Drop Foreign Keys',
                                       @WorkFlowStepTableRowCount,
                                       @StartingDateTime,
                                       @EndingDateTime,
                                       @UserAuthorizationKey;
END;

     -- =============================================
     -- Author:      Hosna Hasnatul
     -- Create date: 05-12-24
     -- Description: Loads Data into the Building Location Table
     -- =============================================
ALTER PROCEDURE [Project3].[LoadBuildingLocation]
         @UserAuthorizationKey INT
AS
BEGIN
     DECLARE @DateAdded DATETIME2;
     SET @DateAdded = SYSDATETIME();

    DECLARE @DateOfLastUpdate DATETIME2;
     SET @DateOfLastUpdate = SYSDATETIME();

     DECLARE @start AS DATETIME2, @end AS DATETIME2;
    SET @start = SYSDATETIME();
```

```
     SET NOCOUNT ON;

         -- Inserting building locations based on the Location column
from Uploadfile.CurrentSemesterCourseOfferings
    INSERT INTO [Location].[BuildingLocation] ([BuildingName],
[UserAuthorizationKey], [DateAdded], [DateOfLastUpdate])
    SELECT DISTINCT
           CASE
                WHEN CHARINDEX(' ', Location) > 0 THEN LEFT(Location,
CHARINDEX(' ', Location) - 1)  -- Extract building name
                ELSE NULL
           END,
        @UserAuthorizationKey,
        @DateAdded,
        @DateOfLastUpdate
    FROM Uploadfile.CurrentSemesterCourseOfferings
    WHERE Location IS NOT NULL AND CHARINDEX(' ', Location) > 0 AND
NOT EXISTS (
        SELECT 1
        FROM [Location].[BuildingLocation] AS BL
        WHERE BL.BuildingName = CASE
                                WHEN CHARINDEX(' ', Location) > 0 THEN
LEFT(Location, CHARINDEX(' ', Location) - 1)
                                ELSE Location
                                           END
    ); -- Ensures no duplicates are inserted

         -- Log the action
    DECLARE @RowCount INT = @@ROWCOUNT;
    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();
    EXEC [Process].[usp_TrackWorkFlow]
        @WorkflowDescription = 'Load Building Location Data',
        @WorkFlowStepTableRowCount = @RowCount,
        @StartingDateTime = @DateAdded,
        @EndingDateTime = @EndingDateTime,
        @UserAuthorizationKey = @UserAuthorizationKey;
END;

     -- =============================================
     -- Author:      Sanchez    Essmer
     -- Create date: 05-12-24
     -- Description:  Loads Data into the Class Table
     -- =============================================
ALTER PROCEDURE [Project3].[LoadClass]
    @UserAuthorizationKey INT
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @DateAdded DATETIME2 = SYSDATETIME();
    -- Inserting data into Class table
    INSERT INTO [CollegeClasses].[Class] (
        [CourseID],
        [CourseAbbreviation],
        [CourseNumber],
        [CourseName],
        [SectionID],
        [InstructorID],
        [Instructor],
        [BuildingID],
        [BuildingName],
        [RoomID],
        [RoomNumber],
        [Time],
        [Day],
        [ModeID],
        [ModeOfInstruction],
        [Enrolled],
        [Limit],
        [UserAuthorizationKey],
        [DateAdded],
        [DateOfLastUpdate]
    )
    SELECT
        C.CourseID,
        C.CourseAbbreviation,
        C.CourseNumber,
        C.CourseName,
        CASE
            WHEN ISNUMERIC(CS.[Sec]) = 1 AND CS.[Sec] NOT LIKE '%.%'
THEN CAST(CS.[Sec] AS INT)
            ELSE ROW_NUMBER() OVER (ORDER BY (SELECT NULL))
        END AS [SectionID],
        I.InstructorID,
        ISNULL(I.FullName, 'TBA') AS [Instructor],
        BL.BuildingID,
        BL.BuildingName,
        RL.RoomID,
        RL.RoomNumber,
        CS.[Time],
        CS.[Day],
        MI.ModeID,
        MI.[ModeOfInstruction],
        CAST(CS.[Enrolled] AS INT),
        CAST(CS.[Limit] AS INT),
        @UserAuthorizationKey,
        @DateAdded,
        @DateAdded
    FROM
        [Uploadfile].[CurrentSemesterCourseOfferings] CS
    LEFT JOIN
        [CollegeClasses].[ModeOfInstruction] MI
        ON CS.[Mode of Instruction] = MI.[ModeOfInstruction]
    JOIN
        [CollegeClasses].[Course] C
        ON C.CourseNumber = SUBSTRING(CS.[Course (hr, crd)],
PATINDEX('%[0-9]%', CS.[Course (hr, crd)]), 3)
        AND C.CourseAbbreviation = LEFT(CS.[Course (hr, crd)],
CHARINDEX(' ', CS.[Course (hr, crd)]) - 1)
    JOIN
        [Location].[BuildingLocation] BL
        ON BL.BuildingName = CASE WHEN CHARINDEX(' ', CS.Location) > 0
THEN LEFT(CS.Location, CHARINDEX(' ', CS.Location) - 1) ELSE NULL END
    JOIN
        [Location].[RoomLocation] RL
        ON RL.BuildingCode = BL.BuildingID
           AND RL.RoomNumber = CASE WHEN CHARINDEX(' ', CS.Location) >
0 THEN SUBSTRING(CS.Location, CHARINDEX(' ', CS.Location) + 1,
LEN(CS.Location)) ELSE NULL END
    LEFT JOIN
        [Departmental].[Instructor] I
        ON LTRIM(RTRIM(I.FullName)) = LTRIM(RTRIM(CS.[Instructor]));

    -- Log the action
    DECLARE @RowCount INT = @@ROWCOUNT;
    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();
    EXEC [Process].[usp_TrackWorkFlow]
        'Load Class Data',
        @RowCount,
        @DateAdded,
        @EndingDateTime,
        @UserAuthorizationKey;
END;

     -- =============================================
     -- Author:      Rozpedowski     Damian
     -- Create date: 05-12-24
     -- Description:  Loads Data into the Course Table
     -- =============================================
ALTER   PROCEDURE [Project3].[LoadCourse] @UserAuthorizationKey INT
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @DateAdded DATETIME2 = SYSDATETIME();
    DECLARE @StartingDateTime DATETIME2 = SYSDATETIME();

    INSERT INTO [CollegeClasses].[Course](
        [CourseAbbreviation] -- Course (parse letters)
        ,[CourseNumber] -- Course (parse number)
        ,[CourseCredit] -- Course (parse second number in (,))
        ,[CreditHours] -- Course (parse first number in (,))
        ,[CourseName] -- Description
        ,[DepartmentID] -- fk
        ,UserAuthorizationKey
        ,DateAdded
    )
    SELECT DISTINCT
        LEFT([Course (hr, crd)], PATINDEX('%[ (]%', [Course (hr,
crd)]) - 1) -- CourseAbbreviation
        ,SUBSTRING(
                [Course (hr, crd)],
                PATINDEX('%[0-9]%', [Course (hr, crd)]),
                CHARINDEX('(', [Course (hr, crd)]) - PATINDEX('%[0-
9]%', [Course (hr, crd)])
            ) -- CourseNumber
        ,CAST(SUBSTRING(
                [Course (hr, crd)],
                CHARINDEX(',', [Course (hr, crd)]) + 2,
                CHARINDEX(')', [Course (hr, crd)]) - CHARINDEX(',',
[Course (hr, crd)]) - 2
                ) AS FLOAT) --CourseCredit
        ,CAST(SUBSTRING(
                [Course (hr, crd)],
                CHARINDEX('(', [Course (hr, crd)]) + 1,
                CHARINDEX(',', [Course (hr, crd)]) - CHARINDEX('(',
[Course (hr, crd)]) - 1
                ) AS FLOAT) -- CreditHours
        ,C.Description -- CourseName
        , ( SELECT TOP 1 D.DepartmentID
            FROM [Departmental].[Department] AS D
            WHERE D.DepartmentName = LEFT([Course (hr, crd)],
PATINDEX('%[ (]%', [Course (hr, crd)]) - 1))
        ,@UserAuthorizationKey
        ,@DateAdded
    FROM
    [Uploadfile].[CurrentSemesterCourseOfferings] AS C;

    DECLARE @WorkFlowStepTableRowCount INT;
    SET @WorkFlowStepTableRowCount = (
                                       SELECT COUNT(*)
                                       FROM [CollegeClasses].[Course]
                                       );
    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();
    EXEC [Process].[usp_TrackWorkFlow] 'Load Course Data',
                                          @WorkFlowStepTableRowCount,
                                          @StartingDateTime,
                                          @EndingDateTime,
```

```sql
@UserAuthorizationKey;
END;

      -- =============================================
      -- Author:      Sanchez    Essmer
      -- Create date: 05-12-24
      -- Description:  Loads Data into the Department Table
      -- =============================================
ALTER PROCEDURE [Project3].[LoadDepartments] @UserAuthorizationKey INT

AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @DateAdded DATETIME2 = SYSDATETIME();
    DECLARE @StartingDateTime DATETIME2 = SYSDATETIME();

    INSERT INTO [Departmental].[Department] (
        DepartmentName, UserAuthorizationKey, DateAdded
    )
    SELECT DISTINCT
        LEFT([Course (hr, crd)], CHARINDEX(' ', [Course (hr, crd)]) -
1) AS DepartmentName,
        @UserAuthorizationKey,
        @DateAdded
    FROM [Uploadfile].[CurrentSemesterCourseOfferings]
    ORDER BY DepartmentName

    DECLARE @WorkFlowStepTableRowCount INT;
    SET @WorkFlowStepTableRowCount = (
                                    SELECT COUNT(*)
                                    FROM [Departmental].[Department]
                                    );

    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();

    EXEC [Process].[usp_TrackWorkFlow] 'Load Department Data',
                                       @WorkFlowStepTableRowCount,
                                       @StartingDateTime,
                                       @EndingDateTime,

@UserAuthorizationKey;
END;

      -- =============================================
      -- Author:      Kurian Hannah
      -- Create date: 05-12-24
      -- Description:  Loads Data into the Instructor Table
      -- =============================================
ALTER PROCEDURE [Project3].[LoadInstructor]
    @UserAuthorizationKey INT
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @DateAdded DATETIME2 = SYSDATETIME();
    DECLARE @DateOfLastUpdate DATETIME2 = SYSDATETIME();
    DECLARE @StartingDateTime DATETIME2 = SYSDATETIME();

    -- Temporary table to extract instructor data
    CREATE TABLE #TempInstructor (
        LastName NVARCHAR(50),
        FirstName NVARCHAR(50),
        DepartmentName CHAR(5),
        FullName NVARCHAR(101)
    );

    -- Populate Temporary table with instructor data
    INSERT INTO #TempInstructor (LastName, FirstName, DepartmentName,
FullName)
    SELECT DISTINCT
        TRIM(COALESCE(NULLIF(
            SUBSTRING(Instructor, 1,
                CASE WHEN CHARINDEX(',', Instructor) = 0 THEN
LEN(Instructor)
                     ELSE CHARINDEX(',', Instructor) - 1 END), ''),
'TBA')) AS LastName,
        TRIM(COALESCE(NULLIF(
            SUBSTRING(Instructor, CHARINDEX(' ', Instructor) + 1,
LEN(Instructor)), ''), 'TBA')) AS FirstName,
        LEFT([Course (hr, crd)], CHARINDEX(' ', [Course (hr, crd)]) -
1) AS DepartmentName,
        TRIM(COALESCE(NULLIF(Instructor, ''), 'TBA')) AS FullName
    FROM [Uploadfile].[CurrentSemesterCourseOfferings]
    WHERE Instructor IS NOT NULL AND LEN(Instructor) > 0;

    -- Filter out rows with invalid names
    DELETE FROM #TempInstructor
    WHERE LastName = 'TBA' AND FirstName = 'TBA'
       OR FirstName = ','
       OR FirstName = ''
       OR LastName = ''
       OR DepartmentName = '';

    -- Temporary table to hold department data
    CREATE TABLE #TempDepartment (
        DepartmentID INT,
        DepartmentName CHAR(5)
    );

    -- Populate Temporary table with department data
    INSERT INTO #TempDepartment (DepartmentID, DepartmentName)
    SELECT DepartmentID, DepartmentName
    FROM [Departmental].[Department];

    -- Insert into main Instructor table using Temporary table data
    INSERT INTO [Departmental].[Instructor] (FirstName, LastName,
DepartmentName, DepartmentID, FullName, UserAuthorizationKey,
DateAdded, DateOfLastUpdate)
    SELECT DISTINCT
        i.FirstName, i.LastName, i.DepartmentName, d.DepartmentID,
i.FullName, @UserAuthorizationKey, @DateAdded, @DateOfLastUpdate
    FROM #TempInstructor i
    JOIN #TempDepartment d ON i.DepartmentName = d.DepartmentName
    WHERE NOT EXISTS (
        SELECT 1
        FROM [Departmental].[Instructor] di
        WHERE di.FirstName = i.FirstName
          AND di.LastName = i.LastName
          AND di.DepartmentName = i.DepartmentName
          AND di.FullName = i.FullName
    );

    -- Drop temporary tables
    DROP TABLE #TempInstructor;
    DROP TABLE #TempDepartment;

    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();
    DECLARE @WorkFlowStepTableRowCount INT;

    -- Get row count for tracking workflow
    SELECT @WorkFlowStepTableRowCount = COUNT(*) FROM [Departmental].
[Instructor];

    -- Track workflow
    EXEC [Process].[usp_TrackWorkFlow] 'Load Instructor Data',
                                       @WorkFlowStepTableRowCount,
                                       @StartingDateTime,
                                       @EndingDateTime,
                                       @UserAuthorizationKey;
END;

     -- ============================================
     -- Author:      Rozpedowski     Damian
     -- Create date: 05-12-24
     -- Description:  Loads Data into the Mode of Instruction Table
     -- ============================================
ALTER PROCEDURE [Project3].[LoadModeOfInstruction]
```

```
    @UserAuthorizationKey INT
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @DateAdded DATETIME2 = SYSDATETIME();
    DECLARE @StartingDateTime DATETIME2 = SYSDATETIME();


    INSERT INTO [CollegeClasses].[ModeOfInstruction] (
        [ModeOfInstruction],
        [UserAuthorizationKey],
        [DateAdded],
        [DateOfLastUpdate]

    )
    SELECT DISTINCT
        [Mode of Instruction],
        @UserAuthorizationKey,
        @DateAdded,
        @DateAdded

    FROM [Uploadfile].[CurrentSemesterCourseOfferings]


    DECLARE @WorkFlowStepTableRowCount INT;
    SET @WorkFlowStepTableRowCount = (
        SELECT COUNT(*)
        FROM [CollegeClasses].[ModeOfInstruction]
    );

    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();

    EXEC [Process].[usp_TrackWorkFlow] 'Load Mode of Instruction
Data',
                                       @WorkFlowStepTableRowCount,
                                       @StartingDateTime,
                                       @EndingDateTime,
                                       @UserAuthorizationKey;
END;

     -- =============================================
     -- Author:      Kurian     Hannah
     -- Create date: 05-12-24
     -- Description:  Loads Data into the Room Location Table
     -- =============================================
ALTER   PROCEDURE [Project3].[LoadRoomLocation]
         @UserAuthorizationKey INT
AS
BEGIN
```

```sql
      SET NOCOUNT ON;
      DECLARE @DateAdded DATETIME2;
      SET @DateAdded = SYSDATETIME();

    DECLARE @DateOfLastUpdate DATETIME2;
      SET @DateOfLastUpdate = SYSDATETIME();

      DECLARE @start AS DATETIME2, @end AS DATETIME2;
    SET @start = SYSDATETIME();

    -- Temporary table to hold parsed Location data
    IF OBJECT_ID('tempdb..#RoomData') IS NOT NULL
        DROP TABLE #RoomData;

    CREATE TABLE #RoomData (
        BuildingName VARCHAR(255),
        RoomNumber VARCHAR(12)
    );

    -- Parse Location into BuildingName and RoomNumber
    INSERT INTO #RoomData (BuildingName, RoomNumber)
    SELECT
        LEFT(Location, CHARINDEX(' ', Location) - 1) AS BuildingName,
        SUBSTRING(Location, CHARINDEX(' ', Location) + 1,
LEN(Location)) AS RoomNumber
    FROM Uploadfile.CurrentSemesterCourseOfferings
    WHERE
        Location IS NOT NULL AND
        CHARINDEX(' ', Location) > 0 AND
        LTRIM(RTRIM(LEFT(Location, CHARINDEX(' ', Location) - 1))) <>
'' AND
        LTRIM(RTRIM(SUBSTRING(Location, CHARINDEX(' ', Location) + 1,
LEN(Location)))) <> '';
      -----------------------------------------------------------------
------------------
    -- Insert data into RoomLocation
    INSERT INTO [Location].[RoomLocation] (
        [RoomNumber],
        [BuildingCode],
        [UserAuthorizationKey],
        [DateAdded],
        [DateOfLastUpdate]
    )
    SELECT DISTINCT
        rd.RoomNumber,
        bl.BuildingID,
        @UserAuthorizationKey,
        @DateAdded,
        @DateOfLastUpdate
    FROM #RoomData rd
    INNER JOIN [Location].[BuildingLocation] bl ON rd.BuildingName =
bl.BuildingName
    WHERE NOT EXISTS (
        SELECT 1
        FROM [Location].[RoomLocation] rl
        WHERE rl.RoomNumber = rd.RoomNumber AND rl.BuildingCode =
bl.BuildingID
    )
    ORDER BY rd.RoomNumber;


         -- Log the action
    DECLARE @RowCount INT = @@ROWCOUNT;
    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();
    EXEC [Process].[usp_TrackWorkFlow]
        @WorkflowDescription = 'Load Room Location Data',
        @WorkFlowStepTableRowCount = @RowCount,
        @StartingDateTime = @DateAdded,
        @EndingDateTime = @EndingDateTime,
        @UserAuthorizationKey = @UserAuthorizationKey;
END;

     -- =============================================
     -- Author:      Group Three
     -- Create date: 05-12-24
     -- Description: Main Procedure, Drops Keys, Truncates Tables,
Loads the Tables from
     --                 Uploadfile.CurrentSemesterCourseOfferings, and
adds the foreign keys.
     -- =============================================
ALTER PROCEDURE [Project3].[LoadStarSchemaData]
-- Add the parameters for the stored procedure here
AS
BEGIN
     SET NOCOUNT ON;
     declare @start as datetime2, @end as datetime2;
     set @start = SYSDATETIME()
     --
     --   Drop All of the foreign keys prior to truncating tables in
the star schema
     --
     EXEC  [Project3].[DropForeignKeys] @UserAuthorizationKey = 1;
     --
     --   Check row count before truncation
     EXEC [Project3].[ShowTableStatusRowCount]
          @GroupMemberUserAuthorizationKey = 2,  -- Change -1 to the
appropriate UserAuthorizationKey
          @TableStatus = N'''Pre truncate of tables'''
     --
     --   Always truncate the Star Schema Data
     --
     EXEC  [Project3].[TruncateStarSchemaData] @UserAuthorizationKey =
2;
     --
     --   Load the star schema
     --
     EXEC  [Project3].[LoadDepartments] @UserAuthorizationKey = 3;
     EXEC  [Project3].[LoadInstructor] @UserAuthorizationKey = 4;
     EXEC  [Project3].[LoadCourse] @UserAuthorizationKey = 2;
     EXEC  [Project3].[LoadModeOfInstruction] @UserAuthorizationKey =
2;
     EXEC  [Project3].[LoadBuildingLocation] @UserAuthorizationKey =
5;
     EXEC  [Project3].[LoadRoomLocation] @UserAuthorizationKey = 4;

     EXEC  [Project3].[LoadClass] @UserAuthorizationKey = 3;
     --   Recreate all of the foreign keys prior after loading the
star schema
     --
     --
     --   Check row count before truncation
     EXEC [Project3].[ShowTableStatusRowCount]
          @GroupMemberUserAuthorizationKey = 1,  -- Change -1 to the
appropriate UserAuthorizationKey
          @TableStatus = N'''Row Count after loading the star
schema'''
     --

     EXEC [Project3].[AddForeignKeys] @UserAuthorizationKey = 1;
-- Change -1 to the appropriate UserAuthorizationKey

     declare @rowcount as int
    set @rowcount = 0
    set @end = SYSDATETIME()
    EXEC [Process].[usp_TrackWorkFlow]
       @WorkFlowDescription = N'Loaded All Data',
       @WorkFlowStepTableRowCount = @rowcount,
       @StartingDateTime = @start,
       @EndingDateTime = @end,
          @UserAuthorizationKey = 1

--
END;

     -- =============================================
     -- Author:      Group Three
     -- Create date: 05-12-24
     -- Description: Counts the rows of each of the tables
     -- =============================================
```

```sql
ALTER PROCEDURE [Project3].[ShowTableStatusRowCount]
      @GroupMemberUserAuthorizationKey int,
      @TableStatus NVARCHAR(30)

AS
BEGIN
      SET NOCOUNT ON;
      SELECT TableStatus = @TableStatus,
            TableName = 'CollegeClasses.Class',
            [Row Count] = COUNT(*)
    FROM CollegeClasses.Class
    UNION ALL
    SELECT TableStatus = @TableStatus,
            TableName = 'CollegeClasses.Course',
            [Row Count] = COUNT(*)
    FROM CollegeClasses.Course
    UNION ALL
    SELECT TableStatus = @TableStatus,
            TableName = 'Departmental.Instructor',
            [Row Count] = COUNT(*)
    FROM Departmental.Instructor
    UNION ALL
    SELECT TableStatus = @TableStatus,
            TableName = 'CollegeClasses.ModeofInstruction',
            [Row Count] = COUNT(*)
    FROM CollegeClasses.ModeOfInstruction
    UNION ALL
    SELECT TableStatus = @TableStatus,
            TableName = 'Departmental.Department',
            [Row Count] = COUNT(*)
    FROM Departmental.Department
    UNION ALL
    SELECT TableStatus = @TableStatus,
            TableName = 'Location.BuildingLocation',
            [Row Count] = COUNT(*)
    FROM [Location].BuildingLocation
    UNION ALL
    SELECT TableStatus = @TableStatus,
            TableName = 'Location.RoomLocation',
            [Row Count] = COUNT(*)
    FROM [Location].RoomLocation
    UNION ALL
    SELECT TableStatus = @TableStatus,
            TableName = 'DbSecurity.UserAuthorization',
            [Row Count] = COUNT(*)
    FROM [DbSecurity].UserAuthorization
    UNION ALL
    SELECT TableStatus = @TableStatus,
            TableName = 'Process.WorkflowSteps',
            [Row Count] = COUNT(*)
    FROM [Process].WorkflowSteps;

END;

      -- ============================================
      -- Author:      Group Three
      -- Create date: 05-12-24
      -- Description: Truncates the tables after their foreign keys
have been dropped
      -- ============================================
ALTER PROCEDURE [Project3].[TruncateStarSchemaData]
@UserAuthorizationKey int
AS
BEGIN
      -- SET NOCOUNT ON added to prevent extra result sets from
      -- interfering with SELECT statements.
      SET NOCOUNT ON;
      DECLARE @StartingDateTime DATETIME2 = SYSDATETIME();

      TRUNCATE TABLE [CollegeClasses].Class;
      TRUNCATE TABLE [CollegeClasses].Course;
      TRUNCATE TABLE [CollegeClasses].ModeOfInstruction;
      TRUNCATE TABLE [Departmental].Instructor;
      TRUNCATE TABLE [Departmental].Department;
      TRUNCATE TABLE [Location].BuildingLocation;
      TRUNCATE TABLE [Location].RoomLocation;
      TRUNCATE TABLE [Process].[WorkflowSteps];


      DECLARE @WorkFlowStepTableRowCount INT;
    SET @WorkFlowStepTableRowCount = 0;
    DECLARE @EndingDateTime DATETIME2 = SYSDATETIME();

      EXEC [Process].[usp_TrackWorkFlow] 'Drop Foreign Keys',
                                        @WorkFlowStepTableRowCount,
                                        @StartingDateTime,
                                        @EndingDateTime,
                                        @UserAuthorizationKey;

    EXEC [Process].[usp_TrackWorkFlow] 'Truncate Data',
                                        @WorkFlowStepTableRowCount,
                                        @StartingDateTime,
                                        @EndingDateTime,
                                        @UserAuthorizationKey;
end
```